HO-06 Kecerdasan Buatan Methods of Inference (1)

Forward Chaining &Backward Chaining

Opim S Sitompul

Erna Budhiarti Nababan

Pengantar

- Struktur **IF-THEN** berkaitan dengan informasi atau fakta yang diberikan pada bagian IF dan tindakan pada bagian THEN.
- Sebuah *rule* memberikan deskripsi bagaimana menyelesaikan sebuah problem.
- *Rules* relative mudah d ibuat dan dipahami
- Setiap rule diri dari dua bagian: bagian **IF**, disebut *antecedent* (*premise* or *condition*) dan bagian **THEN** disebut *consequent* (*conclusion* or *action*).

Operator dalam Rule

- Antecedent sebuah rule menggabungkan **object (linguistic object)** dan **nilainya** yang dihubungkan oleh sebuah operator.

Contoh:

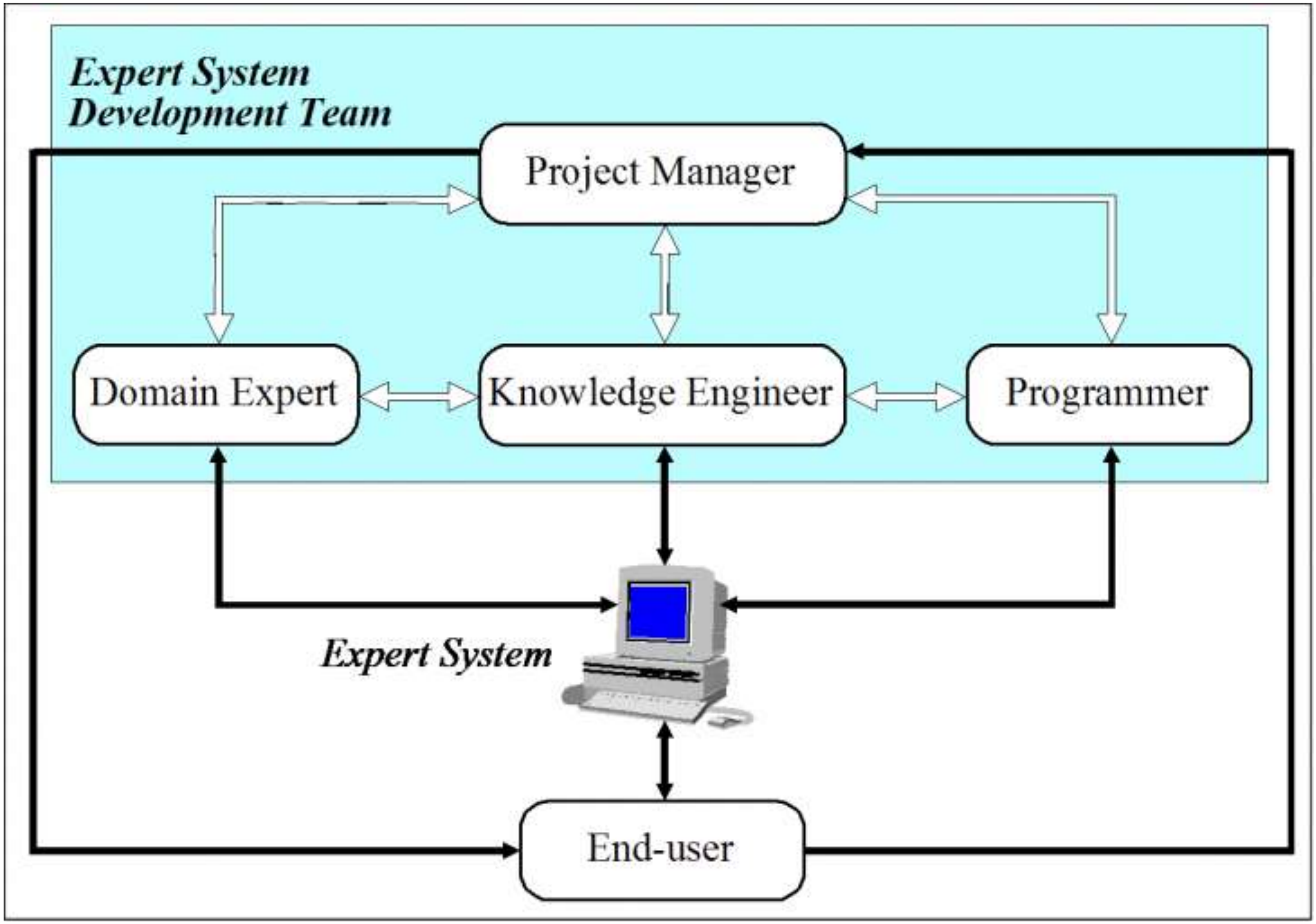
IF *bankBalance* ≥ *requestedAmount* **THEN** action is *processTransaction*

Rules dalam Expert System (ES)

Expert System terdiri dari lima komponen :

- domain expert
- knowledge engineer,
- programmer,
- project manager,
- end-user.

