Expert System의 개요

梁 在 宇
韓國電氣通信研究所 研究員

Ⅰ. 序 論

과학자들은 오래 전부터 인간 수준의 지능을 갖는 기계를 만들려는 노력을 지속해 왔다. 비록 성공하지 못하였더라도 Babbage가 계산기를 설계한 것도 생각하는 기계를 만드는데 목적이 있었다. 컴퓨터가 발명된 이후에 지능을 기계에 실현하려는 노력은 가속되었다. 1947년 Turing은 지능을 갖는 컴퓨터인가를 시험하는 방법으로 'Turing Test'를 제안하였고, 1956년 몇 명의 과학자들이 모여 이 분야를 인공지능(Artifical Intelligence)이라고 명명할 때 최초의 expert system이라고 할 수 있는 '논리학자(Logical Theorist) 프로그램이 선보였다. 이 프로그램은 Whitehead와 Russell이 저술한 Principia Mathematica에 수록된 일부 공식들을 증명해 내었는데 그 중 어떤 것은 수학자보다 더나은 방법으로 증명을 도출하였나 컴퓨터가 증명하였다는 이유로 당시 학술지에 게재되지 못하였다.[1]

그 이후 학자들은 사고의 기본 원칙을 발견하면 고속 컴퓨터에 고도의 지능을 갖는 시스템을 구현할 수 있을 것으로 생각하고 범용문제 해결방법(general purpose problem solving method) 개발에 몰두하였다. 그 결과 얻어진 general problem solver등의 범용문제 해결방법은 인간의 지적 능력이 잘 발휘되는 복잡한 문제 해결을 감당하기에는 미흡한 것으로 드러났다. 60년대 중반이후 국한된 전문 분야의 문제해결에 노력을 집중시켜 70년대에 의료진단, 분자구조분석 등등의 분야에 여러가지 expert system을 개발해 냈다. 이 시스템들이 괄목할만한 성능을 나타낸 데에 힘입어 인공지능학자들은 범용지식표현 방법을 연구하였다. 이러한 시도는 초기의 범용문제 해결방법을 구하는데 겪는 어려움을 또 다시 반복하기는 하였으나, 여러가지 expert system tool을 개발해 내었고, expert system의 성능은 그 시스템이 사용하는 문제해결 도식(problem solving formalism) 보다는 시스템에 저장된 지식에 의존한다는 것을 알았다. 80년대에 이르러 expert system은 각 계의 주목을 끌어 실용화 및 상품화연구도 확산되고 있다.

Ⅱ. Expert System과 지식공학

Expert system이란 전문가의 지적 능력을 발휘하는 시스템이다. 전문가들은 그들이 보유한 경험과 지식에 의해 일반인과는 달리 특수 분야에 뛰어난 능력을 발휘한다. 그런데 전문가는 그 수효가 적고, 양성하는데 많은 시간과 비용이 소요되므로 전문가를 이용는 데는 여러가지 제약이 따른다. 그러므로 expert system 개발은 전문가의 지식을 값싸게 널리 보급시키는 이득을 가져다 준다. 전문가의 지식은 복잡하여 기존의 프로그래밍 방법으로는 전산화하기 힘들다. expert system은 일반프로그램이 데이타를 프로그램과 분리하여 기술하듯이 지식을 분리하여 표시하는 기술을 사용함으로써 문제해결 이외에 해답추출의 근거 및 당위성을 설명하는 기능 및 새로운 지식을 추가하는 기능을 가지고 있다.

그러나 전문지식을 프로그램하는 과정은 일반화된 기법이 정립되어 있지 않다. 따라서 이는 프로그래머의 기량에 크게 의존하게 되며 이에 대한 연구를 지식공학(knowledge engineering)이라 한다. 지식을 프로그래밍하기 어려운 이유는 대부분의 지식이 수학 공식과는 달리 부정확하고 불완전한 속성을 가지고 있기 때문이다. 지식공학은 지식의 도식화를 꾀함으로써, expert system의 개발은 물론 지식의 본질을 연구하는 데에도 많은 도움을 준다. expert system 분야가 다른 인공지능 분야와 다른 점은 특정 분야의 전문가의 능력을 자동화하는데 주력하는 것과 문제해결 방법 및 과정을 스스로 설명하는 기능을 채택한 점이다. 현재 expert system에 의해 자동화할 수 있는 분야는 인식

