□ 特輯 : 電子計算機 □

人工知能技法

金 元 燮
(全北大 工大 教授)

■ 차 례 ■

1 緖 論

人工知能이란 무엇인가?

인간의 知的行動을 인간과 같이 行하는 컴퓨터시스템을 人工知能(AI : artificial inteligence)이라 한다. 知能을 가진 기계나 생물을 人工的으로 만드는 아이디어는 옛날부터 있었으며 書籍이나 物件 繪畵 等에서 종종 볼 수 있다. 그러나 硏究의 수단이 없었기 때문에 그들의 아이디어는 空想의 범위를 벗어날 수가 없었던 것이다.

그런데 컴퓨터의 實現으로 인하여 이 아이디어는 단순한 空想에서 實際로 硏究의 대상이 될 수 있는 것으로 변화되어 왔다. 그 實現의 可能性과 방법에 대한 探求가 計算機科學의 硏究者들에 의해서 시작되어 人工知能은 科學으로서의 한分野를 형성하는데 이르게된 것이다. 사실상 人工知能의 역사는 30년을 上廻하고 있으나 試驗的인 性格을 벗어나 實社會의 문제를 다루기 시작한 것은 최근의 일이다. 1970年代末까지도 이 분야의 先進國이라 할 수 있는 美國에서 조차 인간의 高度한 知能에 속하는 일을 기계에 思考케하는 것은 불가능한 일이라 생각하고 硏究投資에 消極的이였던 것이다. 그러던 것이 1982年 4月, 日本 通商省(MITI)의 정책적인 主導下에 新世代컴퓨터 開發프로젝트(FGCS), 硏究機關으로 ICOT(신세대 컴퓨터기술개발기구)가 발족되어 10個年 長期計劃으로 政府投資 4억 5천만불, 민간기업체투자 5억불 등 9.5 억불을 總投資하여 본격적인 5世代컴퓨터開發을 着手하여 착착 中間結果가 成功하게되자 컴퓨터의 先進國이라 할 수 있는 미국도 國防省의 DARPA의 10억불, 18개 民間企業體가 結成한 MCC를 비롯하여 SRC, MCNC같은 共同機構를 組織하여 박차를 가하므로써 이제 人工知能 컴퓨터의 출현은 다만 時間問題라 할 수 있을 정도로 成功이 確定的이다.

그런데 여기서 人工知能研究의 대상이 되는 「知的行動」이란 무엇을 말하는 것인가를 명확히 해두지 않으면 안된다.

컴퓨터는 人間代身에 數學的인 계산을 시키기 위해서 發明된 것이다. 계산의 속도와 정확함에 있어서는 인간을 훨씬 능가하는 能力을 갖고있으며 그 때문에 컴퓨터出現 이전에는 계산할 수 없었던 大量의 데이터를 처리하는 것도 가능하게된 것이다. 가령 벡터場에 의해서 演算을 行하거나 1000元連立 一次方程式을 풀거나 하는 일들이다. 통상 方程式을 풀기 위해서는 知性의 작용을 요하는 것이나 그렇다면 連立一次方程式을 푸는 컴퓨터는 人工知能이라 할 수 있을까.

答은 否라 할 수 있는 것이다. 혹은 적어도 그와같은 계산에 관한 能力은 人工知能 研究의 대상으로는 되지않는다. 왜냐하면 連立一次方程式에는 解를 구하기 위한 완전한 순서가 있어 그 순서를 機械的으로 좇아서 하면 되기 때문이다. 컴퓨터는 그것을 대단한 高速으로 實行하는 것뿐이며 계산을 行하는 順序는 外部에서 인간에 의해서 주어진다. 인간을 훨씬 능가하는 計算能力을 갖고 있는 반면 컴퓨터는 인간이 간단히 하고있는 「보거나, 듣거나, 理解하거나」하는 것은 대단히 다루기 어렵다. 컴퓨터가 무엇인가를 하는데는 그것을 위한 완전한 順序가 항상 필요하며 한편 인간이 「보거나, 듣거나, 理解하거나」하기 위한 順序는 충분히 解明되어 있지 않기 때문이다. 그 다루기 어려운 일을 컴퓨터에 시키려고 하는 것이 人工知能研究의 目的이다.

또 이론적으로는 완전한 順序가 존재하는 問題라도 그 順序를 좇아서 하는데는 막대한 時間이 걸리기 때문에 현실적으로는 다른 方法을 필요로 하는 경우도 있다. 가령 체스(Chess) 의 서로 다른 對局의 種類는 10^{120}가지 있다고 한다. 그 모든 것을 探索하면 다음에 指定할 最長의 한수가 구해지는 것이 되어 체스에 이긴다고 하는 問題는 解가 얻어진다. 그러나 합리적인 時間內에서 最良의 方法을 발견하기 위해서는 모든 가능한 對局을 探索하지 말고 그것을 구할 필요가 있다. 인간의 체스플레어도 그와 같이 할 셈으로 「이렇게 하면 원만히 잘 된다」고 하는 휴리스틱스(經驗則 : 체스의 경우의 定石 등)를 適用하면서 探索의 범위를 좁혀져 있다고 생각된다. 그 制限의 방법도 人工知能의 주요테-마의 하나이다.

人工知能이 실현되면 從來 인간이 하지 않으면 안되었던 판단이나 意思決定을 기계가 일부 혹은 모두 대행하게 되어 應用은 工學의 모든 분야에 미친다. 그 有用性은 명백하다.

또 인간의 知的行動을 기계에 模倣시키는 것을 통하여 인간의 知能 그 自體에 관한 解明에도 퍽 有益하리라고 기대된다.

[2] 人工知能의 諸分野

人工知能에서 취급되는 여러 研究分野를 專門的인 상세함을 피하는 방향에서 言及하려고 한다. 인간의 知的行動의 대행을 주 테-마로 한 人工知能의 연구는 學問的 色彩가 짙은 분야로 表1과 같은 극히 多方面에 걸치는 研究分野를 包含하고 있다.

인간의 認識의 構成에 관한 研究, 問題表現 狀態 空間의 探索 知識表現등의 基礎的 研究로부터 시작하여 自然語理解·機械翻譯 自動programming Expert ssstem 등 시스템構築法에 대한 研究分野에 이르기까지 여러 分野가 人工知能의 틀속에 포함된다. 더우기 計算科學과는 직접 關係가 없는 科學·工學의 諸分野에 있어서도 人工知能의 성과를 受容하려는 움직임이 활발하여 人工知能研究의 동향에 關心을 갖는 技術者는 相當數에 올라있는 것으로 思料된다.

表1. 人工知能研究分野

人工知能	問題解決
	自動演澤
	知識表現
	自然言語理解
	expert system
	自動programming
	學習
	computer vision

특히 80年代에 들어와서 몇 分野에 있어서 人工知能의 應用이 採擇되어 Expert system開發道具나 機械翻譯에서는 實用製品이 나오기 시작하여 널리 一般의 관심을 끌고 있다.

人工知能은 아직 幼兒段階의 學問分野이고 現在 周邊諸分野와 關運하면서 急速度로 成長 變化하고 있다. 그렇기 때문에 全體像을 명확하게 파악하는 것은 반드시 용이한 일은 아니다. 그러나 人工知能의 研究는 計算機에 知的能力을 주기 위한 共通課題와 구체적인 知的行動에 대해서 實驗하고 아이디어의 正當性을 확인하는 應用分野로 나눠 생각할 수도 있어 이런 意味에서 그림 1은 그 一例로서 4개의 基本課題와 8種의 應用分野를 나타내고 있다. 이를테면 서로 깊은 關聯을 갖고 있어 하나씩 獨立的으로 해결되는 것이 아니고 分類도 便宜的이다. 여기서는, 最近着된 外誌에서 人工知能에 관한 주된 研究分野를 概說하고 소개하는 정도로 그친다.

