特輯 技術解說

인공지능 기술의 동향

林 英 煥
韓國電子通信研究所 人工知能研究室長

Ⅰ. 서 론

인공지능(AI, artificial intelligence)이란 컴퓨터를 이용한 문제 해결에 경험적인 지식을 어떻게 잘 이용할 수 있을까 하는 방법을 연구하는 분야라고 할 수 있다. 즉 인공지능에서 가장 중요한 핵심적인 요소는 지식을 이용한다는 점이다. 컴퓨터가 지능(intelligence)을 가지고 있다는 것은 무엇인가? 우선 실시간(real time)으로 수행하여 예상치 못한 상황의 입력에도 적응할 수 있고, 주위 환경에서 새로운 지식을 배워 시간이 지나감에 따라 발전이 있어야 한다. 그러므로 인공지능 기술이 잘 적용될 수 있는 분야는 기존의 시스템에 지능(intelligence)을 부여하는 분야이다. 또 다른 인공지능의 특징은 문제의 성격상 인간만이 해결할 수 있는 분야, 혹은 문제를 해결할 수 있는 알고리즘(algorithm)이 존재한다 할지라도 기존의 기계로는 기억장치의 부족이나 수행시간이 너무 오래 소요되기 때문에 해결할 수 없는 분야를 해결해 보려는 것이다. 이를 위하여 인간의 경험적인 지식을 이용하려는 것이고 최선의 해결책을 찾으려하기 보다는 받아들여질 수 있는 대안을 찾으려는 시도이다.

이러한 인공지능 기술은 전산학이 존재하기 이전부터 연구되어 왔지만 할때는 절망적인 시기도 있었다. 최근 인공지능 기술을 이용한 시스템이 실제 생활에 활용할 수 있게 됨에 따라 인공지능에 대한 연구가 세계 여러나라에서 경쟁적으로 이루어지고 있지만 우리나라에서는 극히 초보적인 단계에 있다. 일본의 5세대 컴퓨터 사업에 영향을 받아 국내에서도 인공지능에 대한 인식은 많이 하고 있지만 그에 대한 부작용으로 인공지능이면 모든 것을 해결해 줄 수 있는 것처럼 오해하고 있는 것도 사실이다. 한편으로는 기대한 만큼 새롭고 환상적인 제품이 나오지 않기 때문에 실망하고 있는 것도 사실이다. 그러나 현재 인공지능 기술은 실용적인 기술로서 인정을 받아 학교나 연구소의 실험실에서 머물던 시대를 지나 기존의 개발 제품에 통합됨으로써 지능(intelligence)을 제공해 줄 수 있는 핵심기술로 산업체에 정착되어 있는 단계이다.

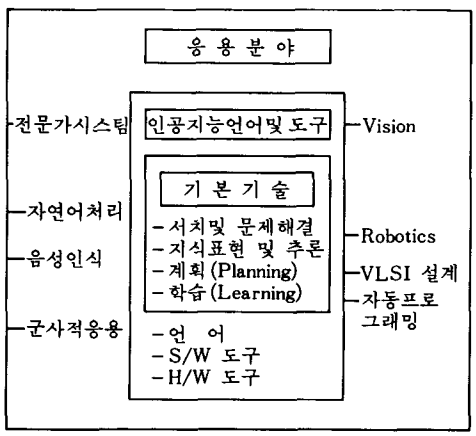

그림 1. 인공지능 기술의 개념도

여기서는 인공지능의 기본기술이 무엇이며 그것이 어떻게 이용되고 있는지 살펴 보았다. 인공지능 전 분야는 그림 1에서 처럼 계층적으로 구분해 볼 수

있다. 인공지능 기본기술은 지식을 컴퓨터에 표현하고 처리하는 기술에 관한 것이고, 인공지능 언어 및 도구는 기본기술을 응용시스템에 도입할 때 필요한 소프트웨어 도구 및 하드웨어 도구에 대한 것이다. 그리고 응용분야로는 최근에 상품화 되어 나온 제품과 연구소나 학교에서 수행하고 있는 과제 중심으로 전개했다.

Ⅱ. 인공지능 기본 기술

인공지능은 경험적인 지식을 이용하는 학문이다. 여기 기본기술에는 지식을 이용하여 문제를 해결하기 위한 표현방법 및 추론과 계획수립(planning)그리고 주위환경과 경험에서 배우는 학습(learning) 기술에 대하여 살펴 보았다.

1. 서치(search) 및 문제해결(problem solving) 기술

인공지능 프로그램은 주어진 문제를 해결하기 위하여 여러가지 접근방식이 고안되었는데 다음 3가지 방법이 대표적인 것이다.

- 상태공간방법(state-space approch)
- 문제축소방법(problem reduction approch)
- 정리증명방법(theorem proving approch)

이러한 문제 해결 시스템은 기본적으로 문제표현(problem description), 연산자(operator)집합, 그리고 제어전략(control strategy)등 세가지 기본요소로 구성되어 있다. 인공지능이 실생활에 적용됨에 따라 위의 세가지 방법은 생성시스팀(production system)이라는 정형화된 시스팀으로 발전되었다.

1) 상태공간 방법(state-space approach)

이 방법은 문제를 상태(state)로 표현하고 연산자는 문제가 나타낼 수 있는 모든 경우에 대하여 적용될 수 있는 상태를 변환시키는 연산자이다. 문제를 해결한다는 것은 초기상태(initial state)에서 적용할 수 있는 모든 연산자를 적용시켜 새로운 상태로 변환해가면서 목적상태(goal state)를 찾는 것이다. 이 과정에서 제어 전략과 서치방법은 모든 경우의 수를 전부 다 발생시키지 않고도 목적 상태를 효과적으로 찾으려 하는 것으로 A *알고리즘이 대표적인 것이다. 이 방법은 순방향 추론 방법에 해당하는 것으로 문제 해결을 일반적인 그래프(graph)로 모델화하여 연구되어 왔다.

2) 문제축소(problem reduction approch)

문제축소 방법은 주어진 문제를 해결하기 보다 쉬운 부문제(subproblem)로 축소시키는 과정을 반복하여 문제해결이 쉽거나 혹은 이미 해결된 문제인 원시적 문제(primitive problem)의 집합으로 축소시켜 해결하는 방식이다. 문제축소 방식은 AND/OR 그래프로 일반화되어 제어전략을 연구하는데 대표적인 것으로 A *알고리즘이 있다. 이 방식의 특수한 경우로 게임트리를 구성할 수 있는데, 이것은 어떤 게임에 있어서 가장 그럴듯한 다음 수를 찾는데 이용된다. 대표적인 서치방법으로는 미니맥스(minimax) 서치와 알파베타($\alpha\beta$)서치방법이 있다. 문제 축소방식은 원칙적으로 역방향 추론에 해당된다.

3) 정리증명 방법(theorem proving approch)

이 방법은 문제를 해결하는 것은 정리를 증명하는 것과 마찬가지라는 관점에서 출발한다. 즉 문제의 정의나 공리 그리고 가정 및 증명하고자 하는 정리를 논리적인 공식으로 나타내고, 정리가 문제의 논리적인 결과(logical consequence)인지 증명하는 방식이다. 대표적인 것으로는 술어논리를 이용한 표현방식과 분해방법을 이용한 추론 방법이 잘 개발되어 있다.

2. 지식표현 및 추론 방법(knowledge representation and inference mechanism)

인공지능에 대한 중요성이 점차 인식되고 문제 표현 방식이 문제해결의 효과에 중요한 영향을 미친다는 것이 밝혀짐에 따라 이 분야에 대한 연구가 최근 10여년간 활발하게 이루어져 왔다. 지식표현은 단순한 지식 자체의 표현 뿐만 아니라 표현된 지식을 처리하는 방법도 동시에 연구되어야 한다. 즉 지식표현은 자료구조와 그것을 해석하여 컴퓨터가 지적인 행위를 할 수 있게 하는 해석과정의 결합이라 할 수 있다. 인공지능에서 다루는 지식은 주로 다음 네가지로 객체에 대한 지식, 사건에 대한 지식, 성능에 대한 지식, 그리고 지식을 위한 메타지식 등이다. 지식표현을 위한 원리로는 크게 두가지, 점차적(proceduaral) 표현방식은 전통적인 알고리즘이나 프로시져를 지식으로 표현한 것과 마찬가지로서 각 선언적 지식간의 관계를 정의해 주는 지식을 나타내는 방식이다. 선언적 표현 방법으로는 서술논리(predicate calculus)방식, 생성 규칙(production rule), 의미네트워크(semantic network), 프레임(frame)이나 스크립(script) 등이 있다.

1) 논리적 표현방식

인간의 지식을 표현하는 방식으로 논리(logic)는 이

미 철학자나 수학자들이 오래전부터 사용하던 것으로 지식베이스는 논리적 공식(logical formular)의 집합이다. 논리적 표현방식에는 현재 알고 있는 지식에서 다른 새로운 지식을 추측할 수 있는 추론 규칙이 잘 연구되어 있으며 표현방식이 단순하고 그 의미가 분명하다. 그러나 지식베이스를 구성하는데 조직화 할 수 없다는 단점도 있다. 논리적 표현방식은 주로 일차술어논리(first order logic)를 많이 이용해 왔으며 현재 지식습득(knowledge acquisition), 믿음(belief), default 등의 문제로 확장되면서 더욱 중요한 분야로 연구되고 있다. 논리를 이용한 방법의 예로 STRIPS와 Omega 등이 있다. STRIPS(Stanford Research Institute Problem Solver)는 로보트가 물체를 재배치하고 어지럽게 늘려져 있는 환경에서 돌아다니는 문제를 계획하고 해결하는 것을 목적으로 하는 시스템이다.

2) 의미 네트워크(semantic network) 표현방식

이 방식은 의미가 있는 노우드(node)와 아크(arc)로 구성된다. 노우드는 해당분야의 개체, 개념, 상황 등을 나타내며 아크는 그들간의 관계를 표현한다. 이런 의미 네트워크는 60년대 말부터 심리학자들에 의하여 인간 기억의 심리학적 모형으로 개발되어 왔다. 의미 네트워크가 갖는 가장 큰 특징은 상속기법이 쉽게 제공되는 것으로 논법이 주로 복잡한 분류(taxonomy)에 바탕을 둔 문제는 이 방식이 자연스러운 표현 방식이다. 그러나 이 방식의 문제점은 표현된 구조가 무엇을 의미하는가 하는 정형적인 의미가 없다는 점이다. 네트워크 구조에 주어지는 의미는 그 네트워크를 다루는 과정의 성격에 의해서 결정되고 추론은 주로 매칭(matching)에 의해서 이루어진다. 이 방식을 이용한 시스팀의 예로는 1968년 Raphael의 SIR 프로그램이 최초의 것이고 1976년 Walker가 SRI에서 주제영역을 표현하는데 이 방식을 사용했으며, BBN의 Woods는 자연어 처리에서 이 방식을 사용했다. Mylopoulos는 동료들과 TORUS 시스템을 위한 PSN 이란 네트워크를 개발하였으며 MIT의 Winston은 시각적 지식을 표현하는데 일종의 의미네트워크를 사용하였다. Bradhman은 구조적 상속 네트워크라는 보다 복잡한 구조의 의미 네트워크를 사용하여 KL-1 언어를 개발하였다.

3) 프레임(frame)과 스크립(script)

MIT의 M. Minsky가 1975년 프레임이론을 발표한 이후 이 방식은 지식표현의 수단으로 가장 널리 쓰이게 되었다. 프레임은 일단의 slot으로 구성으로 있으며 각 slot은 프레임이 나타내고자 하는 개념 기술의 한 측면을 나타낸다. 이 slot의 값은 전형적인 값을 갖거나 특정상황에 맞는 특정한 값을 나타낸다. 이러한 프레임을 바탕으로 과거 경험에서 얻어진 개념을 통해서 새로운 데이터를 이해하는 구조가 제공된다. 한편 프레임은 전형적인 개념을 나타내고 비교에 의해 특정한 개념을 표현할 수 있기 때문에 심리학에서 발전된 원형(prototype) 이론을 쉽게 반영할 수 있다. 즉 일반적인 프레임은 바로 객체의 원형을 나타내며 실증화는 이 원형과의 비교에서 이루어지는 것이다. 이것이 프레임 이론이 각광을 받는 가장 큰 장점 중의 하나이다. 스크립(script)은 프레임과 유사한 지식표현 수단으로 특정한 문맥에서 전형적인 사건의 연속을 기술하기 위한 구조이다. 스크립은 프레임보다 복잡한 구조로서 실제 세계에서 사건의 발생에 대한 유형을 갖고있기 때문에 사건의 연속관계에 대한 기술에 더욱 유용하다. 즉 스크립은 사건이 어떻게 서로 관계를 가지면 어떤 순서로 이루어지는가를 나타낼 수 있는 장점이 있다. 스크립은 Yale 대학의 R. Schank 팀이 주로 자연어 처리와 이해를 위한 프로그램에 지식표현 수단으로 사용하였다. 그 대표적인 것으로 SAM이 있다. 프레임은 발표된 이래 Borbrow 팀이 GUS라는 자연어 이해 시스팀을 개발하는데 이용되었다. 이와 동시에 Bobrow와 Winograd는 프레임에 바탕을 둔 프로그래밍 언어로서 KRL(knowledge representation language)를 개발하고 여러분야에 실험적으로 적용시켜 보았다. 또다른 이용으로 Goldstein과 Roberts의 NUDGE 시스템을 들 수 있다. 이 시스팀은 불완전하고 모순이 있는 경영 예정 요구를 이해하여 기존의 예정 알고리즘에 대한 완전한 명세를 제공하는데 사용되었다. 이때 사용된 지식표현 언어는 FRL(frame representation language)로서 1977년 MIT의 R. Bruce Roberts와 Ira P. Goldstein에 의해 개발된 것이다.