에 관련된 지식에 한하며 비교적 손쉽게 적용 가능한 분야는 표 1과 같다.[2]

표 1. 지식공학의 적용 분야

Category	Problem Addressed
Interpretation	Inferring situation descriptions from sensor data
Prediction	Inferring likely consequences of given situations
Diagnosis	Inferring system malfunctions from observables
Design	Configuring objects under constraints
Planning	Designigning actions
Monitoring	Comparing observations to plan vulnerabilities
Debugging	Prescribing remedies for malfunctions
Repair	Executing a plan to administer a prescribed remedy
Instruction	Diagnosing, debugging, and repairing student behavior
Control	Interpreting, predicting, repairing, and monitoring system behaviors

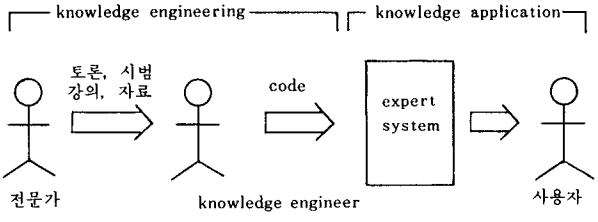

그림 1 지식의 흐름

Ⅲ. 지식의 표현 및 추론

효과적으로 모든 분야의 지식에 적용될 수 있는 일반적인 지식표현 및 추론 방법은 발견되지 않았다. 따라서 expert system도 여러가지 방법 및 형태로 제작된다. 여기서는 간단한 forward-chaining rule-based system의 예를 통해 expert system의 동작을 살펴보기로 한다.[4]

Rule-based system에서는 if~, then~형태로 표시되는 rule을 지식표현 방법으로 사용한다. rule은 인간 사고 형태의 한 유형으로 볼 수 있으며 다음과 같은 여러가지 의미를 가지고 있다.

- Situation/Action
  If temperature≧300℃ then turn off boiler
- Premise/Conclusion
  If Stain is grampos then organism is strep
- Antecedent/Consquent
  If X is a dog then X is an animal

Rules
Rule 1 : if stain is grampos then organism is strep
Rule 2 : if stain is grammeg then organism is e. coli
Rule 3 : if orgranism is strep or organism is bacteriods then penicilin is indicated
Rule 4 : if ? drug is indicated and not (allergic to ? drug) then prescribe ? drug

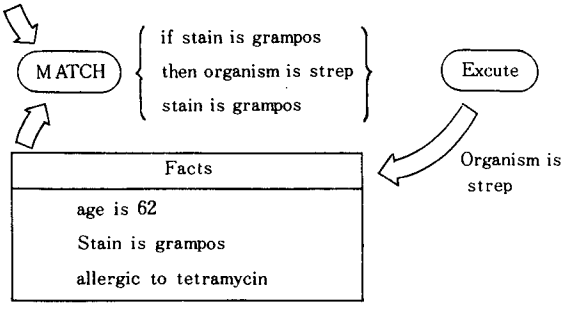

그림 2. Forward-chaining Rule-based system의 예

그림 2는 rule 및 추론형태를 보여 주고 있다. rule 내의 밑줄친 단어들은 reserved word를 표시하며, 물음표로 시작되는 단어는 변수(variable)을 나타낸다. 이러한 rule과 환자에 관한 자료인 'Facts'를 통해서 약의 처방에 대한 정보를 얻어 낸다. 먼저, fact와 각 rule의 if절과 비교하여 match되는 rule은 then절 수행시켜서 새로운 fact를 얻는다. 그림에서 먼저 'organism is strep'을 새로운 fact로 포함시키게 되고, 두번째의 match-execute cycle을 거치면 'penicilin is indicated'를 얻어낸다. 그리고 최종 cycle에서는 'prescribe penicilin'라는 결론을 도출한다. 이와 같은 추론방법을 forward chaining이라고 한다.

이상의 진행에서 추론과정을 기록해 두면, 다음과 같은 질의 응답을 가능하게 할 수가 있다.

Q : Why did you prescribe penicillin ?
A : Because the stain is grampos (indicating strep) and the patient is not allergic to penincillin.

이상의 예는 간단하여 일반 프로그램 기법으로도 손쉽게 전산화되지만, 좀 더 깊이 살펴 보면 기본적으로 다음과 같은 capacity의 차이가 있다. 즉 지식이 세부 mechanism과 분리되어 뚜렷 표현되어 있고 새로운 rule의 추가가 용이하며 복잡한 rule도 손쉽게 표현 수 있는 것등이다.