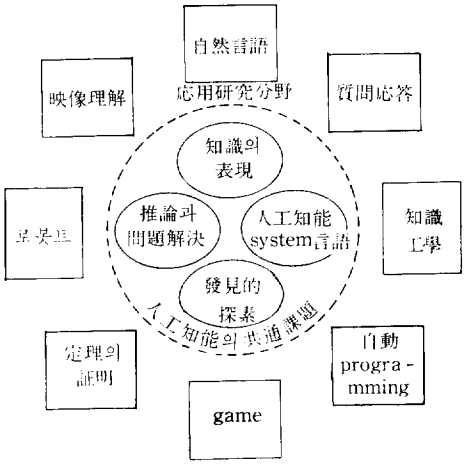


그림 1. 人工知能의 研究分野

2.1 探　索

지금까지 情報處理는 인간이 주어진 課題를 푸는 方法을 생각해서 그것을 프로그램 言語로 써내려가 計算機는 그것에 따라서 論理演算을 하는데 不過하였다. 換言하면 處理의 順序 즉 how를 준 情報處理이다.

人工知能의 研究에서는 기계에 達成할 課題 즉 what를 주면 獨力으로 그 目標에 到達하는 順序를 생각하는 것이 그 특징이다. 이것이 共通課題의 하나의 問題解決(problem solving)이며 人工知能의 各分野에 있어서 그 手法은 모두 받아들이고 있다.

초기의 研究에서는 주어진 問題를 ㉮ 현재의 狀態에서 目標에 도달한 graph 또는 ㉯ 보다 용이한 문제와 分解하는 AND－OR그래프로 變換하여 發見的探索(heuristic search)에 의해서 문제를 푸는 理致를 발견하였다. 그래프의 노－드(Node)數가 현저하게 많아 有望할 듯한 理致만을 깊게 探索하는 것이 필요하여 評價函數를 이용하여 分岐의 選擇을 하는 手法이 게임의 플레이 等의 예에서 研究되었다.[18] 취급하는 대상의 범위가 정해지면 그 분야에서 成立하는 많은 法則을 활용할 수 있어 探索의 能率은 현저하게 향상된다. 즉 法則에서 論理的으로 존재하지 않는 理致는 제외할 수 있어 깊은 論理思考가 가능하게 된다.[19] 物體認識의 研究에 있어서 積木의 世界의 線畵의 解釋은 이 一例이며[19),20)] 音聲理解, 自然言語處理 등 적용범위가 넓고 人工知能의 하나의 유력한 道具가 되어있다.

지금 체스와 같은 두 사람의 플레어가 交互로 한수씩 하는 타잎(type)의 게임에 대하여 생각한다. 퍼즐(puzzle)이나 게임에 있어서 모든 가능한 상태의 集合을 狀態空間이라 한다. 개개의 상태는 체스로 말하면 어느 時點에 있어서 盤面에 대응하고 있다. 이중에서 게임 開始時의

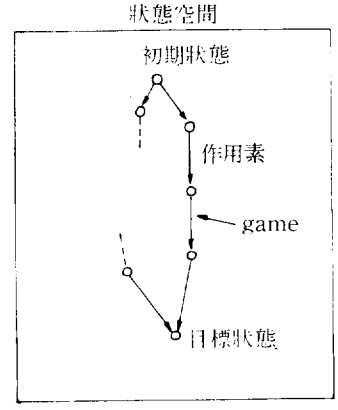
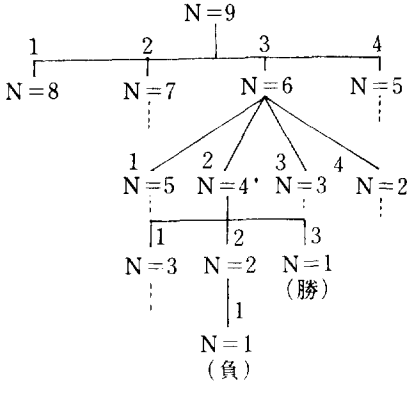


그림 2. 狀態空間의 作用素

盤面에 대응하는 상태를 初期狀態, 終了時에 대응하는 상태를 目標狀態라 한다. 체스의 駒를 한수 움직이면 상태는 별도의 상태로 옮겨가나 이 한수를 作用素라 한다. 作用素는 상태의 集合을 定義域 및 値域으로 취하는 函數이다.

이와 같이 생각하면 체스에 勝利한다는 問題는 初期狀態에서 目標狀態로 이르는 作用素의 系列을 발견한다고 하는 問題에 置換된다. 이와 같은 表現을 문제의 狀態空間表現이라 하는데 그림 2는 狀態空間의 槪念을 나타낸다. 또 初期狀態를 根으로 하고 作用素를 가지(枝)로 한 게임木에 의해서 게임의 모든 가능한 플레이는 表示된다. 가령 그림 3은 경기게임의 게임木의 一部이다.



N개의 棒에서 1～4개씩 取合하여
最後의 1개를 取하면 敗하게 된다.
圖는 N29의 경우의 게임木의 一部

그림 3. 棒取合게임木

이론적으로는 게임木을 샅샅이 뒤져 探索하므로서 게임의 解는 모두 얻어지는 셈이 된다. 그러나 그와 같은 探索法으로 자명한 問題以外의 것을 푼다는 것은 불가능이라 할 수 있다. 왜냐하면 探索하는 경우의 數는 手數N에 대하여 指數函數的으로 增大하기 때문에 일반의 문제에서는 대단히 큰 數가 되어 合理的인 時間內에 探索을 마칠 수가 없기 때문이다. 이 현상은 組合論的 爆發이라 불리우며 거의 모든 人工知能의 分野에 공통의 困難이다. 그래서 探索이라는 分野의 中心的課題는 여하히 探索할 節點의 數를 減少시켜 探索에 걸리는 시간을 합리적인 時間內로 거둬들이냐의 문제이다. 이 때문에 다음과 같은 수법이 硏究되어 있다.

㉮ 狀態空間이 적게 되도록 問題 그 自體를 바꾸어 만든다. 이 수법의 예로서 가끔 나오는 것이 그림 4에 나타낸 체스보드의 問題이다. → 이것은 A와 같은 兩隅가 欠된 체스보-드를 B의 板으로 埋立해 가는 것을 실시한다. 盲目的으로 探索하는 경우 探索空間은 크나 B가 赤黑의 枡目을 하나씩 埋立하는 것에 着眼하면 문제는 自明하다.

㉯ 휴리스틱스(經驗則的인 知識)을 이용하여 探索經路를 제한한다.

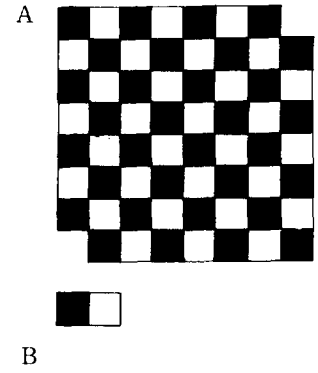


그림 4. 체스보드

2.2 知識表現 이용과 학습

人工知能시스템의 特徵은 大量의 知識을 有效하게 이용하여 주어진 課題를 解決하는 것이다. 물론 課題마다 어느 知識을 이용하는가는 計算機가 獨力으로 探索하고 발견한다. 따라서 활용하기 쉬운 形으로 知識을 表現(Representation of Knowledge)[19),21)] 하고 시스템이 이들의 知識을 獲得하는(Knowledge acquistion) 것은 人工知能의 중심의 課題이다.

知識에는 「그 課題에 대하여 성립하고 있는 사실」, 「사실에 대하여 성립하는 法則에 대한 知識」, 「知識을 조합시켜 問題를 푸는 方策에

관한 知識」등 여러가지 Level의 것이 있어 그들을 計算機內로 蓄積하여 이용하기 위한 많은 방식이 研究되었다.