지금까지 최근 지식표현의 몇가지 방법에 대하여 살펴보았다. 최근 지식 표현의 경향은 지식베이스를 일의 성격에 따라 분류하고 각기 일의 성격을 가장 잘 대변할 수 있는 표현 방법으로 나타내고 그 위에 여러가지 표현기법을 조정하는 소프트웨어 도구가 많이 개발되고 있다.

3. 계획수립(planning)

계획수립이라는 것은 어떤 일을 실행하기 이전에

문제해결 행위의 전과정을 결정하는 것을 의미한다. 결국 계획은 행위과정에 대한 표현이다. 계획수립의 장점은 탐색량을 줄이고 목적의 상충을 해결하며 오류 회복의 기반을 마련하는데 있다.

인공지능에서 계획수립에 대한 접근 방식은 크게 네 가지로, 비계층적(nonhierarchical), 계층적(hierarchical), 스크립(script), 기회적(opportunistic) 계획수립으로 분류할 수 있다.

비계층적 계획수립이라는 것은 문제의 각 목적을 성취하기 위하여 문제해결 행위의 연속을 개발하는 것이다. 이를 위하여 원래의 목적을 보다 단순한 부목적으로 축소시키거나 현 상황과 목적상황과의 차이를 줄이기 위한 meanends 분석 방식을 사용하기도 한다. 대표적인 비계층 계획자료는 STRIPS, HACKER, INTER PLAN 등이 있다.

계층적 계획수립 방식은 먼저 계획은 모호하지만 완전하게 윤곽을 잡은 뒤 이 모호한 부분을 좀더 상세한 부분계획으로 기술하며 이를 상세한 문제해결 연산자의 연속으로 구성될 때까지 반복한다. 이 방식의 장점은 처음부터 계획을 지나치게 상술할 필요가 없다는 것이다. 예로서 ABSTRIPS, NOAH, MOLGEN 등이 있다.

세번째 스크립(script) 계획 방식은 계략적인 계획은 사용하지만 계층적 계획수립과는 달리 개략적으로 생성되는 것이 아니라 계획을 저장한데서 호출하는 방식으로 이루어졌다. 이 방식은 MOLGEN의 한 부분에서 이용되었다.

네번째 접근방식은 Barbara Hayes-Rothe와 Friedrick. Hayes-Roth가 인간의 계획수립 방식에서 발견한 기회적 방식이다. 이 방식은 인간계획 수립을 모형화하기 위한 blackboard 라는 제어구조를 채택하였다. Blackboard는 계획단계에 대해 제어하는 것으로 이 제언은 계획수립을 잘하는 전문가가 만든다. 각 전문가는 특정한 종류의 계획수립을 결정하도록 하지만 특별한 순서로 운영하지는 않는다. 즉 어떤 이유가 발생했을 때에만 계획수립의 결정을 내리게 된다.

4. 컴퓨터 학습

컴퓨터가 단순히 지시한 일만 따르지 않고 새로운 것을 배우고 새로운 상황에 적응하게 될 때 비로소 컴퓨터가 지능을 가졌다고 할 수 있다. 일반적인 학습에는 크게 두가지 형태로, 영어나 수학을 배우는 것처럼 "지식의 습득"형태와 피아노를 배우는 것처럼 반복을 요하는 "숙련"이 주가 되는 형태가 있다. 컴퓨터 학습의 분야는 "지식의 습득"에 중점을 두는 것으로 다음과 같은 방법이 있다.

1) 단순한 암기에 의한 학습

이 방법은 컴퓨터가 경험하는 것과는 관계없이 단순히 외부에서 넘겨주는 지식을 기억하고 있다가 그와 꼭 같은 상황이나 조건을 만났을때 꺼내 쓰는 가장 간단한 방법이다. 이 방법은 Samual의 체커 프로그램에 사용하여 어느정도 효과를 거두었지만 알고 있는 방법을 응용할 수 없다는 단점이 있다.

2) 가르침에 의한 학습

이 방법은 교사나 트레이너가 컴퓨터를 가르치는 것으로 컴퓨터는 배우는 입장에서 입력된 지식을 스스로 이용 가능한 형태로 변환시켜 간다. 그리고 새로운 지식을 효과적으로 사용하기 위해서 기존의 지식과 서로 상충되지 않도록 잘 결합하여야 한다. 그러나 학습자인 컴퓨터가 스스로 추출해서 배울 수 있는 양이 그리 많지 않으며 교사의 부담이 크다는 점이 있다.

3) 유사성을 이용한 학습

이 학습방법은 새로운 개념이나 기술과 매우 유사한 기존의 지식을 새로운 상황에 유용한 형태로 변환시켜 새로운 지식을 습득케 하는 방법이다. 유사성에 의해 배우는 컴퓨터는 자기가 가지고 있는 기존 프로그램을 약간만 변형함으로 다른 문제를 해결할 수 있기도 하다.

4) 예를 통한 학습

이는 어떤 개념에 대한 궁극적인 예와 부정적인 예만으로 구성되어 있을 때 이들로부터 궁극적인 예는 다 포함하지만 부정적인 예는 하나도 포함될 수 없는 그런 일반화된 하나의 개념을 스스로 도출함으로 컴퓨터가 배우는 방법이다. 이 방법은 교사에 의해 일반개념 자체가 바로 주어지지도 않고 유사성을 이용할 수 없기 때문에 더 많은 추론을 해야 한다.

5) 발견에 의한 학습

이는 가장 높은 차원의 학습으로 수학처럼 정형적이고 체계적인 영역에서만 응용할 수 있다. 이 방법은 이미 알려진 사실과 시도해 보지 않았던 방법을 결합하여 새로운 사실을 밝혀 낸다. 이 경우 과거의 사실로 부터 새로운 결론을 끌어낼 수 있는 통찰력이 필요하나 이는 인간의 인식분야에서도 분명하게 규정되어지지 않은 부분이다. 현재의 인공지능 시스템에서는 광범위한 지식의 습득이나 일반적인 문제풀이 기술이 충분하지 못하기 때문에 가까운 장래에

실현하기는 힘들다.

5. 인공지능 언어 및 도구

인공지능 프로그램이란 경험적인 지식을 문제해결에 이용하는 것으로 주로 부호조작(symbolmanipulation)이나 규칙(rule)을 이용하는 프로그램이다. 이와 같은 프로그램을 개발하기 위한 환경으로 고급 프로그래밍 언어, 소프트웨어 도구와 같은 프로그램이나 LISP 기계와 같은 하드웨어가 필요하다.

1) 인공지능용 언어

인공지능 언어는 그림 2에서 보는 바와 같이 C, LISP, PROLOG와 같은 고급 프로그래밍 언어에서부터 KEE나 EMYCIN과 같은 전문가 시스템 구축 도구에 이르기까지 다양하다. 그러나 여기에서 주의해야 할 점은 LISP나 PROLOG, 혹은 소프트웨어 도구를 이용하여 프로그램했다고 해서 전부가 인공지능 프로그램이 되는 것은 아니다. 인공지능 프로그램은 FORTRAN이나 COBOL, C, 심지어는 어셈블리어로도 개발할 수 있다. 중요한 점은 개발하는 도구에 있는 것이 아니고 프로그램 자체가 문제해결을 위하여 경험적인 지식을 이용했느냐 하는 점이다. 인공지능 언어는 단지 하나의 도구로서 프로그램의 생산성을 향상시키거나 효과적으로 수행하기 위하여 선택된 것일 뿐이다.

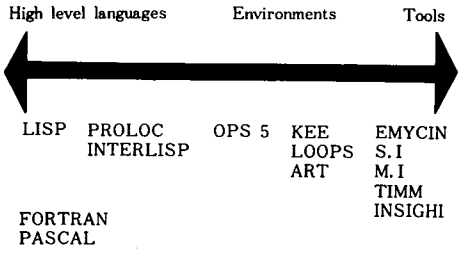

그림 2. The language-tool continum

한편 인공지능 응용분야를 보면 초기에는 부호조작을 이용하는 것이 주류를 이루었기 때문에 부호처리에 용이한 LISP가 많이 사용되었다가 지식을 이용하는 방법이 달라짐에 따라 규칙을 처리하기에 편리한 PROLOG도 주의를 끌고 있다. 또 한편 기존의 개발된 소프트웨어와의 호환성을 고려하여 C나 FORTRAN도 많이 사용되고 있다.

(1) LISP

부호처리에 적합한 최초의 언어는 1957년도에 Carnegie-Mellon대학의 Newell과 Shaw, 그리고 Simon에 의하여 개발된 IPL이다. 이듬해 IPL의 list 처리방법을 이어받은 LISP가 McCarathy에 의해 개발되었다. 순수한 함수언어로서 LISP를 살펴보면 LISP가 처리하는 데이타 구조는 list이고 이것을 다루는 CAR, CDR, CONS와 같은 몇개의 매우 간단한 명령어로 이루어져 있다. 이외에 LISP의 구성요소로, 조건적으로 수행하는 COND와 새로운 명령어를 정의할 수 있는 기능이 있고 프로그램하기 쉽도록 여러가지 새로운 기능이 추가 되었다. LISP가 인공지능언어로서 인정을 받게된 배경은 첫째 대화식으로 프로그램할 수 있어 터미널에서 자유로이 LISP 프로그램이나 데이터를 변경시킬 수 있다는 점이다. 둘째 개발환경에 관한 것으로 프로그램하기 편리한 소프트웨어 도구가 LISP로 많이 개발되어 있다는 점이고 list processing 이라는 원어가 말해주듯이 부호처리에 적합하다는 점이다. 마지막으로 LISP는 프로그램과 데이터가 같은 형식으로 나타나고 데이터도 프로그램처럼 수행될 수 있다는 점이다.

(2) PROLOG

PROLOG는 서술논리(predicate logic)에 기반을 둔 언어로서 Kowalski가 기본개념을 제안하였고, 1972년 Marseille 대학에서 Colmerauer와 Roussell에 의해 처음으로 제작되었다. 그 후 Pereira와 Warren은 DEC-10에 컴파일러를 개발하였으며 유럽 각지에서 확장 발전되어 오다가 최근 일본에서 제5세대 컴퓨터의 핵심언어로 채택되면서 각광을 받기 시작했다. 프로그래밍 언어로서 PROLOG는 종래의 컴퓨터 언어와는 달리 논리에 기반을 둔 언어이다. 즉 논리를 문제표현 도구로 이용하고 질문에 답하기 위해 논리의 추론방법을 이용하였다. 일반적으로 PROLOG 프로그램은 지식베이스(프로그램), 추론 방법, 그리고 제어전략이 통합된 일종의 생성시스템(production system)이다. 사용자는 제어전략이나 추론방법에 대하여 알 필요 없이 단지 지식베이스인 PROLOG 프로그램을 선언하기만 하면 된다. PROLOG가 인공지능 프로그램에 적합한 점은 첫째 규칙을 처리하기가 용이하다는 점이다. 둘째 PROLOG는 일반 컴퓨터 언어와는 달리 문제해결 방법에 대하여 프로그램하는 것이 아니고 문제를 PROLOG 언어로 묘사하면 된다는 점이다. 마지막으로 추론기능을 포함하고 있어 주어진 사실과 규칙들간의 관계를 이용하여 새

로운 규칙만 정의하면 된다는 점으로 지식 베이스의 지식만 수정하면 프로그램 전체를 수정하지 않고서도 확장할 수 있다는 점이다. 그러나 PROLOG는 제어기능의 비효율성과 지식을 표현하는 능력에 한계점을 지니고 있기 때문에 여러 방향으로 수정보완되고 있으며 현재는 PROLOG 전용기계를 제작하기 위한 연구도 진행중이다.