Expert system 제작에 사용되는 다른 중요한 지식 표현 기법으로는 semantic network, frame등이 있다. semantic network은 다음과 같이(그림 3) node와

link로써 표시되는데 대상 시스템의 세부구조 등을 정확하게 표시하는데 편리한 잇점이 있는 반면에 너무 일반적이어서 지식표현 및 추론이 복잡해질 우려가 있다. frame은 복잡한 상황을 나타내는데 많이 이용되고 있다. frame은 여러개의 slot을 포함하는데 각 slot에는 필요한 값 또는 값을 산출하는데 필요한 procedure가 연결된다.

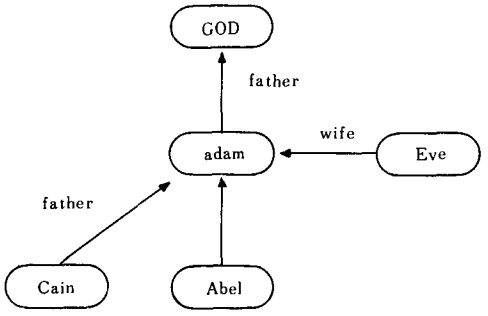

그림 3. Semantic network의 예

Frame : Meeting-for-weetplanning
a-kind-of : Regular-meeting
place : default : Room-364
if added : if a room is available for time then reserve room for time
time : 10 o'clock monday
if-removed : notify attendants
attendants : Kim, Yun, Yong

Ⅳ. Expert System의 예

다음으로 expert system의 예를 살펴보기로 하자. 진단분야에 사용되는 expert system의 특성을 잘 나타내 주는 예로는 MYCIN이 있다. MYCIN은 70년대에 Stanford 대학에서 개발된 것으로 전염병을 진단 처방하는 시스템이다. MYCIN은 다음과 같은 rule을 써서 지식을 표현한다.[3]

Rule 27
If 1) the gram stain of the organism is gram negative, and
2) the morphology of the organism is rod, and
3) the aerobicity of the organism is anaerobic,
Then There is suggestive evidence (7) that the identity of the organism is Bacteroides.

MYCIN은 이와 같은 rule을 400내지 500개를 보유하고 있다. 불확실한 지식을 표현하는 방법으로 certainty factor (위의 경우 0.7)을 사용하며, 추론방법은 backward-chaining을 이용한다. 다음은 MYCIN 프로그램의 사용 예를 나타낸 것으로 사용자(의사)의 입력은 별표 2개(**) 이후에 나타나 있다.

---------- PATIENT-248 ----------
1) Patient's name : (first-last)
**Dick H.
2) Sex :
**MALE
3) Age :
**52
4) Have you been able to obtain positive microbiological information about a possible infection of Dick H. ?
**YES
---------- INFECTION-1 ----------
5) What is the infection ?
**ENDARTERITIS
⋮
The first significant organism from this blood culture (CULTURE-1) will be refereed to as :
---------- ORGANISM-1 ----------
9) Enter the identity of ORGANISM-1 :
**Unknown
10) Is ORGANISM-1 a rod or coccus (etc.) :
**ROD
11) The gram stain of ORGANISM-1 :
**GRAMNEG
12) Have there been POSITIVE cultures yielding organisms about which you will NOT be seeking advice ?
**YES

이러한 방식으로 질의 응답을 통해서 병의 원인을 추적해 나간다. 사용자는 그 답을 모를 경우 9번의 물음의 예와 같이 답할 수 있는데 이때에는 MYCIN은 가지고 있는 지식을 최대한 이용하여 이를 규명하려 한다. 위와 같은 질의응답을 30회내지 90회를 거쳐 다음과 같은 진단결과를 얻는다.

INFECTION-1 is ENDARTERITIS with BACTEREMIA
〈Item 1〉 E. COLI [ORGANISM-1]

〈Item 2〉 SALMONELLA(species unknown)
[ORGANISM-1]
〈Item 3〉 KLEBSIELLA-PNEUMONIAE[ORGANISM-1]
〈Item 4〉 PSEUDOMONAS-AERUGINOSA
[ORGANISM-1]
〈Item 5〉 ENTEROBACTER[ORGANISM-1]
〈Item 6〉 PROTEUS-NON-MIRABILIS
[ORGANISM-1]

이는 병의 원인이 불투명한 예이며, 현재의 의학기술로는 확실한 진단결과를 얻을 수 없는 경우가 발생한다고 한다. 사용자는 다음과 같이 진단 결과의 근거를 물어 볼 수가 있다. c.f.는 confidence factor를 뜻한다.