초기의 研究에서는 表[23]나 評價函數形으로 표현하였으나 도저히 복잡한 問題는 取扱할 수 없다. 다음에 1階의 述語論理式을 이용하여 事實·法則을 表現하여 導出原理(Resoulution principle)[18]나 GPS(General Problem Solver)[23]를 이용하는 質問應答시스템[24]이나 로봇트의 頭腦[25]가 研究되었다.〔그림 5.ⓐ〕이들은 事實·法則의 知識이 한결 같이 모듈構造로 자유로 追加削除가 가능하고 또 어떠한 問題에도 이용할 수 있으나 그 反面 解를 발견하는 능력이 나쁘고 실용성이 부족하다.

手續型知識(Procedural Knowledge)[26),27)] 은 사실과 법칙을 별도의 형식으로 표현한다.〔그림 5.ⓑ〕 법칙은 데이터를 操作하는 사브르찡形으로 蓄積되어 주어진 課題에 대한 시스템의 制御構造가 이들을 아크세스시켜 解를 구한다. 이 特徵은 사브르찡을 呼出하는데 그 내용을 나타내는 패턴(pattan)을 이용하는 것으로 推論過程에서 차차로 적절한 知識을 아크세스시켜 깊은 論理思考에 의해서 제법 복잡한 問題도 능률좋게 해명할 수가 있다. 그러나 그 때문에 개개의 問題에 대하여 유효한 形態의 知識을 프로그램할 필요가 있고 또 知識사이에 相互作用이 있으므로 擴張性에 問題가 있다.

상술한 2種의 方式의 問題點을 극복하기 위하여 學習시스템HACKER가 提案되었다.〔그림 5.ⓒ〕[28] 취급하는 問題에 대한 法則을 인간이모듈構造로 준다. 具體的問題를 入力하면 HACKER는 問題를 실제로 풀면서 手續形知識의 解法을 작성한다. 이 知識은 프로그램이므로 인간이 프로그램을 만드는 것과 같은 과정에서 만들어진다. 즉 HACKER는 試作한 프로그램을 실행하여 誤謬를 발견하고 그것을 分類하여 대책을 생각하여 프로그램을 作成한다. 이 手續을 返復하는데 이상의 프로그램作成過程에 있어서 成功이나 失敗의 結果를 일반화하여 要約하여 시스템으로 記憶시켜 問題解決의 방법을 學習한다. 이와 같이 HACKER는 學習能力을 갖는 自動프로그래밍시스템이라 말할 수 있다.

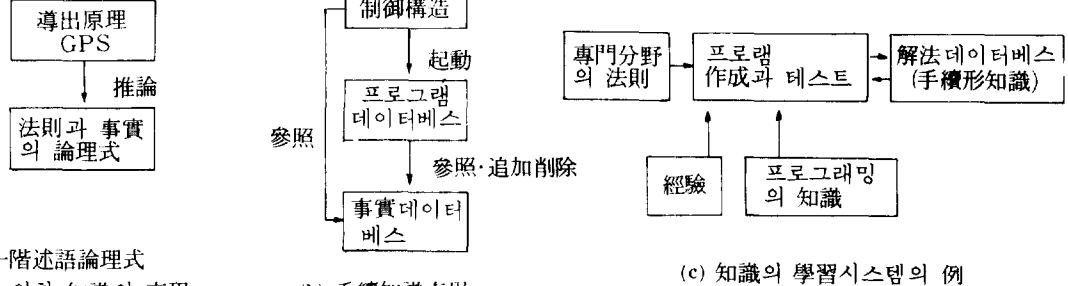
膨大한 知識을 취급하는데는 知識單位의 어느 種類의 統合으로 현재 想起하고 있는 知識에 關連깊은 것과 그렇지 않는 것을 區別하는 메카니즘이 필요하다. 關連하는 概念에 링크를 付하므로서 連想能力을 갖는 세만틱네트웍(semantic network)[29]는 그 一例이지만 밍스키(M. Minsky)는 후렘(Frame)[30]이라는 概念을 導入하여 각각의 場面에 대응한 知識單位를 아크세스하는 방식을 提案하여 이후의 研究에 큰 영향을 주었다.

知識表現形式의 能力은 일반으로 다음과 같은 觀點에서 평가된다.

ㄱ) 대상으로 하는 세계의 事象을 충분히 표현할 수 있는지 없는지의 與否.

ㄴ) 컴퓨터에 의한 操作, 즉 표현된 知識에서 새로운 知識을 引出하는 것이 용이하게 할 수 있는지 없는지

知識表現은 人工知能의 各分野에 크게 영향을



(a) 一階述語論理式에 의한 知識의 表現

(b) 手續知識表現

(c) 知識의 學習시스템의 例

그림 5. 知識의 表現

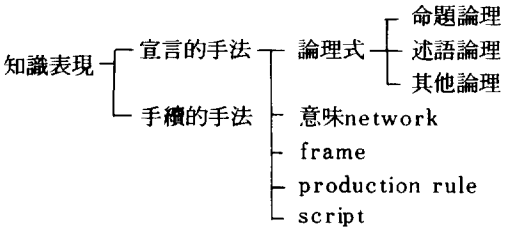


그림 6. 주된 知識表現手法

가져다주는 分野이고 人工知能의 基盤을 지탱하여 주는 課題의 하나라고 할 수 있다. 현재의 여러가지 知識長現手法은 어느 것이나 一長一短을 갖고 있어 각각의 特徵에 마춰서 이용할 필요가 있다. 그림 6은 주된 知識表現手法을 나타낸 것이다.

2.3 推 論

一定의 知識表現形式으로 표현된 事實, 혹은 事實間의 關係를 記述한 規則을 조작하므로써 知識으로써 表面에는 記述되지 않았던 새로운 事實을 引出하는 것을 推論이라 한다.

推論의 매카니즘과 그것이 操作하는 知識表現과는 서로 밀접한 관련을 갖고 있으며 어느쪽인가를 단독으로 취급할 수는 없다. 推論매카니즘은 知識表現形式에 의하여 제약을 받으며 또 그 逆도 성립한다.

特定의 表現形式에 의한 知識의 데이터 베스(知識베스)와 推論매카니즘(推論엔징)을 組合시키므로서 현재 expert system이라 불리는 應用分野가 열려있다. 이들의 시스템은 前述의 파즐(puzzle)이나 게임等이 주가 되는 장난감問題를 떠나 實用的인 問題의 解決에 人工知能을 응용하는 것을 目標로 하고 있다.

2.4 自然言語理解

우리 인간이 통상 사용하는 言語를 프로그램言語에 대하여 自然言語라 한다. 計算機에서 自然言語가 취급될 것을 論한 1950年代 中半에서 언어의 自然翻譯의 어려움이 認識하게 되어 말이 갖는 意味의 研究를 하지 않으면 안될 것이 명백히 되었다. 1970年代에 들어와서 말의 意味뿐 아니라 文脈이나 社會的인 知識과 演繹·推論의 能力이 文章의 解析에 필요하다는 것이 명백히 되었다. 計算機에 의한 自然言語의 理解를 위해서는 이들 모든 要素를 포함하여 이들을 柔軟하게 사용할 수 있는 시스템을 만들 것이 필요하게 된다. 1970年代에 들어와 人間과 計算機와의 對話 質問應答시스템등이 열심히 研究되도록 되어왔으나 自然言語의 理解라는 것의 構造가 명백히 되지않으면 이와 같은 시스템은 성공하지 않는다고 생각된다.

一方 計算機가 小型化 廉價가 되면서 事後處理등 여러 分野에 사용되어 데이터베스와 情報檢索이 번성함에 따라 計算機가 自然言語를 취급하는 것이 實用上에서 중요한 問題가 되어 왔다. 이러한 背景에서 自然言語處理의 研究가 한창 번성하게 된 것이다. 금후 언어의 構造가 보다 명확히 되어 文脈의 構造, 對話의 흐름의 構造가 명백히 되어가면 自然言語가 計算機에의 아크세스의 하나의 有力한 수단이 될 것이다. 또, 이와 같은 研究成果의 結果 言語의 自動翻譯도 재차 研究되어 實用의 段階까지 도달할 가능성이 나올 것이다.