2) 소프트웨어 도구

앞절에서 살펴본 고급 프로그래밍 언어만으로는 많은 양의 지식을 처리하는 응용시스템을 개발하는데 불편할 뿐만 아니라 실험모델(prototype)을 빨리 개발해야 하는데 적합하지 않으므로 EMYCIN이나 KEE와 같은 소프트웨어 도구가 등장하게 되었다. 소프트웨어 도구는 주로 전문가시스템 개발용 패키지들이 대부분이다. 전문가시스템은 현재 인공지능 분야에서 가장 강조되고 있는 분야중의 하나로서 대부분의 인공지능 응용프로그램이 일종의 전문가 시스템이거나 혹은 지식기반 시스템이라고 생각할 수 있다. 전문가시스템이란 지능적인 컴퓨터 프로그램으로서 전문적인 지식을 가진 인간만이 해결할 수 있는 정도의 어려운 문제를 풀기 위하여 인간의 경험적인 지식이나 추론과정을 이용하는 것이다. 전문가시스템은 그림3에서와 같이 지식베이스와 추론방법, 그리고 워킹메모리로 구성되어 있다. 그러나 전문가시스템의 승패는 얼마나 정교한 추론방법과 지식표현방법 등을 사용하느냐에 있는 것이 아니고 얼마나 좋은 지식을 모을 수 있느냐에 달려있다.

현재 사용 가능한 전문가시스템은 수도 헤아릴 수 없이 많이 나와 있다. 그 중에도 Teknowledge사의 M.1과 S.1이 있다. M.1은 지식공학의 실제 적용을 위한 소프트웨어 도구로 IBM PC에서 수행되도록 개발되었다. S.1은 보다 광범위하고 전문적인 지식시스템을 개발하기 위한 복합적 소프트웨어로 Xerox 1100과 1108과 1108 workstation에서 수행된다. Intellicorp사는 인공지능 응용을 위한 대표적인 도구의 하나로 KEE(knowledge engineering environment)시스템을 개발하여 제공하고 있으며 Inference사는 ART(automated reasoning tool)를 제공하고 Xero사는 loops 시스템을 제공하고 있다.

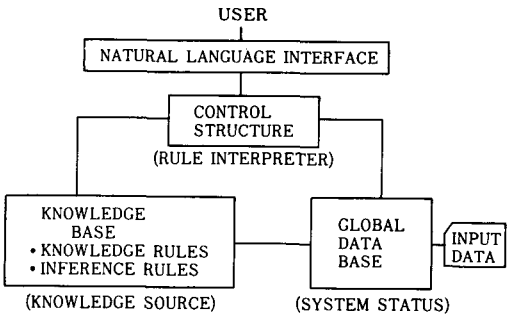

그림 3. Basic structure of an expert system

3) 하드웨어 도구

인공지능 언어는 전통적인 컴퓨터의 Von-Neumann 구조에서는 효과적으로 수행될 수 없으므로 LISP나 PROLOG를 위한 전용기계가 개발되어 보급되고 있다.

(1) Symbolics 3600

Symbolics, Inc.의 이 시스템은 1983년에 소개된 것으로 MIT의 AI Lab의 연구를 바탕으로 개발된 시스템이다. 중앙처리장치는 고성능의 Lisp 전용 처리기이고 가상기억장치가 있으며 30 메가 바이트의 주기억 장치와 64K MOS ROM을 가지고 있다. 입출력 보조시스템으로 MC68000을 이용한 전위처리기기가 있어 Ethernet, SMD 디스크, 고상도 흑백화면과 같은 고속 입출력 장치를 처리한다. 주요한 소프트웨어로는 ZetaLisp, Flavor에 입각한 윈도우 시스템, INTERLISP와 호환적인 패키지, 강력한 대화식 소프트웨어가 있다.

(2) LAMBDA

LAMBDA는 다중처리기 구조를 가지고 있어 LISP는 LISP 전용처리기에서 수행되고 다른 68000 처리기는 시스템 진단장치(SDU)로서 사용된다. 이러한 부품들은 NuBus로 연결되고 multibus와의 연결도 가능하다. 현재 LAMBDA 시스템은 개인용 뿐만 아니라 다수의 사용자가 이용할 수 있는 LAMBDA 2×2/PLUS, 그리고 LAMBDA 4×4 시스템을 계속 발표하고 있다. 또한 병렬처리를 위한 LISP나 ZETA-LISP, 그리고 고속으로 PROLOG를 수행할 수 있는 LM-PROLOG 등을 갖추고 있다.

(3) EXPLORER

이 기계는 Texas Instruments 사에서 개발된 32비트 개인용 LISP 전용 마이크로 프로세스를 이용하여 책상밑에도 설치할 수 있을 정도로 규모를 줄이는데 성공한 기계이다. 소프트웨어는 고해상의 그래픽스 윈도우, 마우스 등을 기반으로하여 객체지향 프로그래밍 기법을 COMMON LISP에 결합하였고

MIT의 LISP 기계의 많은 라이브러리 프로그램과 결합되어 있다.

(4) SUN workstation

Sun Micro사에서 제공하는 SUN workstation에는 인공지능 응용을 위하여 LISP이나 PROLOG를 제공하고 있다. 이 시스팀은 4.2 BSD UNIX OS를 기반으로 하는 것으로 전통적인 소프트웨어와 호환성이 필요한 인공지능 프로그램을 개발하기 위하여 적합한 workstation이다. 최근에 LISP나 PROLOG의 처리속도를 높이기 위하여 플로팅포인트 가속기와 같은 가속기를 제공하기도 한다.

Ⅲ. AI 컴퓨터 아키텍춰

컴퓨터가 인간이 사고하는 방식과 유사하게 추론하고 동작되도록 하려면 컴퓨터에 인간이 소유하고 있는 지식과 유사한 지식(인공지능)을 넣어주고 이 지식을 빠르고 적절하게 처리할 수 있도록 하드웨어를 구성하여야 한다. 그러나 기존의 컴퓨터 시스템에 인공지능을 구현하기에는 많은 문제점이 대두되고 있다. 따라서 최근에는 인공지능 처리에 적합한 컴퓨터 아키텍춰에 대한 많은 연구가 수행되고 있다.[23,30]

본 절에서는 AI 아키텍춰가 제공해야 하는 주요 기능에 대해 알아보고 기존의 AI 아키텍춰들을 언어기반형(language - based) AI 시스템과 지식기반형(knowledge-based) AI 시스템으로 분류하여 각각 아키텍춰들의 특성에 대해 알아보기로 한다.[23]

1. AI 아키텍춰의 주요 특성

AI의 응용분야는 기존의 컴퓨터 시스템의 응용분야와 다소 차이가 있고 주요 연산 또한 다르다. 따라서 AI 아키텍춰는 인공지능의 응용 분야에 대한 프로그램들이 갖는 특성들을 고려하여 적절히 설계되어야 한다.

AI 처리라 함은 데이타 및 지식의 표현방법에 따라 지식을 습득하고 이들을 효율적으로 이용하여 원하는 답을 얻을 것이라 할 수 있다. 가지고 있는 지식을 이용하여 원하는 답을 얻기 위해서는 논리적 추론을 수행해야 하고 원하는 정보를 축출해야 하는데 이러한 처리는 주로 search 연산에 의해 수행되며, 이를 효율적으로 수행하기 위해서는 하드웨어에 의해 seach 연산 기능이 제공되어야 한다. 또한 이러한 처리는 매우 높은 빈도로 메모리 접근을 수행해야 하며, I/O 빈도도 매우 높게 된다. 따라서 기존의 컴퓨터 아키텍춰로는 프로세스/메모리 사이의 병목현상을 피할 수 없다.[52] AI 아키텍춰는 이러한 병목현상을 피할 수 있도록 설계되어야 하며 비교(comparison), 선택(selection), 정렬(sorting), 패턴 매칭(pattern matching), 논리 집합 연산(logical set operations) 등의 기본적인 연산 기능을 제공해야 한다.

2. 언어기반 AI 머신(language based AI machine)

언어기반 AI 머신은 AI 머신을 설계하는 방법 중에서 보수적인 방법으로서 Lisp이나 Prolog 등과 같은 특정 고급언어만을 효율적으로 지원하기 위하여 하드웨어가 특별히 설계된 머신을 의미한다. 이때 이러한 머신의 지식표현이나 추론 메카니즘은 해당 고급언어로 작성된다. 따라서 여러 종류의 지식 표현 방식이나 휴리스틱 탐색 프로시듀어를 유연성있게 지원한다.

언어기반 AI 머신에서는 지식표현 방식이 해당 언어로 작성되어야 하기때문에 응용 프로그램과 하드웨어 사이에 별도의 소프트웨어 계층을 두어야 한다. 이로인해 수행시간이 늦어지는 단점이 있다. 이러한 언어기반 AI 머신 중에서 Lisp 머신, Prolog 머신에 대하여 특징을 각각 알아보도록 한다.

1) Lisp 머신 : Symbolics 3600 계열

Lisp 프로그램은 function의 집합으로 간주할 수 있고 이들 function을 중심으로 병렬수행이 가능하다. 또한 Lisp 프로그램은 recursive한 특성이 있고 untyped 데이타를 사용하는 특성으로 부터 다음과 같은 Lisp 머신의 기능이 요구된다. 우선 AI 응용의 공통적인 요구로서 대용량의 메모리가 필요하다. 그리고 stack 기능과 function call 기능을 효율적으로 지원하여야 하며, untyped 데이타를 효율적으로 지원할 수 있어야 한다. 이러한 Lisp의 요구로 부터 Lisp 머신의 대표적인 예인 Symbolics 3600 계열에서 구현한 내용에 대하여 알아보도록 한다.[23,33]

Symbolics 시스템은 untyped 데이타를 효율적으로 지원하기 위하여 프래그래머의 관여가 필요없는 자동 타입 검사를 가능하게 하였다. 그리고 새로운 function을 기존의 프로그램에 추가하는 것을 용이하게 하기 위하여 incremental 컴파일을 가능하게 하였다.

또한 function call과 list를 효율적으로 표현하는 방식에 중점을 두었다. Function call을 신속하도록 하는 것은 모듈화된 프로그래밍 기법과 프로그램의 유

지보수에 중요한 의미를 갖는다. Symbolics 시스템에서는 function call을 신속하게 하기 위하여 stack 머신모델을 기반으로 하여 인스트럭션 call을 구현하였다. 그리고 control stack을 접근하는데 1 stack frame당 1 머신 사이클이 소요되도록 구현하였다.

List는 Lisp 프로그램에서 이용되는 기본적인 데이타 표현 방식이므로 신속히 접근이 가능하고 작은 공간에 저장될 수 있는 형태로 구현되어야 한다. Symbolics에서는 cdr 코드로서 2-bit 데이타 태그를 사용하는데 이 데이타 태그를 이용함으로써 list를 저장하는데 필요한 데이타의 양을 반으로 줄일 수 있게 된다.

Symbolics에서는 프로세서 아키텍춰로서 stack-oriented 방법을 채택하고 있는데, stack-oriented 머신은 모든 인스트럭션의 오퍼랜드를 stack에서 가져오는 순수한 stack 머신과는 달리 stack을 이용하지 않고 오퍼랜드를 구하는 것이 가능한 머신을 의미한다. 그리고 Lisp 인스트럭션을 1 머신 사이클에 수행되고, dual bus를 이용하여 오퍼랜드가 동시에 functional unit에 전달되도록 하고 있다. Symbolics의 최대 가상 메모리는 256 Mword이며, 최대 실제 메모리는 7 Mword이다.

2) Prolog 머신 : PIM

Prolog를 기반으로 하는 프로그램의 특징은 주로 untyped 데이타를 허용하고, 수행속도의 향상을 기할 수 있는 다양한 종류의 병렬성을 내포하고 있다는 점이다. Prolog 기반 프로그램이 내포하고 있는 병렬성으로서는 AND 병렬성, stream 병렬성, OR 병렬성 그리고 unification 병렬성 등이 있다. 우선 이들 병렬성에 대해 알아보도록 한다.

AND 병렬성은 논리절에서 AND clause(이때 이 clause들을 각각 goal이라고 한다)들은 동시에 수행하도록 하는 것을 의미한다. 만일 두 goal이 공통적인 변수를 갖고 있다면 이 공통변수에 동일한 값이 할당되어야만 한다. AND 병렬성을 구현하기 위해서는 공통변수에 동일한 값이 할당되었는 가를 검사하는 consistency check가 필요하다. 그러나 이 consistency check는 수행 시에 많은 부담이 되어 병렬성으로 얻은 성능의 향상을 저하시키게 되는 문제가 있다.

Stream 병렬성은 goal 들을 순차적으로 처리하는 것과 AND 병렬성에서 consistency check의 문제점을 절충한 것이다. 즉 공통 변수를 갖는 goal 들을 producer-consumer의 관계로 연결하여 producer goal은 공통변수에 대하여 하나씩의 값을 consumer goal에게 파이프라인 식으로 전달하도록 하는 방식이다.

OR 병렬성은 Prolog 프로그램의 해를 구하는 탐색 트리에서 대체가능한 여러개의 경로를 동시에 추적하면서 해를 구하는 것을 의미한다. 초창기의 연구에서는 OR 병렬성이 AND 병렬성보다 더 많은 병렬성을 제고한다고 판단하였고 consistency check의 부담이 없었기 때문에 OR 병렬성이 선호되었다. 그러나 연구가 진행되면서 OR 병렬성은 고도의 병렬성은 제공할 수 없다고 판단하고 있다.[36]

Unification 병렬성은 Prolog 데이타베이스에 있는 clause와 goal clause의 매칭인 unification 작업을 각 변수에 대하여 동시에 수행하는 것을 의미한다. 따라서 변수에 대하여 값을 부여하게 되는 instantiation이 동시에 발생하게 된다.