**HOW DID YOU DECIDE THAT ORGANISM-1 WAS AN E. COLI?
I used rule 084 to conculde that the identity of ORGANISM-1 is e. coli. This gave a cumulative c. f. of(.51).
I used rule 003 to conclude that the identity of ORGANISM-1 is e. coli. This gave a cumulative c.f. of(.43).

이상의 6가지 병의 원인에 대하여 각각 별도의 항생제를 처방할 경우 환자에게 치명적인 결과를 가져올 수 있다. MYCIN은 다음과 같이 처방을 제시한다.

[Rec 1] My preferred therapy recommendation is as follows:
In order to cover for Items〈1 3 4 5 6〉:
Give: GENTAMICIN
Dose: 128 mg(1.7 mg/kg) q8h IV(or IM) for 10 days
Comments: Modify dose in renal failure
In order to cover for Item〈2〉
Give: CHLORAMPHENICOL
Dose: 563 mg(7.5 mg/kg) q6h for 14 days
Comments: Monitor patient's white count
Do you wish to see the next choice therapy?
**NO

이와 같이 MYCIN은 그 최종 처방을 사용자에게 선택할 수 있도록 하며 사용자는 다른 처방을 요구할 수가 있다. 실용화에 성공한 대표적인 예로는 XCON(일명 R1)이 있다. XCON은 DEC회사에서 동사의 컴퓨터인 VAX 및 PDP-11의 configuration작업을 자동화시키는 프로그램이다. DEC사는 UNIBUS configuration작업에 많은 어려움이 있었고, 이를 기존 프로그래밍 기법으로 자동화하려고 세번 시도하였으나 실패하였다. XCON은 이의 해결을 위해 Carnegie Mellon 대학에서 개발한 OPS-5라는 그림 2와 동작원리가 유사한 rule-based 언어를 이용하여 만들어졌다. XCON은 컴퓨터 사용자의 구매요구서로부터 적절한 configuration을 얻어 낸다. 다음은 VAX-780 configuration 작업의 예이다. 예에서 구매요구에는 누락된 cable등이 추가되고(10번, 11번) 부품의 모델이 적절하게 변경된 것을 알 수가 있다. (2번→20번, 3번→21번)[4]

Components ordered

Line	Qty	Name	Description	Comment
1	1	780XA-AJ	11780 2mb 64k 240/50	
2	1	RUA80-CA	ra80-ca, uda50 ctl, 2/cab	not-configured
3	1	TEU77-FB	1600/800 tape sys 11/780 60h	not-configured
4	8	DZ11-DP	8line async mux(eia)	
5	4	DMF32-LP	multifunction 8asyn, 1syn, lpd	
6	4	DZ32-AP	vax 8 line async(eia)	
7	1	LA120-DA	la120 univ pwr supply num pa	
8	1	QE001-HM	vax/vms upd n/s 16mt9	
9	1	VS100-AA	vaxstation 100	

Components added

Line	Qty	Name	Description	Comment
10	1	BA11-KV	10 1/2inch expander box 240 needed to provide box space for unibus modules	
11	3		bco51 cable needed to connect DW780*and BA11-KV*	
20	1	RUA80-AD	ra80-ad, uda50 ctl, no cab for RUA80-CA because of frequency mismatch	
21	1	TEU77-FD	1600/800 tape sys 11/780 50h for TEU77-FB because of frequency mismatch	

그림 4. XCON의 출력 例

XCON은 이 밖에도 각 모듈의 위치를 지정해 주는 그림을 출력하며, 이러한 작업을 위해서 다음과 같은 rule을 약 2000개 보유하고 있다.