自然言語는 인간의 知識의 構造에 가장 잘 적응한 언어이며 대단히 우수한 知識表現能力을 갖고 있다. 그러나 표현의 構造가 복잡하기 때문에 컴퓨터에 의한 操作으로 意味의 解析을 行하는 것이 현재는 아직 곤란하며 활발한 研究의 課題가 되어 있다.

自然語에서 意味를 取出하는 것이 가능하게 되면 먼-머신-인터훼스는 비약적으로 向上하는 일이 豫想된다. 또, 人間의 言語行動自體에 대해서도 어떠한 知見을 얻을 수가 있을 것이다. 그러나 현재까지의 自然語理解 시스템이나 機械翻譯시스템의 研究에서 實用이 되는 시스템의 出現에는 아직 상당한 研究期間을 요하는 것으로 예상되어 있다.

自然語理解의 技術을 이용한 시스템으로서는 質問應答시스템, 機械翻譯시스템, 要約시스템등이 있다.

自然語를 解釋한다고 하는 것은 自然語에 의한 入力文을 處理하여 元文의 의미에 대응하는 內部表現을 生成하는 것으로 생각된다. 그 일반

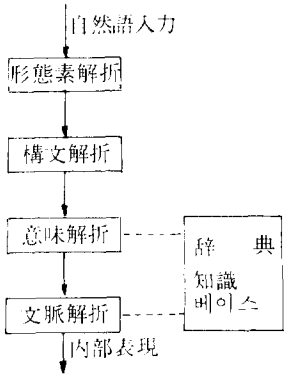
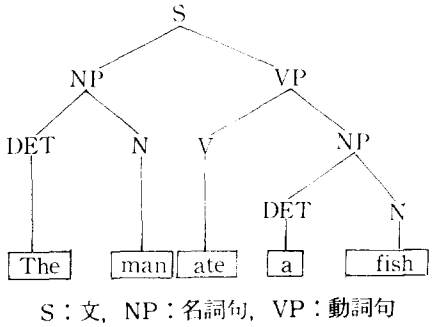


그림 7. 自然語處理 手順

적인 處理手順은 그림 7에 나타낸 바와 같이 몇 段階의 처리로 나눌 수가 있다.

(ㄱ) 形態素解析 : 이것은 文을 구성하기 위한 基本單位이며 英語의 單語나 接尾語, 日本語의 語幹이나 語尾등도 각각 하나의 形態素라 생각할 수가 있다. 入力文을 形態素에 分割하는 작업이 形態素解析이다. 日本語와 같이 文章이 單語로 구별되지 않는 言語의 形態素解析은 英語등에 비해 보다 곤란이 크다.

(ㄴ) 構文解析 文中에 있어서 單語끼리의 關係(單語의 品詞나 修飾 / 被修飾등)을 抽出하는 段階이며 構文解析을 위해서는 文의 構造를 記述하는 文法과 單語의 文法情報를 記載한 辭典이 필요하다. 構文解析을 위한 다음과 같은 것이 있으며 構文解析의 結果 얻어지는 內部表現은 사용하는 文法의 種類에 의존한다.



S : 文, NP : 名詞句, VP : 動詞句
N : 名詞, V : 動詞, DET : 冠詞

그림 8. 句構造文法에 依한 文章解析例

(a) 句構文法
(b) 變形生成文法
(c) 格文法
(d) 擴張遷移文法

그림 8에 句構造文法에 의한 文의 解析例를 나타낸다.

(ㄷ) 意味解析

입력된 文章이 어떠한 意味情報를 가지고 있는가를 抽出하는 처리이다. 意味解析을 行하기 위한 일반적인 방법은 아직 존재하지 않으나 原則으로는 辭典中에 登録되어 있는 各單語의 意味記述을 句構造마다 정리하여 최종적으로 文의 意味表現을 얻는다는 手段으로 行해진다. 意味表現形式으로서는 意味네트워크, 후렘, 述語論理등이 사용된다.

(ㄹ) 文脈解析

일반으로 文章은 전혀 독립으로 존재하여 外部의 影響없이는 意味를 가진 것은 아니고 前後의 文章과 相互로 영향을 주면서 각각의 의미를 갖는 것이다. 이와 같은 文脈의 처리를 시스템으로 여하히 行할 것인가가 현재 주목되어 있는 문제의 하나이다. 이와 같은 各段階를 거쳐 얻어진 內部表現을 사용하여 시스템은 翻譯·要約·應答文의 生成등의 처리를 行하는 것이 된다. 이 분야의 중요한 課題로서 常識의 문제를 들 수 있다. 인간들 사이의 對話에 있어서는 對話者의 双方이 共有하고 있는 것을 前提로 한 一般常識的인 知識이 이용되고 있으며 그 部分은 文中에는 表現되지 않는다. 이 막대한 量의 一般常識을 여하히 컴퓨터에 저장시켜 檢索, 利用하는 것이 문제가 된다.

또 인간은 對話中에 무엇에 대한 對話인가를 理解하고 있으며 그 知識을 이용하여 다음에 와야할 文章의 내용에 대해서 미리 어느정도 豫測하고 文章의 理解를 용이하게 하는 것으로 생각된다. 이러한 「豫期」의 能力을 컴퓨터上에 實現하기 위한 方法을 檢討하는 것은 극히 중요한 문제이다.

2.5 音聲理解

인간의 音聲, 즉 말自體를 理解하는 시스템構

築에 관한 테마로 前述의 自然語理解와도 밀접히 관련되어 있다. 먼-머신 인터훼스의 向上이라는 應用面에서도 그 성과가 기대되고 있는 분야이다.

音聲認識과 後述의 畵像認識과는 공히 패턴認識이라 불러지는 硏究領域에 속하는 분야이다. 가령 音聲의 경우 패턴은 귀에 들리는 音聲의 波形이고 그것을 解析하여 무슨 單語인가를 決定하는 것이 패턴認識의 프로세스이다.

視聽覺에 한하지 않고 인간이 패턴認識을 行하는 프로세스의 상세한 것은 거의 알 수 없다. 音聲理解시스템에 있어서는 다음과 같은 아프로-치를 들 수 있다.

ㄱ) 미리 준비된 音素나 單語의 標準 패턴과 入力된 音聲의 패턴과를 비교하여 그 類似性에 의해서 判斷을 行한다.

ㄴ) 入力 패턴의 여러가지 性質에 着眼하면서 一連의 判定을 順次 行하고 判斷을 行한다.

그외 音聲理解시스템은 發話者의 制限(特定話者/不特定話者), 入力語의 形式(離散語/連續語) 分析의 單位(音素單位 / 單語單位)등 觀點에서 分類된다.

音聲理解시스템硏究의 目標가 認識率의 향상에 있는 것은 말할 필요도 없으나 이것을 實用上問題가 없는 線까지 향상시키는 것은 꽤 곤란하며 현재의 實用시스템에서는 사용하는 單語의 種類를 數十單語에 묶으므로서 所期의 性能을 얻는 것이 實情이다. 특히 單語가 連續하는 경우에는 各單語의 發音이 前後의 單語에 영향되어서 변화하기 때문에 간단한 單語Level의 패턴맛칭으로는 처리할 수 없다. 認識率을 올리기 위한 별도의 工夫가 필요하게 된다.

그 때문에 단순한 音聲學的인 처리뿐아니라 入力文의 意味情報를 이용한 처리를 行하므로서 認識率을 올리는등 방법이 硏究되고 있다. 따라서 이 경우에도 前述의 常識이나 豫期등 사항이 문제가 된다.

2.6 畵像認識

TV카메라 등에서 入力된 畵像을 처리하여 그 畵像中에 포함하는 대상을 認識하는 硏究이다. 인간의 눈에 해당하는 機能 로봇트 等의 機械에 갖게하는 응용이 생각된다.