일본의 제5세대 컴퓨터 프로젝트(FGCS)에서 목표로 삼고있는 멀티컴퓨터인 PIM[19,26]에 대하여 알아보도록 한다. PIM은 AND 병렬성, OR 병렬성, unification 병렬성을 구현하고 있다. PIM은 1000개의 PE(processing element)로 구성되며 전체적으로 100 MLips(logical inference per second) ~1 GLips의 성능을 목표로 하고 있다. PE 간의 통신 부담을 줄이기 위하여 전체 PE는 8개의 PE가 모인 cluster의 집합으로 구성된다. Cluster 내의 PE들은 shared bus를 이용하여 tightly coupled 되어 있다. 그리고 cluster들 끼리는 메시지 패싱 네트워크를 이용하여 서로 연결되어 있다. 또한 cluster 내에서는 병렬 캐싱에 의하여 각 PE의 캐시 메모리와 shared 메모리의 일관성이 보장된다.

각 PE는 untyped 데이타를 구현하기 위하여 태그 아키텍춰를 갖는다. 그리고 프로세스 스위칭을 신속하게 하고 파이프라인 처리를 할 수 있도록 되어있다. PIM은 goal cluase와 지식 베이스의 cluase를 동시에 매칭할 수 있고 변수들을 동시에 instantiation 시킬 수 있는 하드웨어를 제공한다.

3. 지식 기반형 아키텍춰

지식 기반형 아키텍춰는 지식의 표현 및 관리를 위해 채택된 강력한 지식 처리모델을 기준으로 아키텍춰가 설계된다. 이들 지식 기반형 시스템들은 채택된 지식 처리 모델에 따라 다음의 표1과 같이 분류된다.[22] 이들중 대표적인 몇개 시스템의 아키텍춰에 대해 알아보기도 한다.

표 1. 지식기반형 시스템의 분류

지식 표현 방법	Reasoning Technique	대표적인 시스템
Predicate logic	Resolution	FGCS
Data encapsulation	Inheritance	iAPX432
Production systems	Forward chaining	DADO
Frame	Procedural attachments	FAIM
Procedures	Program execution	Zmob
Semantic network	Marker propagation	Connection Machine

1) Rule-based production AI 아키텍춰

Production 시스템은 규칙에 대한 데이타 베이스인 production memory(PM)와 fact들에 대한 데이타 베이스인 working-memory(WM)로 구성되며 다음과 같은 3단계의 처리과정을 반복하는 시스템이다.

○1단계(match) : 규칙의 조건부가 모두 만족되는 규칙들을 찾는다.

○2단계(select) : 1단계에서 선택된 규칙들중 하나를 선택한다.

○3단계(fire) : 선택된 규칙의 then-part를 수행하여 WM의 fact들을 제거, 변경, 첨가한다.

이러한 절차는 순차적으로 수행되며 결론을 얻기까지는 많은 시간이 소요된다. Production 시스템의 하나인 DADO 시스템은 이러한 3단계의, forward-chaining을 직접 하드웨어에 의해 수행하게 한다.[50]

DADO2 시스템은 1023개의 노드가 10-level의 이진 트리 형태로 구성되어 있으며 각 노드는 8-bit 마이크로프로세서이고 64K의 RAM과 하나의 switch로 구성되어 있다. Tree의 root 노드는 규칙들의 한 개를 선택하여 수행된 결과를 subtree에 분산하여 수행되도록 broadcat한다. Subtree는 이들 작업을 동시에 수행하게 됨으로써 speed-up을 얻는다. 그러나 많은 연구원들이 DADO2 형태의 massive parallelism이 production 시스템의 수행을 얼마나 speed-up 할 수 있는가를 연구한 결과 50-100개 가량의 프로세서들에 의해서 speed-up을 최대한 얻을 수 있으며 그 이상으로 프로세서를 늘려도 speed-up을 얻을 수 없음을 알게되었다. 이에 따라 DADO 개발 팀에서는 병렬성을 증가해주는 program 변환 기술을 개발하게 되었다.

미국의 Southern California 대학에서 개발한 SNAP 시스템도 production system의 하나라고 할 수 있다. 이 시스템은 2차원 mesh 구조로 되어 있으며 연상 메모리를 갖고 있다. 또한 마이크로 인스트럭션들을 이용하여 상위 레벨의 인스트럭션들을 제공하고 있다.

2) Semantic network AI 아키텍춰

Semantic network에서 각 노드는 object 혹은 concept를 나타내며 이들 각 노드들 사이의 관계는 두 노드 사이의 연결에 의해 표현된다. Semantic network으로 구성된 지식에서 원하는 정보를 얻기 위해서는 몇개의 기본 연산(집합에 대한 logic 연산, sorting, pattern matching, 새로운 fact를 추론등)들의 반복적인 수행이 요구된다. 이들 연산을 각 노드들 마다 동시에 수행하도록 함으로써 speed-up을 얻을 수 있는데 이상적인 경우는 각 노드당 하나의 프로세서가 할당되고 각 노드들 사이의 관계 표현을 위한 충분한 통신 능력이 제공되어야 한다. 이러한 시스템의 한 예로 connection machine을 들 수 있다.[55]

Connection machine의 각각의 PE(processing element)들은 64K의 memory를 갖고 있으며, 각 PE들은 12-dimensional hypercube 네트워크로 연결되어 있다. 따라서 두 PE사이의 연결 기능 path는 여러 개가 존재하게 된다. 또한 각각의 PE는 하드웨어 router를 갖고 있어 두 노드 사이의 통신을 처리한다.

기타 대표적인 semantic network AI machine 들에 대한 특징들을 표 2에서 볼 수 있다.

표 2. Semantic network AI 머신

시스템 및 개발자	아키텍춰의 주요 특징	비 고
Connection Machine, Thinking Machines, Inc., (1986년 시판)	SIMD hypercube	Connection Machine Lisp(CmLisp)제공 semantic network 지식처리, 병렬 searching 응용 분야
Thistle, CMU	Massively Parallel fine-grained Value & Marker passing	Semantic network 지식처리 low-level vision 처리
SNAP, Univ. of Southern California	2D mesh 구조 Associative Memory cellular array processing	Semantic network 지식처리 discrete 'relaxation for vision

3) Object-oriented AI 아키텍춰

객체 지향형 프로그램은 데이타와 그 데이타를 다루는 모든 절차들을 함께 하나의 통일된 형태(object)

내에 넣어 놓는다. 이들 객체들 사이에 통신이 필요한 경우는 메시지를 해당 object에 보냄으로써 원하는 동작을 수행하게 된다. 이러한 형태의 수행은 변수들에 대한 타입 선언이 수행 중에 dynamic하게 결정되므로 dynamic 데이타 typing 및 빈번한 precedure 호출, 이를 위한 메모리 관리등에 많은 시간이 소요된다.[8] 이러한 주요 기능들을 하드웨어가 보조해 줌으로써 수행을 빨리 하도록 아키텍춰가 설계된 시스템들이 object-oriented AI 머신이다.

이러한 종류의 대표적인 시스템들에 대한 주요 특성들에 대해 표 3에 요약되어 있다.[22]

표 3. Object-oriented AI 머신

시스템명 및 개발자	주 요 특 성
iAPX432 Intel Corp.	Multiprocessor-expandable Ada 및 Object-oriented Lang.
SOAR UC Berkeley	Smalltalk을 위한 단일 프로세서 Register-based expression 수행 tag를 이용한 dynamic type checking
FAIM Fairchild	Hexagonal mesh topology OIL 언어의 수행 Stack-based, parallel tag-checking
Dragon Xerox	Tightly coupled multiprocessor Cedar, Interlisp, Smalltalk Support Stack & Memory-to-Memory 인스트럭션
AI-32 Hitachi	Uniprocessor Stack-based Smalltalk Support

Ⅳ. 휴먼-컴퓨터 인터액션

인공지능은 컴퓨터가 우리들이 일상생활을 해나가는데 도움을 줄 뿐만 아니라 중요한 일을 결정할 때 가장 신뢰성 있는 판단을 내려 줄 수 있는 환경을 제공하는 분야이다. 그러므로 이 분야는 그 어떤 분야보다도 일반 사용자와 밀접한 관계를 갖고 있다. 신체적인 조건, 성격, 문화 환경과 같은 사용자의 개인적인 특성이 인공지능 시스템과 그 응용 프로그램을 설계하고 개발하는데 가장 큰 영향을 미치는 요소 중의 하나인 것이다. 최근 조사에 따르면[40] 응용 프로그램 코드를 작성하기 위한 노력의 50% 이상이 사용자 인터페이스를 구성하기 위한 것임을 감안할 때 얼마나 많은 시간과 노력을 사용자와 컴퓨터 사이에 효율적인 의사 전달 방법에 관한 연구를 하는데 투자하였는가가 인공지능 분야의 성패뿐만 아니라 어느 컴퓨터가 세계 컴퓨터 시장을 점유할 것인가를 결정짓는 요인이 될 것이다. 최근까지만 하더라도 소프트웨어 개발자들은 다음과 같은 이유로 보잘것 없는 사용자 인터페이스를 사용하여 소프트웨어 제품을 개발하여 왔다.[9,10,16,17,43] 첫째로, 인간공학 자체가 모호한 것이었다. 어떠한 것이 진정으로 올바른 인간공학인가 조차도 의견이 모아지지 않았었다. 둘째로, 시스템 설계자와 사용자 사이에 그들이 갖고 있는 지식과 배경이 판이하게 다를 때가 비일비재 하였다. 세째로, 자연어처리와 같은 고도의 기술을 필요로 하는 사용자 인터페이스를 설계하기 위해서는 심리학이나 언어학과 같은 분야에 대한 이해가 요구되지만, 소프트웨어 개발자들은 대부분이 이에 대한 전문 지식이 결여되었었다. 마지막으로 현존하는 소프트웨어 개발방법이나 설계 도구들이 사용자 인터페이스를 설계, 구현하고 평가하기에 부적합하였다.

직감과 제한된 경험에 기반을 두고 임기응변적으로 시스템을 설계해 온 기존 방식은 사용자와 그들이 사용하는 대화방식이 다양화짐에 따라 심각한 문제를 가지게 되었다. 이를 인식한 선진국에서는 최근들어 이러한 문제들을 해결하기 위하여 휴먼-컴퓨터 인터액션 분야에 적극적으로 투자하고 있다. 컴퓨터 분야에 종사하는 전문인들 뿐만 아니라 인지과학, 인간공학, 심리학, 사회과학 등등에 종사하는 전문인들이 하나의 연구 그룹을 조성하여 일반인도 쓰기 쉽고 배우기 쉬운 사용자 인터페이스를 설계하고 개발하기 위하여 많은 노력을 하고 있다. 이들 그룹에서 추구하는 연구 방향은 보다 정교하고 빠른 알고리즘을 실현하는 것보다 인간과 컴퓨터 사이에 존재하는 정보 처리 체계면에서의 많은 차이점을 분석하여 사용자 입장을 더 고려한 기술을 개발하는 것이다. 표4와 표5에 요약된 이러한 차이점들은 효율적인 사용자 인터페이스를 설계하고 개발하는 것이 얼마나 어려운 직업이 될 것인가를 간접적으로 시사한다.[29]

휴먼-컴퓨터 인터액션 분야에 종사하는 사람은 인간 행위에 대한 인지 이론과 모델에 관한 연구를 하는데, 여기서 다루는 인간 행위에 대한 연구는 전문가와 초보자와의 차이, 인지 형식이나 개성 및 성별과 같은 개인적인 차이, 시스템이나 모델을 평가하기 위한 실험적 방법론에 관한 것이다. 일반적으로 이러한 연구는 매우 어려워 많은 비용과 시간이 요

표 4. 인간과 컴퓨터의 정보처리 방법

구 분	컴퓨터	인 간
메모리	레지스터 램 디스크 테이프, 카드	단기 메모리 중급 메모리 장기 메모리 책, 도형
입력 디바이스	키보드, 터치 스크린 음성 인식기, 마우스	눈, 귀, 터치
출력 디바이스	화면, 프린터 음성 합성기	손, 음성, 눈
프로세스	운영 체제 컴파일러 응용 프로그램	수행 제어 유니트 도형 인식 지식과 기술

표 5. 인간과 컴퓨터의 정보 처리 방식에 대한 비교

구분	인 간	컴 퓨 터
장점	강력한 도형 인식 강력한 선택력 학습 능력 무한 용량의 장기 메모리 풍부한 다중접근 장기 메모리	대용량 메모리 영구 메모리 매우 빠른 처리 오류가 없는 처리 능력 신뢰성 있는 메모리 접근
단점	저용량 작업 메모리 급속히 붕괴되는 작업 메모리 저속 처리 오류를 저지르기 쉬운 처리 신뢰성이 결여된 장기 메모리	단순한 템프레이트 정합 제한된 학습 능력 제한된 용량의 장기 메모리 제한된 데이타 인테그레이션 능력

구되지만 이와 같은 연구 없이는 다양한 부류의 사용자가 컴퓨터가 제공하는 강력한 환경을 쉽게 배워서 쉽게 사용할 수 있는 사용자 인터페이스를 개발하기를 바랄 수 없다.