IF : The current subtask is assigning devices to unibus modules
And there is an unassigned dual port disk drive
And the type of controller it requires is known
And there are two such controllers neither of which has any devices assigned to it

And the number of devices which these controllers can support is known
THEN : Assign the disk drive to each controller
And note that each controller supports one device

V. Expert System Tool

초기에는 대부분의 expert system이 LISP등의 symbolic언어로 프로그램되었다. symbolic언어는 pascal, C등의 일반 고급언어보다는 더 높은 레벨의 언어라고 볼 수 있지만, symbolic language로 programming하는 작업은 상당한 시간을 요하는 것이다. 따라서 지식공학자들은 성공적인 expert system에서 적용분야에 관한 특정지식(domain specific knowledge)을 제거시킴으로써 expert system의 골격을 얻어내고, 이에 타분야 지식을 적용시키는 시험을 실시하였다. 이러한 과정에서 expert sxstem tool들이 개발되었다. expert system tool은 두 가지 개념으로 발전되었다. 즉 expert system을 개발하기 위한 각종 utility의 집합으로서의 개념과 보다 높은 레벨의 프로그래밍 언어로서의 개념이다. 전자의 예로서는 EMYCIN을 들 수가 있고, 후자의 예로 OPS-5를 들 수 있다. EMYCIN은 MYCIN에서 혈액 전염성 질환에 관한 진단과 처방에 관한 지식을 제거시킨 것이다. EMYCIN의 기능은 그림5와 같이 표시할 수가 있다.[5]

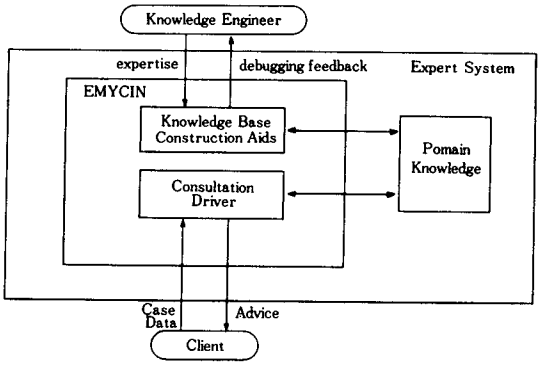

그림5. EMYCIN의 기능

EMYCIN은 MYCIN의 예와 같은 rule의 형태로 knowledge base를 구축하며, 전반적인 제어방식은 backward-chaining방식을 쓴다. rule은 자연언어와 유사한 ARL(abbreviated rule language)로 표시할 수 있으며, rule을 만들때 철자, 오류등을 지적 수정할 수가 있다. 또한 설명기능과 잘못된 rule을 탐색 및 수정하는 utility등을 보유하고 있다. EMYCIN은 진단 고장분석 분류 등의 작업에 잘 적용되는 것으로 알려졌다. OPS-5는 XCON을 만드는데 사용되었으며 순수한 rule-based 언어이다. OPS-5에서 rule은 다음과 같이 표시된다. 세미콜론 이후는 comment를 나타낸다.[6]

```
Cp find-coloured-block ; rule name find-coloured-blok
   (goal               ; If there is a gool
    ∧status active     ;          which is active
    ∧type find         ;    to find
    ∧object block      ;       a block
    ∧colour<x>)        ;          of colour⟨x⟩
   (block              ; and there is a block
    ∧colour⟨x⟩         ; of colour ⟨x⟩
    ∧name ⟨block⟩)     ; whose name is ⟨bock⟩
→                      ; Then
    (make result       ; keep the name of block
    ∧pointer ⟨block⟩) ;
    (modify 1          ; and change the goal statas
    ∧status satisfied)); to 'satisfied'
```

여기에서⟨ ⟩로 표시된 것은 변수를 나타낸다. OPS-5는 앞서 설명한 forward chaining rule-based system 이며, 우수한 pattern matching기능을 가지고 있다. 그러나 설명 및 debugging기능은 미약하다. expert system tool에는 이 밖에도 blackboard라는 global data base를 통하여 여러개의 cooperating agent가 협조하며, 각각 한 분야씩 담당함으로써 복잡한 문제를 해결하는 Hearsay-Ⅲ, 그리고 자연언어와 아주 유사한 형태로 프로그램이 가능한 ROSIE등 여러가지가 있다. 일반적으로 여러 분야에 사용될 수 있는 tool보다는 좁은 분야로 사용범위가 한정된 tool이 개발에 편리한 경우가 많다. expert system tool은 상품화되어 일부 회사에서 판매하고 있으며, expert system의 개발 대상분야가 확장되면서 더욱 많은 종류의 tool이 나올 것으로 예상된다.