畵像認識의 처리는 다른 패턴認識의 處理와 같이 그림 9 와 같은 흐름에 따라서 行해진다. 그러나 畵像情報는 音聲情報에 비해 情報가 갖는 패턴이 극히 變化가 많아 認識의 대상은 일반으로는 대단히 좁은 범위로 限定되어 그 위에 비교적 臨機應變的인 데이터構造와 앨고리즘이 이용된다. 그 때문에 반드시 일반적인수법이 존재 하지는 않는다.

畵像認識은 대단히 곤란한 硏究分野의 하나이며 본격적인 實用化의 전망은 거의 없으나 로봇트工學관련의 應用시스템으로 극히한정된범위의 대상을 인식하는 것은 몇가지 볼 수 있다.

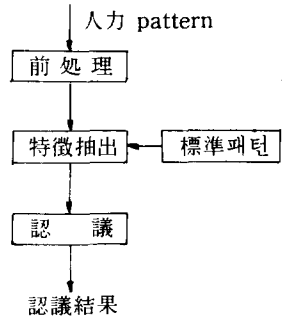


그림 9. 畵像認識 處理

2.7 로봇트工學

視覺·觸覺等의 感覺으로 環境을 인식하여 手足에 의해서 그 環境內에서 행동하는 自動機械를 로봇트이라 한다. 로봇트는 그림10과 같이 感覺, 行動, 知能의 3개의 사브시스템으로 구성된다. 環境의 理解, 指令의 解釋, 問題의 解決, 行動의 計劃, 感覺에 의한 행동에의 歸還 等의 知的機能을 현실의 環境으로 움직이는 機械로서 실현하는 일이 知能로봇트의 주된 硏究課題이다.

計算機를 사용한 로봇트의 硏究는 1960年境 시작되었다. 計算機制御의 기계의 팔(腕)[22] 는 1962年에 TV카메라에서 본 것(積木)을 計算機로 인식시키는 硏究[23]는 1963年에 처음으로 발표되었다. 1960年末부터 1970年代에 걸쳐 TV

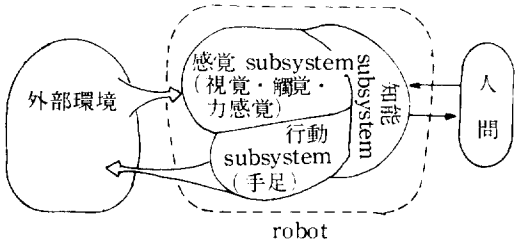


그림10.

카메라로 物體를 보고 기계의 팔로 그것을 핸드링하는 시스템이 몇가지 개발되었다. 이들의 초기의 知能로봇트의 환경은 積木의 세계에 한정되어 있었으나 視覺技術의 실용화에 隨陣하여, 70年代의 오토에손의 變化, 觸覺·力感覺의 導入도 기대되어 있어 더우기 마이크로프로세서를 活用한 制御의 高度化, 産業로봇트의 知能化도 推進되고 있으며 知能로봇트의 應用分野로서 오토메이숀은 성장하고 있다.

로봇트는 元來 인간의 手作業을 대행하기 위하여 발명된 기계로 그 操作프로그램에 人工知能의 成果를 응용하려고 하는 것은 당연한 결과라고 볼 수 있다. 이 분야의 주테-마로서는 다음과 같은 것을 생각할 수 있다.

㉠ 개개의 동작의 最適化

㉡ 어느 目標에서 그것을 달성하기 위한 動作系列을 발생시키는 研究

㉢ 視覺을 갖고 대상을 認識하면서 作業을 進行시키는 로봇트의 研究

2.8 Expert system

Expert system이란 인간의 專門家의 일을 一部 代行하여 特定分野의 問題를 풀거나 助言을 준다거나 하기 위한 프로그램이다. 즉 特定分野의 專門知識을 知識베이스로서 格納하여 두고 특정의 推論매카니즘을 사용하여 그들의 知識을 操作하여 그 분야의 문제를 풀거나 一般 利用者에의 콘설팅(consulting)을 行하는 시스템을 expert system이라한다. 그림11은 이 system 의 基本構成을 나타낸 것이다. 專門家의 일의 一部를 代行하는 것이라 할 수 있다. Expert system

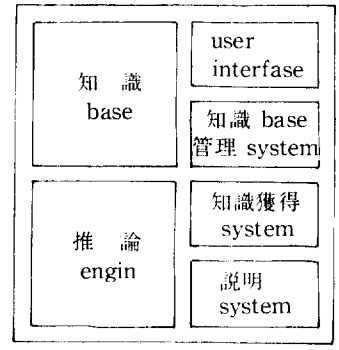


그림11. expert system基本構成

의 構築에 관한 研究分野를 知識工學이라 한다.

知識工學은 스탠포드大學의 Feigenbaum제창에 의한 것으로 人工知能研究의 성과를 「장난감問題」가 아닌 실용적인 問題解決에 應用하려는 시도라 볼 수 있다.

이 분야의 중심적인 과제는 실용적인 문제를 해결할 수 있는 막대한 양의 經驗則的인 지식을 시스템에 移植하여 問題解決에 이용하는 것이다.

Expert system은 현실점에서 가장 실용화에 가깝다고 생각되는 분야이고 많은 周邊分野에서 세찬 관심이 쏟아지고 있다. 그러나 Expert sy-

表 2. Main expert system

名　　称	目　　的	關發機關
AIRPLAN	空港管制業務支援	ATT
DENDRAL	質量分子構造決定	stanford大學
DRILLING ADVISOR	油田採掘用	elf-architenu
MACSYMA	數式處理	MIT
MOLGEN	分子生物學實驗計劃	stanford大學
MYCIN	血液感染症診斷	stanford大學
PROSPECTOR	地質調査鉱脈探査	SRI
PUFF	肺疾患診斷	stanford大學

stem의 構築에는 막대한 受苦와 시간이 걸리므로 코스트도 대단히 높고 그 때문에 Expert system開發을 위한 道具의 出現이 기대되고 있으나 최근 汎用의 推論매카니즘과 知的베이스 構築支援시스템을 조합시킨 제품이 數種類 발표되어 있다.

Expert system과 開發道具에 대해서는 後日 譯述하기로 한다. 表2는 주 Eepert system을 나타낸 것이다.

2.9 自動프로그래밍

컴퓨터가 동작하기 위힘서는 그 동작의 詳細한 方法을 記述한 데이터(프로그램)을 주지 않으면 안된다. 그 순서의 기술은 컴퓨터가 이해할 수 있는 언어로 되어있을 필요가 있다. 그러기 위하여 설계된 언어를 프로그래밍 언어라 한다. 컴퓨터가 직접 實行할 수 있는 것은 機械語라 불리는 코-드이며 초기의 프로그래밍은 機械語를 이용하여 되었다. 이윽고 機械語와 1대1로 대응하는 니모-닉크 코드에서 機械語를 발생하는 아셈브러나 數式이나 記號列을 機械語로 번역하는 콤파일러등의 프로그램에서 만들어져 현재는 프로그래밍 언어라 하면 이들을 가르키는 것이 보통이다.

아셈브러나 콤파일러에 의해 프로그래밍의 능률을 향상되었으나 거기에도 자연히 한계가 있다. 특히 近年이 되어 소프트웨어 需要가 급증을 계속하여 있는데 대하여 프로그래밍의 能率向上의 限界가 보이기 시작한 점에서「소프트웨어의 危機」가 부르짖기에 이르렀다.

이것에 대하여 보다 인간의 思考形式에 부응한 언어로 프로그램의 記述을 行하고 그 記述을 컴퓨터에 의해서 프로그래밍 언어에 의한 記述로 변환하는 시스템의 硏究가 行해진다. 이것을 자동프로그래밍이라 한다. 그림12는 자동프로그래밍의 槪念을 나타낸 것이다.