이 절에서는 지금까지 휴먼-컴퓨터 인터액션 분야에서 진행되어 온 여러 연구 결과물들을 참고로 하여 휴먼 팩터가 사용자 인터페이스를 설계할 때 미치는 영향을 이론적으로 검토하고 사용자 지식을 표현하기 위한 모델과 대화 방식을 간략히 소개한다.

1. 휴먼-컴퓨터 인터액션과 사용자 인터페이스

사용자 인터페이스는 인간과 컴퓨터 사이에 용이한 대화 방법을 제공하는 시스템이어야 한다. 용이한 대화 방법을 제공하기 위해서는 시스템을 설계할 때 사용자가 원하는 요구사항을 면밀히 조사하고, 신중한 계획을 세워 개발하여야 하며, 설계된 인터페이스에 대한 반복적 시험을 통하여 신속히 성능을 평가하고 평가된 결과에 따라 시스템을 개량할 수 있어야 한다. 단순히 사용자 편이를 제공한다는 추상적인 목적만 가지고는 일반 사용자에게 광범위한 호응을 얻을 수 없으며, 어떠한 종류의 사용자들이 무슨 목적으로 사용자 인터페이스를 요구하는 가를 올바로 인식하여 이를 설계시 반영하여야 한다. 이상적인 사용자 인터페이스는 사용자가 컴퓨터의 방해를 전혀 받지 않고 인터페이스 조차도 있다는 사실을 느끼지 못하면서 자신이 추구하는 일에만 전념할 수 있게 해 주는 것이다.

사용자 인터페이스는 컴퓨터를 사용할 사람의 특성 즉 개인의 능력과 경력, 개성 그리고 일하는 스타일에 따라 설계되어야 하므로 인터페이스 설계자는 설계시 고려하여야 할 휴먼 팩터 요소에는 무엇이 있는지 알아야 한다. 이에는 사용자 사이의 신체적 차이에서 발생하는 문제와 인지 능력 및 지각 능력의 차이에서 유발되는 문제, 개성의 차이에서 생기는 문제 등이 있다. 신체적 차이라 함은 인체 특성에 따른 행동 반경, 손가락 놀림 속도, 사물을 지각하는 능력의 차이를 말하는데 특히 사물에 대한 지각과 연관된 분야는 시스템 설계에 많은 영향을 미치므로 인터페이스 설계자들은 인간의 지각 능력의 범주를·인식하여야 한다. 지각 능력 중에서 인터페이스 설계시 가장 민감한 것은 시각인데 이와 관련된 연구 분야에는 시각 자극에 대한 반응 시간에 대한 연구와 화면의 콘트라스트 감도나 프리커링에 대한 연구이다. 시각 이외에도 키보드나 터치 스크린을 만지는 것에 연관된 촉각, 소리나 음성을 듣는 청각 기능도 인터페이스 설계시 고려하여야 할 신체적 특성에 해당한다. 작업 환경도 사용자의 신체적 차이에 따라 꾸며져야 될 것이다. 감지된 화면의 변화를 신속히 해석하고 그에 대한 대응책을 마련하여 컴퓨터에 명령어 스트림을 입력시킬 수 있는 인간의 정보 처리 장치는 다음과 같은 기능을 갖고 있기 때문에 사용자의 인지 능력과 지각 능력은 필히 고려되어야 한다.[46]

○단기 메모리 기능
○학습 기능(장기 메모리 기능)
○문제 해결 기능
○판단을 내리는 기능
○일에 대한 주의력
○탐색 및 스캔닝 기능
○시간 지각 기능

개성의 차에 따라 대화 방식의 속도, 표현 방법,

데이타 표현 등이 다르기 때문에 사용자 인터페이스를 설계할 때 개성의 차는 중요한 요소이다. 특히 특정 부류의 사용자를 위한 시스템을 설계할 때 개성의 차에 대한 이해가 더욱 절실히 요구된다. 사용자의 개성을 분류하는 명확한 방법은 없지만 최근 관심을 끌고 있는 모델로는 Carl Jung의 이론에 바탕을 둔 "Myers-Briggs 타입 지시기"(MBTI)가 있다.[46] MBTI는 직업과 개성사이의 관계, 서로 다른 성격 소유자들 사이에서의 관계에 대한 이론을 제공하여 사용자 부류를 시험하고 설계하는 데에 하나의 지침서를 제공한다.

위와 같은 사용자의 특성을 고려하여 설계된 인터페이스는 주어진 예산과 시한 내에 설계되어야 하며, 신뢰성, 보안성, 통합성을 제공하며, 적합한 기능을 갖추어야 한다. 적합한 기능을 갖추기 위해서는 요구되는 업무를 수행하기 위하여 필요한 기능이 무엇인지 정확하게 규정하여야 한다. 빈번히 행하여 질 업무는 결정하기 쉽지만 가끔 일어나는 일이나 예외적인 업무 혹은 시스템을 사용할 때 발생한 오류를 처리하기 위한 보수 업무 등은 규정 짓기가 매우 어렵다. 기능이 적절하게 규정되지 못하면 인간과의 인터페이스가 아무리 잘 설계되었더라도 시스템 자체가 비효율적으로 되기 쉽다. 반면에 지나치게 많은 기능을 제공하면 시스템을 구현하고, 유지 보수하고, 사용자를 학습시키고, 사용하는 데에 더욱 어려워진다. 사용자가 이용할 수도 있는 타 기종과의 호환성도 시스템 설계시 고려할 중요한 요소이다. 따라서 인터페이스 설계자는 시스템의 성능 자체를 향상시키는 데에만 관심을 두지말고 성능 향상이 사용자 편이성에 주는 영향도 세밀히 검토하여야 한다.

개발된 인터페이스 성능을 평가하기 위하여 일반적으로 측정하는 휴먼 팩터로는 다음과 같은 것이 있다.[46]

- ○학습시간 – 특정 부류의 사용자가 업무 수행과 연관된 명령어 사용법을 배우는데 소용된 시간을 측정한다.
- ○수행 속도 – 시스템이 벤취마크 시험 세트를 수행하는데 걸리는 시간을 측정한다.
- ○사용자 오류율 – 벤치마크 시험 세트를 수행할 때 발생한 오류의 횟수와 종류를 기록한다.
- ○주관적 만족도 – 사용자가 시스템을 좋아하는 정도로서 다양한 부류의 사람들과 인터뷰를 하여 자료를 수집한 후 통계 처리를 하여 측정된다.
- ○시간상의 보류(retention over time) – 시간이 경과한 후 사용자가 인터페이스에 연관된 지식을 얼마나 잘 유지하고 있는가를 측정한다. 시간상의 보류는 학습시간과 깊은 관계를 가지고 있고 사용 횟수가 이를 결정하는 요인이 된다.

2. 사용자 지식에 대한 모델

인간과 컴퓨터 사이의 의사 전달을 위한 대화 방식을 설계하는 사람은 사용자의 지식을 표현하기 위한 이론과 모델을 기반으로 설계과정을 구성한다. 사용자 지식을 표현하기 위한 모델로 신택틱/시맨틱 모델이 있다. 이 모델에서는 디바이스에 종속적인 상세한 구문지식(syntactic knowledge)과 개념에 대한 의미 지식(semantic knowledge)을 사용자가 갖고 있다고 전제한다. 의미 지식은 그림4과 같이 업무개념(task concept)과 컴퓨터 개념(computer concept)으로 나뉘어지고 각 개념은 객체와 행위로 세분된다. 사용자 행동의 신택틱/시맨틱 모델은 프로그래밍을 묘사하기 위해 창안되었고[45,46] 직접조작과[48] 더불어 데이타베이스 조작에[47] 적용되었다.

대화 방식을 모델링하기 위한 다른 방식으로 "천이 다이아그램"(state transition diagram)이 있다.[25] 천이 다이아그램을 사용하면 명령어를 설계하고, 학습시간 및 성능을 예측하고 오류 발생을 예측할 수 있다.

Card, Moran과 Newell은 "GOMS"(goals, operators, methods, and selection rules) 모델과 "Keystroke-level"모델을 제안했다.[13,14] GOMS 모델에서 사용자는 목적과 그 목적을 수행하기 위한 방법이나 절차에 의해 수행되는 부목적을 규정한다고 주장했다. GOMS 모델에서 연산자는 사용자의 지적 상태를 변화시키거나 업무 환경에 영향을 주는데 필요로 하는 기본적 지각, 모터 또는 인식 활동이며, 선택 규칙은 목적을 수행하기 위하여 유용한 방법을 선택하기 위한 제어 구조이다. Keystroke-level 모델에서는 사용자가 시스템에 반응하기 위해 키보드를 치고, 지시하고, 그림을 그리고, 생각하고, 기다리는 시간을 합하여 성능을 예측하는데 업무 수행시 오류를 발생시키지 않는 전문 사용자를 대상으로 하였다.

Kieras와 Polson은 대화 방식 텍스트 편집기에서 조건과 행위를 묘사하기 위해 프러덕션 룰을 사용했다.[25] 프로덕션 룰의 갯수와 복잡도를 통하여 삽입, 삭제, 복사, 이동, 교체와 같은 텍스트 편집 동작 및 학습 시간과 성능을 정확히 측정할 수 있다.

Norman은 다음과 같이 4단계로 구성된 대화 방식

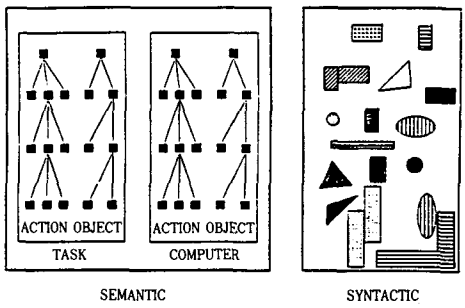

그림 4. 사용자 지식 표현을 위한 모델[49]

모델을 제시하였는데[35], 사용자는 형식화된 개념적 의도를 여러 명령어의 의미로 재형식화 한 후 궁극적으로 스크린에 위치를 선택하기 위해 마우스를 움직이는 행위를 한다고 한점에서 Foley 모델[18]과 유사하다.

○의도 형식화 : 목적에 대한 내부의 지적 특성을 나타낸다.
○행위 선택 : 가능한 행위를 검토하고, 가장 적절한 것을 선택한다.
○행위 수행 : 컴퓨터를 사용하여 행위를 수행한다.
○결과 평가 : 행위 수행의 결과를 검사한다.

3. 대화 방식

업무 분석이 끝나고 업무 객체와 행위의 의미가 확인되면 설계자는 대화 방식을 선택하게 되는데 많이 이용되는 기본적인 대화 방식에는 메뉴 선택, 형식 채우기(form fill-in), 명령 언어, 자연어, 직접 조작이 있다.

1) 메뉴 선택

사용자는 메뉴 항목을 읽고, 업무에 가장 적절한 하나의 항목을 선택하여 행위를 초기화하고 결과를 확인한다. 만일 메뉴 항목에서 사용한 용어와 의미를 명백히 이해할 수 있으면 사용자는 쉽게 배울 수 있으며 많은 입력 스트림을 주지 않더라도 업무를 수행할 수 있다. 가장 큰 장점은 한 시점에서 소수의 선택만이 주어짐으로 처리방법이 매우 간단하다. 그러므로 초심자나 중간급 사용자에게 적절하고 만일 디스플레이와 선택 메카니즘이 매우 빠르면 전문 사용자에게도 유용하다. 메뉴 선택 시스템 설계자는 시스템의 효율성을 증가 시키기 위하여 신중하게 업무를 분석해야 하고 의미에 맞도록 용어를 잘 선택하여 일치성있게 설계하여야 한다.

2) 형식 채우기

형식 채우기는 자료 입력이 요구되는 응용 분야에 적합하다. 사용자는 관련 부분의 디스플레이를 보고 필드 사이에서 커서를 움직여 원하는 곳에 자료를 입력한다. 이 대화 방식을 사용하는 사용자는 필드 레이블을 이해해야 하고 가능한 값과 자료 입력방법을 알고 능히 오류 메시지에 반응할 수 있어야 함으로 사전 지식이 요구되어 약간의 훈련이 필요하다. 따라서 중간급 사용자나 전문 사용자에게 적합하다.

3) 명령 언어

전문 사용자는 명령어의 제어 위치를 명백히 이해하고 있으므로 시스템에서 제공하는 애매모호한 프롬프트를 읽지 않고도 이미 배운 명령 언어를 사용하여 다양한 표현을 할 수 있다. 그러나 구문 오류를 범하기 쉽고, 배우고 기억하기 어렵다. 명령 언어는 업무와 컴퓨터 사이의 맵핑이 어렵고 복잡하기 때문에 온라인으로 사용하기 힘들 뿐만 아니라, 오류 메시지도 처리하기 어렵다.