Ⅵ. 전 망

Expert system에 대한 관심도가 높아지고 고성능 소형컴퓨터가 발달함으로써, 몇 년내로 여러 분야의 expert system이 출현할 것으로 예상되고 있다. 그러나 아직도 해결해야 할 문제점이 적지 않다. 실제적으로 전문가들은 다음과 같은 능력과 특성을 갖고 있는

것으로 알려져 있다.[3]

- 문제해결능력
- 문제해결과정 및 결과에 대한 설명 가능
- 새로운 지식의 추가획득
- 보유지식을 재구성하여 성능을 증대시킴
- 타 전문영역과의 관계를 알고 이를 이용
- 지식의 한계에 다달았을 때에도 성능이 급격히 저하되지 않음

이중, 네번째 이하의 항목에 대해서는 현재의 expert system 기술 영역밖이고, 앞 부분의 3개 항목도 충분히 해결되지 않았다. 일부에서는 expert system이라는 용어 자체가 일반인을 현혹시키는 부적절한 표현이라는 비판이 있으며, expert system이 아직도 실험적 단계에 있다는 것을 부정하기 힘들다. 그러나 몇가지 실용적인 expert system이 이미 사용되어 그 효용성을 입증하고 있으며, 현재로서는 이러한 기능을 실현할 수 있는 대체 방법이 없다. 그리고 expert system이 지식정보산업의 도구로서 엔지니어의 주목을 받기에 충분하고 높은 부가가치를 생산할 수 있다는 점이 expert system개발을 가속시키고 있다.

參 考 文 獻

[1] J.N. Shurkin, "Expert Systems: The practical Face of Artificial Intelligence", *Technology Review*, pp. 72-78, Nov.,1983.

[2] F. Hayes-Roth etal., *Building Expert Systems*, Addision-Wdsley, pp. 3-29, 1983.

[3] P.H. Winston et al., *The AI Business: The commerical Uses of Artificial Intelligence*, MIT Press, pp. 17-49, 1984.

[4] *Tutorial on Expert System*, AAAI, 1984.

[5] B.G. Buchanan et al., *Rule-Based Expert Systems*, Addison-Wesley, pp. 302-328, 1984.

[6] C.L. Forgy, *OPS5 Manual*, Carnegie-Mellon Univ., 1981. *

♣ 用 語 解 說 ♣

simulation과 emulation

시뮬레이터(simulator)는 어셈브리 혹은 프로그래밍 언어로써 쓰여진 소프트웨어 프로그램을 사용하여 어느 시스템과 같은 성격을 나타내는 묘사장치이다. 시뮬레이터는 모방되는 시스템과 같은 자료를 받아들여서 프로그램을 수행하여 같은 결과를 산출한다. 시뮬레이터는 대체로 묘사되는 기기보다 상당히 처리가 느리며, 물리적으로는 닮지 않는 것이 보통이다. 시뮬레이터는 묘사되는 기기에 대한 내부특성 및 동작특성에 대한 연구를 위하여 사용된다. 일반적으로 작은 마이크로프로세서등을 위한 소프트웨어의 개발 혹은 디버깅(debugging)을 목적으로, 이보다 좀 더 큰 소형컴퓨터(소프트웨어 개발 시스템등)등을 사용한다. 에뮬레이터는 시뮬레이터가 소프트웨어를 사용하여 묘사하는데 비하여 에뮬레이터는 마이크로프로그램과 특별한 하드웨어를 사용하여 요구되는 시스템을 거의 같은 속도 혹은 그 이상으로 묘사한다. 예로써 slice micro processor는 소형 컴퓨터를 보다 작은 콤포넌트와 보다 빠른 사이클 타임으로써 에뮬레이션한다. 에뮬레이터는 물리적으로 묘사되는 기기와 흡사하게 만들어진다.

heuristic

1. 어떤 계획이나 프로그램을 개발하기 위한 방법론이나 procedures으로, 기존의 procedures를 향상시켜 시간, 경비, 소요인원 및 결과의 제한적 사용을 고려할 때에 만족할만한 결과나 출력을 얻을 수 있는 것을 말한다.

2. algorithmic과 반대의 개념으로써 문제풀이에 대한 경험적, 탐구적 방법에 관한 것으로, 이때 해법은 최종 결과를 향한 진행상태의 평가에 의하여 얻어진다.