이것이 실현되면 이용자는 컴퓨터에 시키고져 하는 내용의 記述을 보다 自然語에 가까운 언어로 行하는 것이 가능하게 되어 프로그래밍에 요하는 시간을 단축할 수가 있다. 자동프로그래밍 시스템에 입력하기 쉬운 프로그래밍 記述을 仕

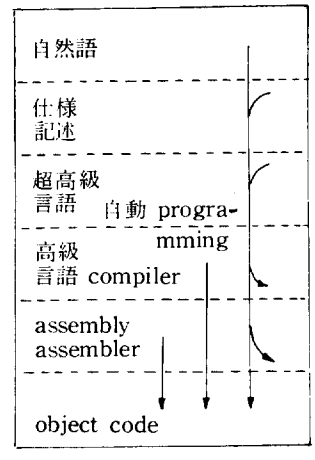


그림12. 自動programming槪念

様記述이라 부르며 프로그램 언어와 구별하고 있다.

仕様記述의 形式으로서는 발생시키고저 하는 프로그램의 入力-出力의 組를 입력하는 것, 시스템과 인터럭티브하게 대화하면서 目的프로그램을 생성하여 가는 것 등이 있다.

이상 人工知能의 각 분야를 槪說하였다. 이들 분야 이외에도 많은 응용분야가 있으나 모든 것을 언급한다는 것은 불가능하다.

전체적으로 보아 아직 基礎的問題가 미해결로 남아 있으나 그 한편에서 명확하게 응용을 指向한 시스템도 나타나기 시작하고 있다. 여기 당분간은 基礎硏究와 應用硏究의 兩者가 서로 影響하면서 병행하여 발전해 나가리라고 생각된다.

여기서 人工知能用言語에 대하여 간단히 그槪觀을 살피고 끝을 맺으려고 한다.

3 人工知能用言語

3.1 槪 要

FORTRAN, C 등의 數値計算에 적격한 언어와는 달리 LISP, Prolog 등의, 非數値데이터를 취급하는데 편리한 計算機言語群이 존재한다. 이들의 언어를 일괄하여 人工知能用言語라 한다. 表3에 그중의 주된 것을 나타낸다.

表 3. 主な人工知能用言語

言　語	開　發　者	開發年
IPL	Newell, Shaw, Simon	1957
LISP	McCarthy	1958
PLANNER	Hewitt	1971
CONNIVER	Suasman, McDermott	1972
SAIL	Feldman	1972
FUZZY	Le Faivre	1977
Prolog	Colmerauer	1972
FOL	Weyrauch	1979

記號處理에 적합한 프로그램언어의 최초의 것은 1950년대 후반에 나온 IPL로서 이것은 리스트 처리의 사고를 최초로 받아들인 언어이다. LISP는 1958年에 막카시(McCarthy)에 의해 만들어진 언어이며 리스트처리 등의 기본적인 구성을 IPL에서 이어받고 있다. 汎用의 언어이지만 복잡한 형식의 데이터를 리스트형식에서 통일적으로 취급하여 처리의 기술이 용이한 점에서 人工知能用의 主力言語가 되고 있어 그 평가는 현재도 보장되고 있다.

한편 LISP를 기반으로 하여 知識베이스에 의한 推論機構를 부가한 問題解次을 위한 언어로서 PLANNER, CONNIVER등이 개발되었다.

또, 1970년대가 되어 1階述語論理를 근간으로 한 Prolog가 歐洲에서 개발되었다. 이것은 프로그램언어라고 하는 것보다 述語論理를 이용한 推論메카니즘 그 자체이다. 근년 歐洲를 중심으로 비교적 잘 사용되어 있으나 現段階에서 LISP에 대신할 지위를 점하는데는 이르지 못하고 있다. 人工知能用言語 중에서 LISP, Prolog의 두가지 예를 들어 간단히 해설한다.

3.2 LISP

가) S式

LISP에서 한단어에 상당한 데이터를 아톰이라 한다. 가령 다음의 것은 아톰이다.

AMERICAN
"TOP"
40

이것들은 차례로 文字아톰, 文字列아톰, 數値아톰이라 한다.

리스트는 아톰 또는 리스트를 一次元的으로 늘어 놓은 것이다. 가령 다음의 것들은 리스트이다.

(SHE SELLS)
(SEASHELLS)
(ON (THE SEASHORE))

아톰과 리스트를 총칭하여 S式(S－expression)이라 한다. 文字아톰은 그 값으로서 S式을 한개 가질 수가 있다. 또 文字아톰은 函數(後述)로서 定義할 수가 있다.

나) S式의 평가

프로그램의 실제의 동작은 시스템이 S式을평가(evaluate)하므로서 일어난다.

아톰을 평가한다고 하는 것은 즉 아톰의 값을 꺼내는 것이다. 아톰이 값을 갖지 않는 경우는 誤謬가 생긴다. 또 리스트를 평가한다는 것은 그 리스트의 최초의 要素를 函數로 간주하고 남은 要素를 引數(函數로 넘어가는値)로 취급하여 函數의 値를 구할 수가 있다. 最初의 引數가 函數로서 定義되어 있는 文字아톰이 아닐 경우는 誤謬가 된다.

다) 기본적인 函數

S式의 조작이나 數値演算을 위한 기본적인 函數는 시스템의 組込函數로서 갖고 있다. 다음에 組込函數의 예를 나타낸다→의 左邊은 평가되는 리스트를, 右邊은 평가의 결과를 나타낸다.

(CAR'(PETER PIPER PICKS))→PETER

CAR는 리스트 최초의 要素를 値로 하여 되돌리는 函數이다. 1은 QUOTE라고 하며 그것에 계속되는 S式을 평가하지 않는다는 記號이다.

(CDR'(PETER PIPER PICKS))→
(PIPER PICKS)

CDR는 리스트 최초의 要素를 取去한 나머지를 되돌린다.

(CONS'(PEE'(KA BOO))→(PEE KA BOO)

(APPEND'(AB)'(CD))→(ABCD)
(LIST'A'B'C)→(ABC)

CONS는 CAR, CDR와는 逆으로, 第2引數의 최초의 要素로서 第1引數를 付加한다. APPEND는 두개의 리스트를 이어 합쳐서 하나의 리스트를 만든다. LIST는 引數를 늘어 세워 리스트를 만든다. 이들 5種類의 函數는 리스트를 처리하는데 있어 가장 기본적인 函數이다.

(SETQ A 45)→45
A→45

SETQ는 아톰에 값을 주는 函數이다. 函數의 値는 항상 第2引數의 値이고 아톰에의 값의 設定은 函數의 「副作用」으로서 일어난다.

(+ 5 4)→9
(− 9 5)→4
(SQRT(+(* 3 3)(* 4 4)))→5

數値演算을 위한 函數도 준비되어 있다.

(EQ'ORANGE'ORANGE)→T
(EQ'ORANGE'APPLE)→NIL
(SETQ SATURN NIL)→NIL
(NULL SATURN)→T

EQ는 두개의 引數가 같은 아톰인지 아닌지를 조사한 函數이며 같으면T(眞), 그렇지 않으면 NIL(像)를 돌려놓는다. 또 NULL은 引數가 NIL이면 T를 돌리는 函數이다. 이와 같은 眞인지 像를 값(値)으로 갖는 函數를 述語라 한다.

(SETQ'A− 6)→− 6
(COND((ZEROP A 0)(PRINT "ZERO")
(T(PRINT "NONZERO")))
→NONZERO

COND는 條件判定을 행하기 위한 函數이며 다음과 같은 형식을 갖는다.

(COND(條件 1 處理 1)
(條件 2 處理 2)
⋮
(條件n 處理n))

각 條件은 위에서 순서로 Test되어 최초로 NIL가 아닌 것이 있으면 그것에 이어지는 처리의 내용이 函數COND의 値가 된다. 그 以下의 조건은 Test되지 않는다.

ㄹ) 函數의 定義

LISP의 Program은 函數의 定義로부터 이루어진다.