4) 자연어

컴퓨터가 임의의 자연어 문장이나 구문에 적절하게 반응할 수 있는 시스템이 되도록 하는 많은 연구가 있어 왔지만 자연어 처리에 존재하는 모호성의 문제 때문에 아직 활용화하기 힘든 상태에 있다. 현재로는 제한된 범위의 업무 영역에서 지식이 풍부하고 명령 언어 훈련이 잘된 중간급 사용자를 위한 대화 수단으로 사용되고 있다.

5) 직접 조작

대화 방식 설계자가 행위 절차를 시각적으로 잘 표현할 수 있도록 설계하면 사용자는 자기가 하고 싶은 업무에 대하여 객체를 직접 조작하게 함으로써 행위를 단순화 시킬 수 있다. 디스플레이 편집기, LOTUS 1-2-3, 항공 교통 관제 시스템, 그리고 비디오 게임 등은 직접 조작의 좋은 예이다. 사용자가 원하는 업무를 객체와 행위의 시각적 표현을 이용하여 다루면 사용자는 업무를 신속히 수행하게 되어 그 결과를 즉시 관찰하게 된다. 직접 조작 대화 방식에서는 명령 언어 방식이나 메뉴선택 방식에서 처리했던 입력들을 시각적 객체와 행위를 표시하는 아이콘을 이용하여 나타낸다. 직접 조작은 초심자도 쉽게 사용할 수 있고 중간급 사용자에게도 쉽게 기억이 되며, 설계의 정도에 따라 전문 사용자에게도 유용하게 쓰일 수 있다.

그러나 어느 하나의 대화 방식만으로는 사용자가

의도하는 업무를 수행하기 어려우며 여러 대화 방식의 장단점은 표6에 요약하였다.

표 6. 대화 방식의 장단점

대화 방식	장 점	단 점
메뉴 선택	학습 기간이 짧다. 약간의 키만을 친다. 의사결정 구조를 갖는다. 대화 관리 도구를 사용한다. 오류 처리를 보조하기 쉽다.	메뉴가 많아질 위험이 있다. 전문 사용자에게는 느리다. 스크린 공간을 낭비한다. 빠른 디스플레이 율을 요구한다.
형식 채우기	데이타 입력을 단순화 한다. 적당한 훈련을 요구한다. 보조가 편리하다. 형식 관리 도구를 사용한다.	스크린 공간을 소비한다.
명령 언어	융통성이 있다. 전문 사용자에게 호감을 산다. 사용자의 독창성을 보조한다. 사용자 정의 마크로를 작성하기가 편리하다.	오류 처리가 빈약하다. 구체적인 훈련과 기억을 요구한다.
자연어	구문 학습의 어려움을 덜어준다.	명백한 대화를 요구한다. 보다 많은 키를 쳐야한다. 문맥을 볼 수 없다. 예측할 수 없다.
직접조작	업무 개념을 시각적으로 표현한다. 배우기 쉽다. 기억하기 쉽다. 오류가 적다. 탐구를 촉진한다. 높은 주관적 만족을 준다.	프로그램 하기가 어렵다. 그래픽 디스플레이와 부가적 장치를 요구한다.

V. 신경회로망

인간은 어린아이일찌라도 현존하는 슈퍼 컴퓨터의 성능으로도 모방하기 어려운 지능적인 작업들을 쉽게 수행해 낸다. 문자를 인식하는 경우만을 보더라도 컴퓨터가 인쇄체 한글문자를 인식하는 데에 있어서 95% 이상을 운운하게 된것이 최근의 일이며 아직 다양한 서체의 인쇄 문자나 필기 문자를 인식하기에는 요원한 실정인 반면 인간에게는 아주 사소한 문제에 지나지 않는다. 이러한 성능의 차이는 컴퓨터의 구조 자체가 인간의 신경 구조와 판이하게 다르다는 사실에 기인한다. 신경회로망은 신경계의 특이한 구조와 기능을 보다 과학적으로 연구하고 그 결과를 컴퓨터에 응용해 보고자 만들어낸 수학적 모델이다.

1. 신경회로망과 폰 노이만 컴퓨터

신경회로망은 인간 두뇌의 정보처리 구조 및 과정을 반영하고 있을 뿐만 아니라 그 특성도 잘 반영하고 있다. 가장 큰 특징은 (1) 병렬분산처리, (2) 학습, (3) fault tolerant 등인데, 이들을 포함한 자세한 특성 몇가지를 폰 노이만형 컴퓨터와 비교하여 설명하면 표 7과 같다.[31]

표 7. 신경회로망과 폰 노이만 컴퓨터의 비교[31]

Von Neumann	신경회로망
○모든 정보처리가 중앙처리장치(cpu)의 제어하에 이루어진다.	○정보의 처리가 각 뉴런에 의해 병렬분산 처리된다.
○cpu의 고장이 전체 시스템의 고장을 야기한다.	○병렬분산처리를 하므로 임의 뉴런의 고장에 시스템의 성능은 저하될지라도 시스템 전체의 고장을 야기하진 않는다.
○digital data 처리를 기본으로 하고 있다.	○analog signal 처리를 기본으로 한다.
○수학적, 논리적 함수에 근거한 yes/no 의사 결정을 한다.	○모호한 정보, 불완전한 정보, 확률적 정보에 대해서도 가중치 의사 결정을 한다.
○미리 결정된 순서에 준하여 데이타를 처리하므로 항상 control이 필요하며, 계산 결과를 예측할 수 있다.	○데이타 처리방법을 독자적으로 학습할 수 있으며 흔히 계산 결과를 예측하기 어렵다.
○충분한 시간이 주어지면 주어진 문제의 정확한 답을 구한다.	○복잡한 문제에 대한 근사해를 신속히 찾아낸다.
○exact match에 의해 자료를 찾는다.	○best match에 의해 자료를 찾는다.
○원하는 데이타만 쉽게 찾아낼 수 있도록 함께 찾을 수 있도록 기억한다.	○단편적인 정보를 통하여 이에 연상되는 정보를 기억한다.

2. 신경회로망의 기본 모델

동물신경계의 기본 단위는 뉴우런이며 뉴우런은 중심의 체세포(soma 또는 cell body)와 이로부터 뻗어나온 수상돌기(dendrite)와 축색돌기(axon)라는 신경 섬유로 구성되어 있다. 뉴우런은 신호를 한 방향으로만 전달하며 수상돌기가 인접 뉴우런으로 부터 신호를 받아들이고 축색돌기가 신호를 내보낸다. 축색돌기의 가지는 시냅스를 매체로 하여 다른 뉴우런의 수상돌기에 접속되어 뉴우런의 출력을 다른 뉴우런의 입력으로 전송하여 준다.

그림5는 일반적으로 소개되고 있는 뉴우런의 모델로서 시냅스가 연결강도(connection weight)이고 다

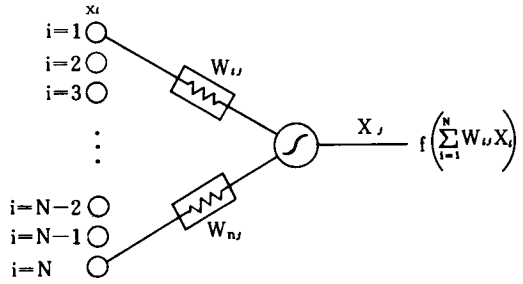

그림 5. 유니트[15]

른 뉴우런의 출력을 연결강도의 비율로 전달받아 이를 모아서 자기의 출력을 결정하는 간단한 처리기(processing element)로 보고 있다. 이후부터 이러한 뉴우런에 해당하는 처리기를 유니트(unit)라 칭한다. 유니트의 기능을 식으로 표현하면 식(1)과 같다.

$$x_j = f(net_j - \theta_j) \qquad (1)$$

$$net_j = \sum_{i=1}^{M} x_i w_{ji}$$

식(1)의 θ_j는 유니트 j의 bias이며, 함수 f는 전달함수(transfer function)로서 이 함수의 특성상 크게 결정적 함수(deterministic function)와 확률적 함수(stochastic function)로 나뉘어진다. 결정적 함수의 경우는 다시 선형(linear), 비선형(nonlinear) 으로 나뉘어지는데 비선형에는 임계함수(threshold function), sigmoid 함수 등이 있다.

유니트 단독적으로는 의미있는 기능을 수행하지는 못한다. 식(1)에서 보는 바와 같이 다른 유니트의 상태가 자기 유니트의 출력을 결정하는 주요 요인이 된다. 다시 말하면 유니트의 기능은 단독적이기 보다 다른 유니트 간의 연결과 이들 간의 정보교환에 의한 총체적 기능의 한 부분일 뿐이다. 이에 따라 유니트를 이용한 임의의 기능 체계를 구성하려면 반드시 유니트를 연결지어 망을 구성하여야 하는 것이다. 이러한 망을 신경회로망(neural network)이라 한다.

3. 다층 퍼셉트론(multi-layer perceptron)

다층 퍼셉트론[41]은 최근에와서 가장 많이 응용되고 있는 신경회로망 모델 중의 하나이다. 본 고에서는 다층 퍼셉트론의 구조와 학습을 보다 자세히 설명함으로써 신경회로망 실체에 대한 이해를 돕고자 한다. 다층 퍼셉트론이란 입력 유니트층과 출력 유니트층 사이에 둘 이상의 연결층이 존재하는 신경회로망으로 sigmoid 함수를 전달함수로 사용한다. 그림6에 두개의 층을 형성하고 있는 다층 퍼셉트론의 예가 나타나 있다.

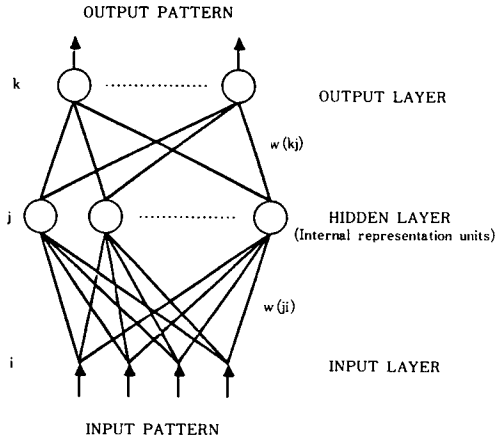

그림 6. 이층 퍼셉트론

다층 퍼셉트론의 학습은 EBP(error back-propagation)[41] 알고리즘에 의하여 수행된다. Error back-propagation 알고리즘은 다층 perceptron의 학습 알고리즘인 least mean square 알고리즘을 일반화한 것으로 기억시켜야 할 모든 입출력 쌍에 대하여 실출력(actual output)과 출력되어야 할 올바른 값(teaching value) 간의 mean square difference를 최소화 시키도록 하는 연결강도를 점진적으로 탐색해가는 알고리즘이다. 이러한 점진적 탐색은 망입력을 제공하고 이에 대응하는 실출력의 오차를 하위 층에 전달(back propagation)하면서 일률적 규칙에 따라 연결도를 조정해 나가는 과정을 반복적으로 적용함으로써 이루어진다. 이 반복과정은 다음과 같이 5단계로 이루어진다.[27]

(1) Initialize weights and biases

연결도와 bias를 작은 random number로 초기화 한다.

(2) Present input and desired output

입력과 이에 대응하는 출력 쌍 하나를 입출력 층에 제시한다. 이때 입력을 $X = <x_0, x_1, \cdots, x_{N-1}>$, 대응하는 출력을 $D = <d_0, d_1, \cdots, d_{M-1}>$ 이라 한다.

(3) Calculate actual output

각 유니트 u_j 마다 입력의 가중치 합 net_j를 식(1)에 따라 구하고 sigmoid nonlinear 함수에 적용하여 출력을 발생한다.

$$o_j = f(net_j) = \frac{1}{1+e^{-(net_j-\theta_j)}} \quad (2)$$

식(2)에서 θ_j는 유니트 u_j의 고유 bias이며, 이때 출력층을 통한 망출력을 $Y=<y_0, y_1, \cdots y_{M-1}>$이라 한다. 즉, u_j가 출력 유니트라면 $y_j=o_j$이다.

(4) Adapt weights

상위층의 유니트 u_i와 하위층의 유니트 u_j 간의 연결도 w_{ij}를 식(3)과 식(4)에 따라 조정한다.

$$w_{ij}(t+1) = w_{ij}(t) + \triangle w_{ij}(t) \quad (3)$$

$$\triangle w_{ij}(t) = \eta \delta_i o_j \quad (4)$$

식(4)는 연결도의 변화값을 나타내는데 o_j는 임의 유니트 u_j의 출력이거나, u_j가 입력 유니트인 경우 망입력 x_j이며, η는 오차의 반영율 내지는 학습율(learning rate, $0<\eta<1$)이고, t는 w_{ij}의 조정이 반복된 횟수이다. 또한, δ_i는 u_i의 오차로서 u_i가 출력 유니트인 경우

$$\delta_i = y_i(1-y_i)(d_i-y_j),$$

출력 유니트가 아닌 내부 유니트인 경우는

$$\delta_i = o_i(1-o_i)\sum_k \delta_k w_{jk}$$

이다. 단, k는 u_i의 차상위층 모든 u_k의 인덱스이다.