函數는 前述과 같은 기본적인 函數를 조합하므로서 行해진다. 가령 어느 LIST의 要素를 역순으로 한 리스트를 돌려주는 函數 REVERSE의 定義는 다음과 같다. (DEFUN은 函數定義를 行하는 函數이다. 定義를 위한 函數名과 사용법은 system에 의해서 다르다)

(DEFUN REVERSE(L)
(COND((NULL L)NIL)
(T(APPEND(REVERSE(CDR L))
(LIST(CAR L))))))
→REVERSE
(REVERSE'(A B C))→(C B A)

L는 函數 REVERSE에 넘겨지는 値를 나타내는 假定의 記號(假引數)이다. 이 定義는 다음과 같이 설명으로 置換할 수 있다.

만일 건네준 리스트L가 NIL이면 NIL를 돌려주고 그렇지 않으면 L의 CDR를 REVERSE한 리스트의 마지막에 L의 CAR를 付加한 리스트를 돌려라.

函數 REVERSE는 定義中에서 自己自身을 呼出하여 사용하고 있다. 이와 같은 定義를 再歸定義라 하면 LISP의 중요한 특징의 하나이이다.

再歸定義를 사용하므로서 복잡한 處理手續을 比較的 容易하게 記述한다고 하는 이점이 생긴다. 그러나 단순한 반복보다도 memory를 많이 소비한다고 하는 결점이다.

ㅁ) 屬性리스트

LISP에서는 文字아톰에 대하여 屬性리스트라 불리는 리스트를 結付할 수가 가능하다. 屬性리스트는 다음과 같은 形式으로 表現可能하다.(리스트의 구체적인 구조는 system에 의하여 다르다)

對象　屬性 1　値 1
　　屬性 2　値 2
　　⋮
　　屬性n　値n

屬性리스트는 어느 대상에 대한 여러가지 성질을 그 대상으로 결합한 形으로 기억시켜 둘 수 있어 편리하다. 가령 酸素에 대한 정보를 屬性리스트로서 일괄하여 표기하면 다음과 같다.

酸素　原子番號　8
　　元素記號　O
　　原子量　15.999
　　族　6B

屬性리스트에의 屬性一値의 付加나 삭제 혹은 屬性리스트의 値를 檢索하기 위해서는 다음과 같은 專用의 函數가 사용된다.

(PUTRPOP　對象　値　屬性)
(PUTPROP　'酸素　8　'原子番號)

PUTPROP는 대상의 屬性리스트에 屬性一値를 추가한다. 그 屬性이 記히 존재한다는 경우에는 値를 書換하고 존재하지 않는 경우에는 屬性을 새로히 두어 값을 記入한다.

(GET　對象　屬性)→値
(GET　'酸素　'元素記號)→O

GET는 對象一屬性의 組에서 屬性리스트를 檢索하여 값을 돌려주는 函數이다.

(REMPROP　對象　屬性)
(REMPROP　'酸素　'原子量)

REMPROP는 屬性리스트에서 屬性一値를 제거하는 函數이다.

LISP의 特徵을 整理하면 이하와 같이 된다.

ㄱ) 리스트가 유일한 bata形式이다. data도 program도 리스트의 形式에서 통일적으로 表現된다.

ㄴ) program이 函數定義에 의하여 쓰여지며 函數가 program의 單位로서 다룬다.

ㄷ) 再歸的인 函數定義가 가능하다. LISP에는 그림13과 같이 많은 種類의 「方言」이 있어 言語仕樣은 統一되어 있지 않다. 현재 人工知能用으로서 가장 잘 사용되어 있는 것은 MIT系의 MAC LISP와 XEROX系의 INTER LISP이다. 어느 것이나 상당히 큰 LISP system으로 잘정비된 開發環境을 갖는다.

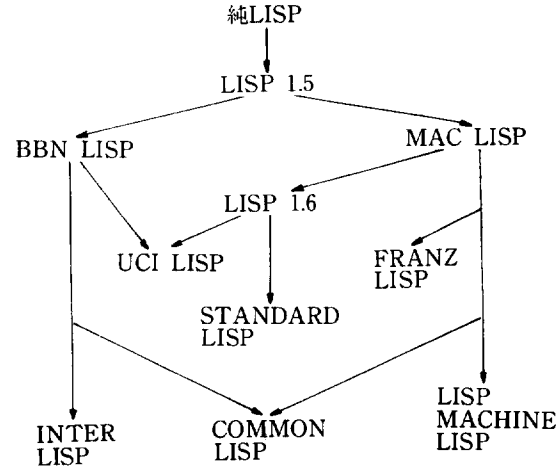


그림13. LISP方言

3.3 Prolog

(1) 項

Prolog란 programming in logic의 略으로 그 名과 같이 1階의 述語論理를 기본으로 하고 있다. 述語論理에 대해서는 3·2절에서 記述하였다.

Prolog program의 最小單位는 項이라 불리워지는 것이다. 項은 다음의 3種類內의 어느 것이다.

㉮ 아톰 : 定數 또는 述語名을 나타낸다. 다음 것은 아톰이다.

john　심 볼
"eat"　文字列
1984　數 値

㉯ 變數 : 아톰, 變數, 述語를 代入하기 위한 것 本章에서는 變數를 大文字의 英字로 나타낸다.

㉰ 述語 : Prolog의 知識表現의 單位이며 다음

形式이다.

述語(引數1, …, 引數n)

다음것은 述語이다.

fater(john, Peter)
*(10, X)

(2) 文

Prolog의 program의 各行은 文이라 불리며 다음과 같은 구성으로 되어 있다.

A : −C1, C2, …, Cn,

A, C1, …, Cn는 述語이다. 이 文은 「述語 C1, …, Cn가 모두 眞이면 述語A가 眞이다」라는 意味를 表現하고 있다. 「,」은 連言(AND)을 나타낸다.

Prolog에는 前提部의 結合에 AND以外의 것을 허용치 않아 이 表記로도 曖昧한 것은 생기지 않는다. 이 形式의 文을 節의 形式으로 變換하면 다음과 같이 된다.

$$\sim C1 \nu \cdots \nu \sim Cn \nu A$$

이 節은 正의(否定을 수반하지 않음) 리테랄을 기껏해야 1개밖에 포함하지 않는다. 이와 같은 節을 혼節(Horn節)이라 한다. Prolog는 Horn節엠 제약된 節集合을 이용한 導出機構라고 할 수 있다.

전술한 文의 基本形에서 다음 2種類의 形式이 派生하고 있다. 第2의 形式은

A

라고 하여 述語A가 眞이라는 것을 주장한다. 第3의 形式은

? −C1, …, Cn.

라 하는 것으로 述語C1, …, Cn가 모두 眞이되는 解의 探索을 行한다. 이것은 system에의 질문이고 Prolog의 program은 이 形式의 文에 의해서 起動된다.

(3) Program의 實行

Program實行의 各 step는 program中의 各文과 入力된 質問文과의 Pattern matching이다.

質問文이 入力되면 System은 Program의 先頭文에서 順次로 質問文과의 맷칭을 시도한다. 맷칭이 失敗하면 System은 program을 거슬러 올라가 失敗하기 直前에 行한 맷칭을 取消하여 맷칭을 續行한다. 質問文의 모든 述語가 眞이 되게하는 解가 발견되거나 모든 가능성을 探索을 마치면 System은 實行을 中止한다. 探索을 위한 戰略은 Program中의 文과 캇팅 오퍼레이터라 불리는 記號에 의해서 制御된다.

캇팅오페레이터는 백트랙을 오페레이터의 位置에서 저지하므로 探索은 그 시점에서 打切되어 맷칭은 실패한다.

Prolog의 Program例를 이하 나타낸다. 이것은 그림14의 家族의 系譜를 文 形式으로 한 것이다.

```
1 : father(john2, john).
2 : mother(john2, mery).
3 : father(john, peter).
4 : mother(john, susan).
5 : father(mery, mike).
6 : mother(mery, jane).
7 : grand-father(X, Y) : -father(X, Z)
                          , father(Z, Y).
8 : grand-father(X, Y) : -mother(X, Z)
                          , father(Z, Y).
```

各行先頭의 數字는 참조하기 위한 行番號이고 Program과는 無關係하다.

Program中 1～6行은 그림의 系圖에 관한 사실을 記述한 것이다. 7～8行은 述語 father, mother을 이용한 述語grand-father의 定義이

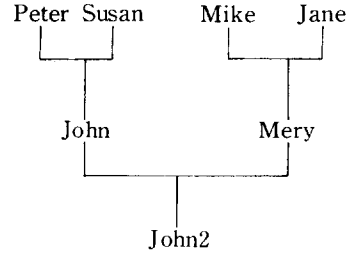


그림14. 어느 家族의 系譜

다. 이 Program에 文

? −grand-father(john2,X).

即 john2의 祖父는 누구인가 라는 質問을 入力하면 system은 探索後 X=peter라 回答한다. Prolog는 하나의 解가구해지면 實行을 解止하므로 또 하나의 解 X=mike을 구하는데는 質問文을

? −grand-father(X, john2), print(X), fail.

로 하면 된다. fail은 반드시 失敗하는 述語 이고 system은 하나의 X가 구해지면 이것을 인쇄하여 더우기 별도의 X를 찾는다.

최초의 解가 구해지기까지의 實行의 흐름을 쫓아보자. 質問文이 入力되면 시스템은 1行부터 차례를 일치하는지를 본다. 최초로 일치하는 것은 行7이고 여기서 x1=john2, Y1=X가 된다. 行7의 變數 X, Y는 質問文의 變數와는 별개의 것이므로 區別하기 위하여 Xn라 表記한다. 行7의 右邊 第1項은 行1과 一致하여 Z=john이 되고 右邊 第2項은 行3과 一致하여 Y2=peter가 되어 質問文의 解가 나온다.

4 結 語

人工知能컴퓨터에 관해서 특히 hardware 面에서 보면 並列處理로 大量의 data를 高速處理하려는 것으로 종래의 컴퓨터와 비교하여 큰 變化는 없으나 Software面에서는 推論機能·學習機能·知識data base等 종래의 技術과 크게 다르며 사용되는 素子도 VLSI나 次世代 素子가 될 것으로 생각된다. 지금 1990年의 時點에서 어느 만큼의 機能을 만족하는 것이 登場할 것인가에 관해서 특히 知識data base와 推論機能의 두가지점에 좁히고 더불어 知的interfase로서自然言語取扱(自動翻譯·音聲認識)과 畵像理解에 대해서 살펴본다면

知識data base로서는 連想檢索機能을 갖는 專門分野에서의 知識data base化가 豫想되며 用途로서는 文献檢索·醫療·教育·各種案內등이 이용될 것이고 推論機能은 並列推論·메타推論 推論머신으로 知識情報處理로 行한다든가 知的CAD의 登場이 있을 것이다. 知的interface(自然言語·畵像理解) 自動翻譯에 대해서는 技術英文과 같은 言語構造의 어느 정도까지는 一次翻譯이 가능하여 完全自動翻譯도 기대할 수 있다. 音聲認識에 대해서도 이것도 대상이나 분야를 제한하면 상당한 정도까지 가능할 것이다.

그리하여 1990年代의 中半까지는 어떠한 形態이든 人工知能컴퓨터가 실현될 것이며 人間에게도 막대한 변화를 가져다 줄 것이다.

參 考 文 献

1) A. Barr and E. A. Feigenbaum : 人工知能ハンドブックⅠ, Ⅱ, 共立出版 (1983)
2) 市川 : 制御がらみた知識工学, 計測と制御, Vol. 22, No. 9, pp. 749~754 (1983)
3) 訴訪 : エキスパートシステム開発の現狀と課題, 計測と制御, Vol. 22, No. 9, pp. 755~760 (1983)
4) 平井 : 知識バースシテム用ツール, 計測と制御, Vol. 22, No. 9, pp. 761~766 (1983)
5) 石塚 : エキスパートテムシス開発のツール, 日経ユンピユータ, (1983 9.19)
6) The LOOPS Manual (Preliminary Version), XEROX (1983)
7) Rule-Oriented Programming in LOOPS (Preliminary Version), XEROX (1983)
8) KEE User's Manual, IntelliGenetics (1984)
9) KEE Training Manual, IntelliGenetics (1984)
10) マニユニル・エバンチユク : 知識バース・システムと自然言語処理システムの商品化が米国で活発に, 日経エレクトロニクス, (1984 4.9)
11) 米田信夫, 他 : 岩波講座・情報科学 9『プロクラム言語』, 岩波書店 (1983)
12) 白井, 辻井 : 岩波講座・情報科学22『人工知能』, 岩波書店 (1982)
13) R. Schank, R. Abelson : Scripts Plans Goals and Understanding, Lawrence Erlbaum Assoc. (1977)
14) R. Schank : Conceptual Information Processing, North-Holland (1975)
15) N. J. Nilsson : Learning Machines : Foundation of Trainable Pattern-Classifying Systems,

McGraw-Hill (1965)
16) M. Shimura Learning procedures in pattern classifiers-introduction and survey, Proc. 4th Int. Joint Conf. Pattern Recognition, pp. 125～138 (1978)
17) N. J. Nilsson Artificial Intelligence, Information Processing 745 pp. 778～801, North-Holland (1974)
18) N. J. Nilsson : Problem Solving Method in Artificial Intelligence, McGraw-Hill (1971) (合田・増田訳：人工知能, 産業図書 (1972))
19) P. H. Winston : Artificial Intelligence, Addison-Welsley, (1977)
20) D. Waltz : Understanding Line Drawings of Scenes with Shadow, 文献6, 2章, pp. 19～92
21) D. G. Bobrow and A. Collins : Representation and Understanding, Academic Press (1975)
22) 長尾・辻井：知識の記述とデータベース, 計測と制御, Vol. 18, No. 1, pp. 71～77 (1979)
23) G. Ernst and A. Newell : GPS : A Case Study in Generality and Problem Solving, Academic Press (1969)
24) G. Green : Theorem-Proving by Resolution and a Basis for Question-Answering Systems, in B. Meltzer and D. Michie (eds.), Machine Intelligence 4, American Elsevier, pp. 183～205 (1969)
25) R. E. Fikes and N. J. Nilsson : STRIPS : A new approach to the application of theorem proving in problem solving, Artificial Intelligence, Vol. 3, No. 3/4, pp. 251～288 (1972)
26) C. Hewitt : Proedural embedding of knowledge in planner, Proc. 2nd Int. Joint Conf. Artificial Intelligence, pp. 167～182 (1971)
27) T. Winograd : Understanding Natural Language, Academic Press (1972)
28) G. D. Sussman : A Computer Model of Skill Acquisition, American Elsver (1975)
29) M. R. Quillian : Semantic Memory, 文献5, 4章, pp, 227～270
30) M. Minsky : A Framework for Representing Knowledge 文献6, 6章, pp. 211～277
31) E. H. Shortilliff : Computer-Based Medical Consultations MYCIN, American Elsevier (1976)
32) B. G. Buchanan, G. Sutherland E. Feigenbaum : Heuristic DENDRAL : A Program for Generating Explanatory Hypotheses in Organic Chemistry, in B. Meltzer and D. Michie (eds.) : Machine Intelligence 4, American Elsevier (1969)
33) G. J. Sussman and R. M. Stallman : Heuristic techniques in computer aided circuit analysis, I EEE Trans. Circuit & Systems, CAS 22, No. 11, pp. 857～865 (1975)
34) E. A. Feigenbaum : The art of artifcial intelligence I : themes and case studies of knowledge engineering, Proc. 6th Int. Joint Conf. Artificial Intelligence, pp. 1014～1029 (1975)