(5) Repeat by going to (2)

모든 입출력 벡터쌍 p=(X, D)에 대한 실제 출력 벡터 Y의 오차 E가 충분히 작지 않은 한 (2)로 다시 돌아간다.

$$E = \sum_p E_p = 1/2 \sum_p \sum_j (d_{pj}-y_{pj})^2 \quad (5)$$

EBP의 학습 속도, 즉 반복 횟수는 주로 학습율 η에 좌우되는데 실험 결과 결과 이 값이 적당히 클 때 속도가 빨라지는 것으로 밝혀졌다.[41] 또한 속도를 조정하기 위해서는 식(4)를 다음과 같이 확장시킬 수 있다.

$$\triangle w_{ij}(t) = \eta \delta_i o_j + \alpha \triangle w_{ij}(t-1) \quad (6)$$

식(6)에서 α는 momentum term 즉, 현재의 탐색방향에 과거 연결도 변화값의 반영율로서 이 α도 η와 마찬가지로 적당히 클 때 속도를 빠르게 하는 것으로 실험적으로 밝혀졌다. 다층 퍼셉트론은 구현이 쉽고 학습이 가능한 이유로 해서 음성 생성[44,56], Road Tracker[39], 음성인식[53,54] 등 현재 가장 많은 분야에서 성공적으로 응용되고 있는 신경회로망 모델이다.

4. 연구 동향

신경회로망은 현재 다양한 기술 및 이론 분야에 응용되고 있다. 이들을 요약하면 다음과 같다.

(1) Pattern classification : 미리 주어진 정보와 자료에 따라 미지의 입력을 분류하는 기술

(2) Self organization / category formation : 미리 정해진 정보나 규칙이 없이도 주어진 많은 양의 자료를 그 유사성에 따라 스스로 분류하고 조작하는 기술

(3) 연상 기억(associative memory) : 일부 또는 손상된 정보로부터 온전한 정보를 되살리는 기술

(4) Sensory data processing : 시각, 청각 정보의 처리 기술

(5) Nonlinear mapping : 학습을 통하여 robot control 등 비선형 mapping을 수행하는 기술

(6) Computational problem : 분산형 연산을 통하여 최적화 문제를 효과적으로 해결하는 기술

(7) Multi-sensor automata : 상호 작용하는 여러개의 모듈을 스스로 제어하는 기술

신경회로망의 연구에는 응용 뿐만아니라 구현기술에도 많은 관심이 모아지고 있다. 현재 진행중인 구현 기술에는 다음 사항들이 있다. 이에 대한 자세한 사항은[15,31]을 참조하기 바란다.

(1) Software simulation

(2) Hardware simulation

(3) VLSI implementation

(4) Optical implementation

(5) Optoelectronical implementation

(6) Molecular/chemical implementation

현재 각국에서는 정부 주도로 대형 프로젝트가 시작되고 있고, 대학 및 연구소, 기업체 등에서 이론 및 응용기술의 개발과 상업화 경쟁에 나서고 있어 조만간에 주목할만한 시장을 형성할 것으로 점쳐진다.

(1) 미국

DARPA (USA Defense Advanced Research Project Agency)가 8년간에 걸쳐 연구비 3억 9천만달러에 달하는 대규모 과제를 시작하였으며 학교 및 연구소에서 연구가 가장 활발한 국가이다. 이미 신경회로망의 시뮬레이터가 상품으로 개발되어 판매되고 있다. 뿐

만아니라 학회활동도 활성화되어 있어 INNS (International Neural Network Society)라는 학회 주도로 IEEE의 지원을 받아 대규모 국제학술회의 IJCNN (International Joint Conference on Neural Network)을 개최하고 있다.

(2) 유럽

서독 정부가 1988년 1월부터 10년간 1억달러가 넘는 "Information Processing in Neural Architecture" 과제를 시작하였고, 프랑스에서는 파리에서만도 6개의 과제에서 neural computer용 micro chip을 개발하고 있으며, ESPRIT Ⅱ에서 이미 PIGMALION과 ANNIE라는 두개의 과제를 수행하고 있다. 학술활동에 있어서도 활발한 편으로 프랑스는 1988년에 nEuro '88이라는 국제학술회의를 개최한 바 있고 1990년에도 국제학술회의 INNC(International Neural Network Conference)를 개최할 예정에 있다.

(3) 일본

전자부분의 선두 기업들이 관심을 보이고 있다. Fujitsu는 방문객들에게 패턴인식과 로보틱스를 포함한 신경회로망의 데모를 보여주고 있다. Mitsubishi는 최근 광학 기술로 연상 기억 신경회로 컴퓨터의 prototype을 개발하였다고 발표하였다. NEC는 최근 광학 문자 인식의 성능을 향상시키기 위하여 신경회로망을 사용하여 인쇄된 문서를 컴퓨터에 입력하기 위한 입력 장치를 개발하였다.

(4) 국내

국내에서도 연구가 활발히 진행되고 있다. 특히 한국전자통신연구소에서는 한글 단어를 발음기호로 변환하는 신경회로망을 개발하였고[5,6] 숫자인식[1,4] 신경회로망을 개발하였으며 한글 필기체 인식을 위한 신경회로망 개발에 착수하였을 뿐만아니라 신경회로망의 VLSI 구현이나 광학적 구현에까지 관심을 쏟고 있다. 한국과학기술원에서는 한글 인쇄체 문자를 인식하는 시스템 망눈[3]을 개발하였다. 이외에도 연세대학교에서는 한글 필기체 인식을 위한 신경회로망 모델[5]의 연구가 성과를 거두고 있고, 경북대학교에서는 신경회로망의 VLSI 구현[6]에 관한 연구가 진행 중에 있으며, 포항공과대학에서는 컴퓨터비젼[7]과 로보틱스[2] 분야의 응용 연구가 진행중에 있다. 최근에는 한국전자통신연구소와 한국정보과학회가 공동으로 '89 신경회로망 응용 워크샵을 개최한 바 있다.

Ⅵ. 지능형 컴퓨터

1. 개요

지능형 컴퓨터 개발은 2,000년대의 세계 컴퓨터 시장의 변화에 따른 우리의 적극적인 대응이라 할 수 있다. 2000년대에는 지식정보처리 업무 분야가 새로이 생겨날 것이고 이를 잘 처리할 수 있는 컴퓨터인 지능형 PC, 지능형 워크스테이션, 지능형 컴퓨터가 등장할 것이다. 이 계획은 앞으로 피할 수 없이 다가올 정보화 사회에서 가장 중요한 역할을 담당할 컴퓨터의 기술자립의 선진화없이는 선진국과 어깨를 나란히 할 수 없다는 인식에서 출발하였다. 또한 국내외 통신망 환경의 변화에 따른 컴퓨터 수요가 급증함에 따라 소량 다품종 통신망 서비스의 신속한 개발 및 유지 보수를 위하여 컴퓨터 처리 능력의 보강이 필요하고, 1994년 국내 ISDN 사용 서비스 개시에 따른 멀티미디어 전송 능력 및 멀티미디어 처리 능력을 갖춘 지능형 컴퓨터의 수요가 증가할 것이다.

현재 추진중인 지능형 컴퓨터 개발 계획을 보면, 총 7년간의 연구기간이 필요하며 1단계, 2단계로 나누어 수행한다. 제 1 단계는 1990년부터 1993년 까지 수행되어 Mega LIPS급 지능형 컴퓨터가 개발될 것이며, 제2단계는 1단계가 진행중인 1992년 부터, 제1단계에서 축적된 설계 기술을 바탕으로 5년간 수행되어 Giga LIPS급 지능형 컴퓨터를 개발하게 될 것이다. 총 연구비는 900억원으로 제1단계에서는 440억원, 제2단계에서는 460억원이 소요될 것이고 1,600여명의 우수한 연구 인력이 투입될 예정이다.

우리나라의 컴퓨터 개발 기술은, 1982년에 시작한 교육용 컴퓨터 개발 과제로 복제기술을 확보하여 현재 세계 PC 시장 3위의 기반 기술을 유도했으며, 1987년 부터 추진되고 있는 행정전산망 주전산기 개발 과제로 하드웨어 설계 기술을 확보하였다. 전체 컴퓨터 개발 기술 수준에서 볼 때 우리나라 기술 수준이 가장 취약한 곳이 프로세서 설계 개발 기술과 시스템 소프트웨어 설계 기술이다. 지능형 컴퓨터개발 계획에서는 이 두 가지 기술을 확보하여 고유 프로세서 설계 기술등 원천 기술 확보로 기본 소프트웨어 개발을 유도하여 우리의 컴퓨터 개발 능력을 선진국과 동일 수준으로 끌어올리는 결과를 낳게 할 것이다.

2. 주요 기능

우선 사용자의 입장에서 볼 때 지능형 컴퓨터의 주요 기능은 다음과 같은 세 가지 특징이 있다고 할 수 있다. 첫째가 우리말과 글로 컴퓨터를 사용하는

것이다. 지금까지의 컴퓨터는 외국의 문화에 맞게 설계되고 개발되었기 때문에 영어로 모든 명령어를 사용하게 되어 있어 전문가가 아니면 컴퓨터를 배우기도 또한 사용하기도 힘들었다. 그러나 지능형 컴퓨터에서는 우리가 일상생활에서 사용하고 있는 우리 말과 글로 컴퓨터와 대화함으로써 보다 컴퓨터의 사용이 쉬워지게 된다. 즉 우리의 음성을 알아듣고, 우리가 쓴 한글을 인식할 수 있어 전문가가 아닌 일반 사용자도 쉽게 컴퓨터를 사용할 수 있을 것이다.

둘째 멀티 미디어의 처리이다. 앞으로 다루어야 하는 정보가 다양해지고 복잡해짐에 따라 컴퓨터 사용자들이 멀티미디어의 처리를 할 수 있는 컴퓨터를 요구하게 될 것이다. 여기서의 멀티미디어란 여러 종류의 미디어를 뜻하며 그 종류로는 음성, 영상, 텍스트들로서 이때까지의 컴퓨터가 제대로 처리할 수 없는 것들까지 다룰 수 있게 된다. 지능형 컴퓨터에서는 이들을 저장, 관리하는 일 뿐만 아니라 편집까지도 할 수 있게 된다.

세째 통합 지식 정보의 처리이다. 이때까지의 컴퓨터는 단순한 숫자 계산을 주로 해왔으나 지능형 컴퓨터에서는 멀티미디어에 의해 표현되는 지식정보들을 처리할 수 있는 기능을 가지게 된다. 이러한 일들은 추론 및 학습등에 의해 이루어지며, 지식의 저장 및 관리 뿐만 아니라 수집 검증까지 할 수 있게 될 것이다.

위에서 열거한 특징을 가진 지능형 컴퓨터는 멀티미디어를 처리할 수 있는 입출력 장치를 가지게 될 것이며, 멀티미디어로 표현된 지식 정보를 처리할 수 있으며, ISDN과의 접속으로 지식정보에 관한 서비스를 제공해 줄 것이다. 광대역 ISDN이 실현되면 멀티미디어 정보 전송이 가능하게 되기 때문에 미래 통신망이 추구하는 고도 서비스를 제공할 수 있을 것이다. 이를 위해서는 기존의 소프트웨어를 수행시킬 수 있어야 하며, 멀티미디어의 처리를 위해서 대용량의 storage가 필요하며,[1] 다양한 종류의 workstation 및 PC와도 접속 가능하게 해야 할 것이다.

이러한 지능형 컴퓨터의 응용 분야로는 터미날앞에서 도서관을 가서 할 수 있는 모든 일을 할 수 있는 전자 도서관, 외국인과도 우리말로 자유로이 통화할 수 있는 자동통역, 어려운 결정을 도와주는 경영자문, 외국어로 된 서적을 우리 글로 읽을 수 있는 기계 번역등 지식 정보를 잘 처리해 줄 수 있는 분야가 될 수 있을 것이다.

3. 하드웨어 구조

지능형 컴퓨터의 하드웨어 구조는 우선 위에서 설명한 특징들을 잘 지원해줄 수 있는 구조를 갖추어야 할 것이다. 이런 일을 가능하게 하기 위해서는 컴퓨터의 기본 구조가 이때까지와의 컴퓨터와는 근본적으로 달라야 하는데, 종래의 하나의 프로세서를 사용하는 컴퓨터와는 달리 1,000개 이상의 프로세서를 사용하게 될 것이고, 음성이나 영상 인식을 위하여는 새로운 신경망 구조를 갖는 입출력 시스템을 가질 것이다.

이러한 구조를 갖는 지능형 컴퓨터는 추론 속도가 10 Giga LIPS (logical inferences per second) 이상이며, 주 기억 장치는 10 Giga byte, 보조 기억 장치는 500 Giga byte 이상이 될 것이다. 또한 프로세서 자체도 멀티미디어의 통합된 정보를 처리하여야 하기 때문에 지금까지의 프로세서가 할 수 있는 능력을 넘어 새로운 구조의 프로세서를 개발하여야 할 것이다.

지능형 컴퓨터의 구조가 현재의 컴퓨터와 크게 다른 점은, 현재의 컴퓨터는 숫자처리를 위주로 하는 컴퓨터인 반면, 지능형 컴퓨터에서는 통합 정보를 잘 처리할 수 있는 구조를 가지게 될 것이다. 전체적인 시스템의 특징으로 보아 지능형 컴퓨터의 하드웨어 구조는 빠른 처리 속도만을 요구하는 것이 아니라 사용자에게 지식정보를 편리(smart)하게 처리해 줄 수 있게 디자인될 것이다.

4. 소프트웨어

우선 우리 말과 글을 잘 처리할 수 있는 자연어 인터페이스를 가지고 있어야 하며, 멀티미디어 정보를 저장하고 꺼낼 수 있는 멀티미디어 DBMS, 멀티미디어 정보를 편집할 수 있는 멀티미디어 편집기, 병렬처리 환경에 맞는 프로그래밍 언어등이 있어야 하며, 통합 정보의 추론을 지원할 수 있는 소프트웨어들이 개발되어야 할 것이다. 운영체제에서는 기존의 운영체제의 주요 기능뿐만 아니라 위에서 설명한 주요 응용 프로그램을 효율적으로 수행하기 위한 기능이 제공되어야 할 것이다.

지능형 운영체제와 기존의 운영체제 사이의 가장 큰 차이는 멀티미디어 정보 처리의 기능 및 지식 정보 처리기능을 제공하는 것이다. 이들 기능을 제공하기 위해서는 프로세서, 화일등에 대한 새로운 개념이 정립되어야 할 것이며, 멀티미디어 처리를 위한 기본 오퍼레이션, 지식 베이스 구성을 위한 기본 오퍼레이션, 각종 추론을 위한 기본 오퍼레이션등와

기능이 제공되어야 할 것이다. 또한 하드웨어 자원의 관리는 병렬하드웨어 및 멀티미디어 입출력 시스템을 잘 관리할 수 있어야 하며, 기존의 운영체제와는 상대적으로 용량이 큰 디바이스들을 관리하여야 할 것으로 생각된다.

그리고 지능형 컴퓨터에서 사용되는 프로그래밍 언어는 기존의 프로그래밍 언어들 외에 지능형 프로그램을 효과적으로 작성할 수 있는 기능이 제공되어야 할 것이다. 그러한 기능들로는 병렬 프로그래밍 기능, 멀티미디어의 간편한 처리기능, 지식 정보 처리를 효과적으로 할 수 있는 기능들을 가져야 할 것이다. 이 밖에도 특수 목적의 언어들로서 아동 교육용 프로그래밍 언어의 개발, 쉽게 프로그래밍할 수 있는 시각 언어의 개발등이 필요하다.

Ⅶ. 결 론

이제까지 인공지능 기술의 동향에 대해 간단히 살펴보았다. 원래 인공지능 분야 자체의 속성상 너무나 광범위한 분야를 포괄하고 있고 또한 지능(intelligence)이라는 개념이 잘 정의되어 있지 않기 때문에 실제 이상의 기대를 하여 설명하기도 했다. 그러나 사회가 점점 정보사회화 되어감에 따라 여러 의미의 지능이 잘 정의되고 구체화되어 그에 필요한 서비스(servicer)개발을 위하여 인공지능 기술이 핵심적인 역할을 할 것임이 틀림없다. 이미 인공지능 기술은 실험실을 벗어난지 오래요, 사용자들이 알게 모르게 정보산업 제품 곳곳에 스며들어 잘 활용되고 있다고 본다.

參 考 文 獻

[1] 김명원, 이영직, 이훈복, "신경회로망을 이용한 특징 추출," '89 신경회로망 응용 워크샵 논문집, pp. 39－49, 1988. 11. 17－18.

[2] 오세영, "신경회로를 이용한 로보트 팔의 제어," '89 신경회로망 워크샵 논문집, pp. 103－110, 1989. 11. 17－18.

[3] 이승호, 조성배, 김진형, "문자 인식에서의 기존의 방법과 신경망 방법의 비교," '89 신경회로망 응용 워크샵 논문집, pp. 69－79, 1989. 11. 17－18.

[4] 이영직, 곽종훈, "신경회로망을 이용한 Classification," '89 신경회로망 응용 워크샵 논문집, pp. 50－58, 1989. 11. 17－18.

[5] 이일병, "신경회로망 방법에 의한 필기체 한글 문자 인식", '89 신경회로망 응용 워크샵 논문집, pp. 80－86, 1989. 11. 17－18.

[6] 정호선, "신경회로망의 VLSI 구현,"전기학회지, vol. 38, no. 2, pp. 39－52, 1989.

[7] 정 홍, "Neural network algorithms for multiscale edge detection, '89 신경회로망 워크샵 논문집, pp. 103－110, 1989. 11. 17－18.

[8] J.M. Anderson, W.S. Coates, A.L. Davis, R.W. Hon et al., "The architecture of FAIM-1," *IEEE Computer,* vol. 20, no. 1, Jan. 1987.

[9] M.E. Atwood, "A report on the vail workshop on human factors in computer systems," *IEEE Computer Graphics and Applications,* vol. 4, pp. 48-66, 1984.

[10] Ketil Bo, Problems of the 80's in man/machine communication, Methodology of Interaction, R.A. Guedj and et al., Amsterdam: Elsevier, pp. 149-158, 1980.

[11] J. Bruner, *Toward a Theory of Instruction,* Harvard University Press, Cambridge, MA, 1966.

[12] J.M. Carroll, J.C. Thomas, and A. Malhotra, "Presentation and representation in design problem-solving," *J. of British Journal of Psychology,* vol. 71, pp. 143-153, 1980.

[13] S. Card et al., "The keystroke-level model for user performance with interactive systems," *J. of Comm. of the ACM,* no. 23, pp. 396-410, 1980.

[14] S. Card et al., *The Psychology of Human-Computer Interaction,* Lawrence Erlbaum Associates, Hillsade, NJ, 1983.

[15] Defense Advanced Research Projects Agency *DARPA Neural Network Study,* Virginia: AFCEA International Press, 1988.

[16] R.W. Ehrich, "DMS-A system for defining and managing human computer dialogues," *Automatica,* vol. 9, pp. 655-662, 1983.

[17] Mohammad U. Farooq and Wayne D. Dominick, "A survey of formal tools and models for developing user interfaces," *Int. J. Man-Machine Studies,* vol. 29, pp. 479-496, 1988.

[18] J. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics,* Addison-Wesley Pub. Co., Reading, MA, 1982.

[19] A. Goto, M. Sato, K. Nakajima et al., *Overview of the Parallel Inference Machine (PIM) Architectue* Proceedings of the International Conference on FGCS '88, ICOT, Tokyo, Japan, 1988.

[20] D. Hatfield, *Personal Communication and Lecture,* Conference on Easier and More Productive Use of Computer System, Ann Arbor, MI, 1981.

[21] P. Heckel, *The Elements of Friendly Software Design,* Warner Books, New York, NY, p. 205, 1984.

[22] K. Hwang, J. Ghosh, R. Chowkwunyun, "Computer architectures for AI processing," *IEEE Computer,* vol. 20, no. 1, Jan. 1987.

[23] K. Hwang, R. Chowkwanyun, J. Ghosh, *Parallel Architectures for Implementing Artificial Intelligence Systems, Parallel Processing for Super-Computers & Artificial Intelligence,* McGraw-Hill, 1989.

[24] E. Hutchins, J.D. Hollan and D.A. Norman, Direct Manipulation Interfaces, D.A. Norman, Draper Stephen W. (Editors), User Centered System Design: New Perspectives on Human-Computer Interaction, Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[25] D. Kieras and P. Polson, "An approach to the formal analysis of user complexity," *Intl. J. of Man-Machine Studies,* no. 22, pp. 365-394, 1985.

[26] T. Kurozumi, *Present Status and Plans for Research and Development,* Proceeding of the International Conference on FGCS'88, ICOT, Tokyo, Japan, 1988.

[27] R.P. Lippmann, An Introduction to Computing with Neural Nets, IEEE ASSP Magazin, Apr. 1987.

[28] A. MacDonald, "Visual programming," *Datamation,* vol. 26, no. 11, pp. 132-140, Oct. 1982.

[29] D.J. Mayhew, Basic Principles and Guidelines in User Interface Design, CHI'89 Conference Tutorial note, May 1989.

[30] John D. McGregor, Arthur M. Riehl, "The future of high performance computers in science and engineering," *Communications of ACM,* vol. 32, no. 9, Sep. 1989.

[31] R.K. Miller, *Neural Networks: Implementating Associative Memory Models in Neurocomputers,* SEAI Technical Pub., Madison, GA, 1987.

[32] M. Montessori, *The Montessori Method,* Schocken, New York, 1964.

[33] D.A. Moon, "Symoblics architecture," *IEEE Computer,* vol. 18, no. 6, Jan. 1987.

[34] T. Nelson, "Interactive systems and the design of virtuality," *J.of Creative Computing,* vol. 6, no. 11, Nov. 1980.

[35] D. Norman, "Stages and levels in human-machine interaction," *Intl. J. of Man-Machine Studies,* no. 21, pp. 365-375, 1984.

[36] R. Onai, H. Shimizu, K. Masuda, *Analysis of Sequential Prolog Programs,* ICOT, Tokyo, Japan, 1984.

[37] S. Papert, *Mindstroms: Children, Computers, and Powerful Ideas,* Basic Books, Inc., New York, NY, 1980.

[38] G. Polya, *How to Solve It,* Doubleday, New York, 1957.

[39] D.A. Pomerleau. *AL VINN: An Autonomous Land Vehicle in Neural Network, Advances in Neural Information Processing Systems.* 1. D.S. Touretzky ed., San Mateo, CA, I Morgan Kaufmann Publisher, 1989.

[40] L.A. Rowe and K.A. Shoens, "Programming laguage constructs for screen definition," *IEEE Transactions on Software Engineering,* vol. 9, pp. 35-39, 1983.

[41] D.E. Rumelhart, J.L. McClelland the PDP Research Group, *Parallel Distributed Processing,* MIT Press, Cambridge, Mass., 1986.

[42] C. Rutkowski, "An introduction to the human applications standard computer interface, Part 1: Theory and principles, *BYTE,* vol. 7, no. 11, Oct. pp. 291-310, 1982.

[43] R. Schvaneveldt and et al., *A Taxonomy of Human-Computer Interactions: Toward a Modular User Interface,* J. of Human-Computer Interaction, E.G. Salvendy, Amsterdam: Elsevier, pp. 121-124, 1984.

[44] T.J. Sejnowski and C.R. Rosenberg, "Parallel networks that learn to pronounce English text," *Complex System,* vol. 1, pp. 145-168, 1987.

[45] B. Shneiderman and R. Mayer, "Syntactic/ Semantic interactions in programmer be-

havior: A model and experimental results, *Intl. J. of Computer and Information Science,* vol. 8, no. 3, 1979.

[46] B. Shneiderman, *Software Psychology: Human Factors in Computer and Information Systems,* Little, Brown and Co., Boston, MA, 1980.

[47] B. Shneiderman, "A note on the human factors issues of natural language interaction with database systems," *Information Systems,* vol. 6, no. 2, pp. 125-129, 1981.

[48] B. Shneiderman, "Direct manipulation: A step beyond programming languages," *IEEE Computer,* vol. 16, no. 8, pp. 57-69, Aug. 1983.

[49] B. Shneiderman, *Designing the User Interface,* Addison-Wesley Pub. Co., Boston, MA, 1987.

[50] S.J. Stolfo, "Initial performance of the DADO2 prototype," *IEEE Computer,* vol. 20, no. 1, Jan. 1987.

[51] H. Thimbleby, *What you see is What you have got?,* Unpublished Paper, University of York, England, 1982.

[52] Benjamin W. Wah, "New computers for AI processing," *IEEE Computer,* vol. 20, no. 1, Jan. 1987.

[53] R. Watrous, *Speech Recognition Using Connection Networks,* TR-MS-CIS-88-96, Dept. of Computer and Information Science, University of Pennsylvania, 1988.

[54] A. Waibel, T. Hanazawa, G . Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics Speech and Signal Processing,* ASSP-37, March 1987.

[55] D.L. Waltz, "Applications of the connection machine," *IEEE Computer,* vol. 20, no. 1, Jan. 1987.

[56] J.W. Yang and D.H. Kim, Neural Network That Converts Korean Words Into Phonetic Symbols, Proceedings of nEuro'88, Paris, France, pp. 551-559, 1988. 8.

筆 者 紹 介

林 英 煥

1954年 9月 16日生
1977年 2月 경북대학교 졸업
1979年 2月 한국과학원 전산학과 졸업(석사)
1985年 6月 미국 노스웨스턴대(Northwestern University) 전산학과 졸업(박사학위)

1979年 1月～1982年 8月 한국전자기술연구소, 선임연구원
1985年 6月～현재 한국전자통신연구소 책임연구원, 인공지능 연구실장