Keberhasilan sebuah ES tergantung dari kerjasama ke lima komponen tsb.



Gambar 2: Interaksi *team* dalam pembangunan ES (Negnevitsky, 2010)

- **Domain Expert**

The most important player in the expert system development team. Mempunyai kepakaran dalam bidang tertentu yang harus dapat ‘ditangkap’ dalam ES

- **Knowledge Engineer**

Orang yang mampu mendisain, membuat dan menguji ES.

- **Programmer**

Bertanggung jawab dlm pemrograman (Prolog)

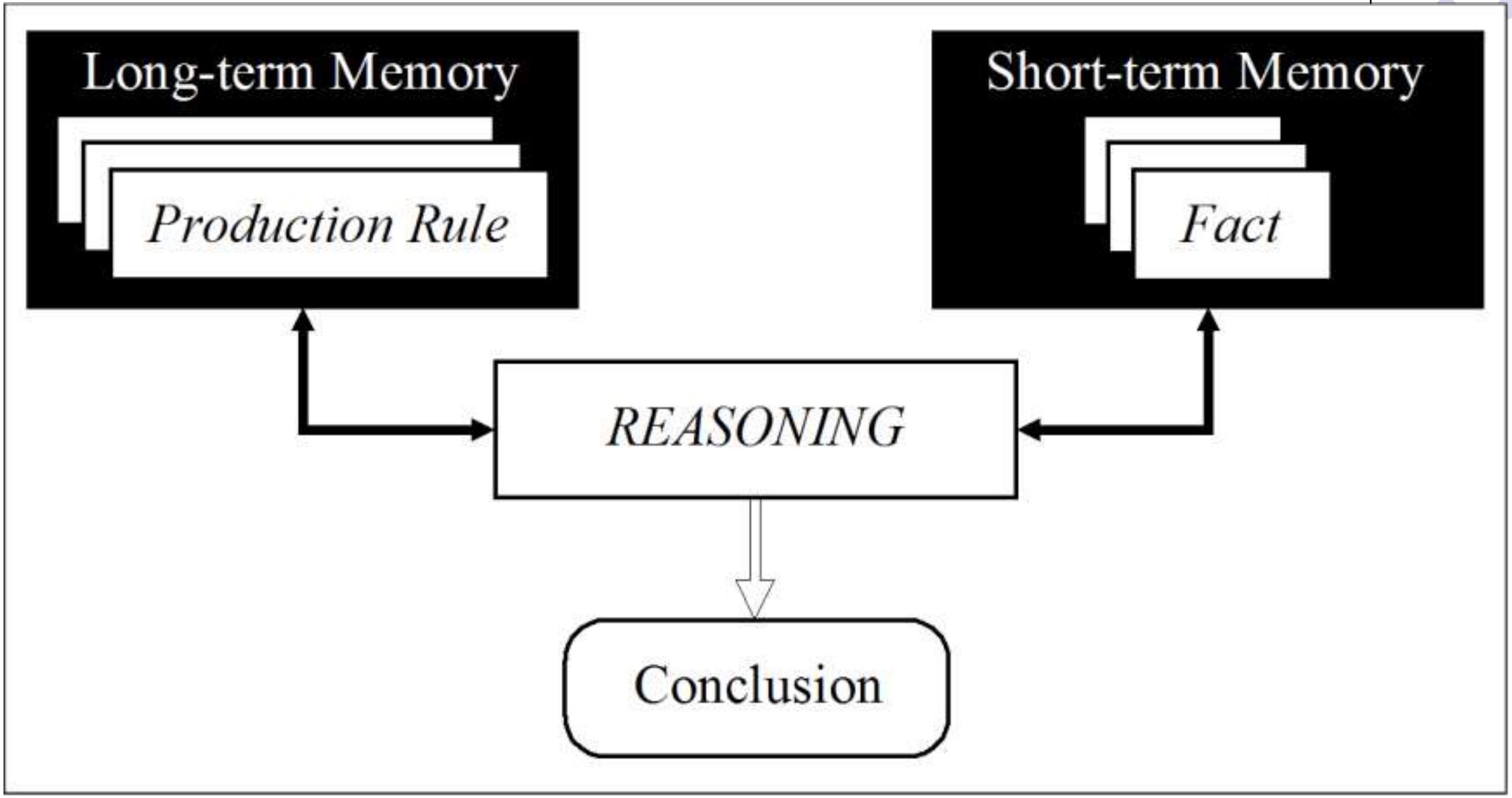
- **Project Manager**

Memastikan project berjalan sesuai dengan yang direncanakan (on track). → Pencapaian sesuai dg yang direncanakan

- End User

Pengguna ES

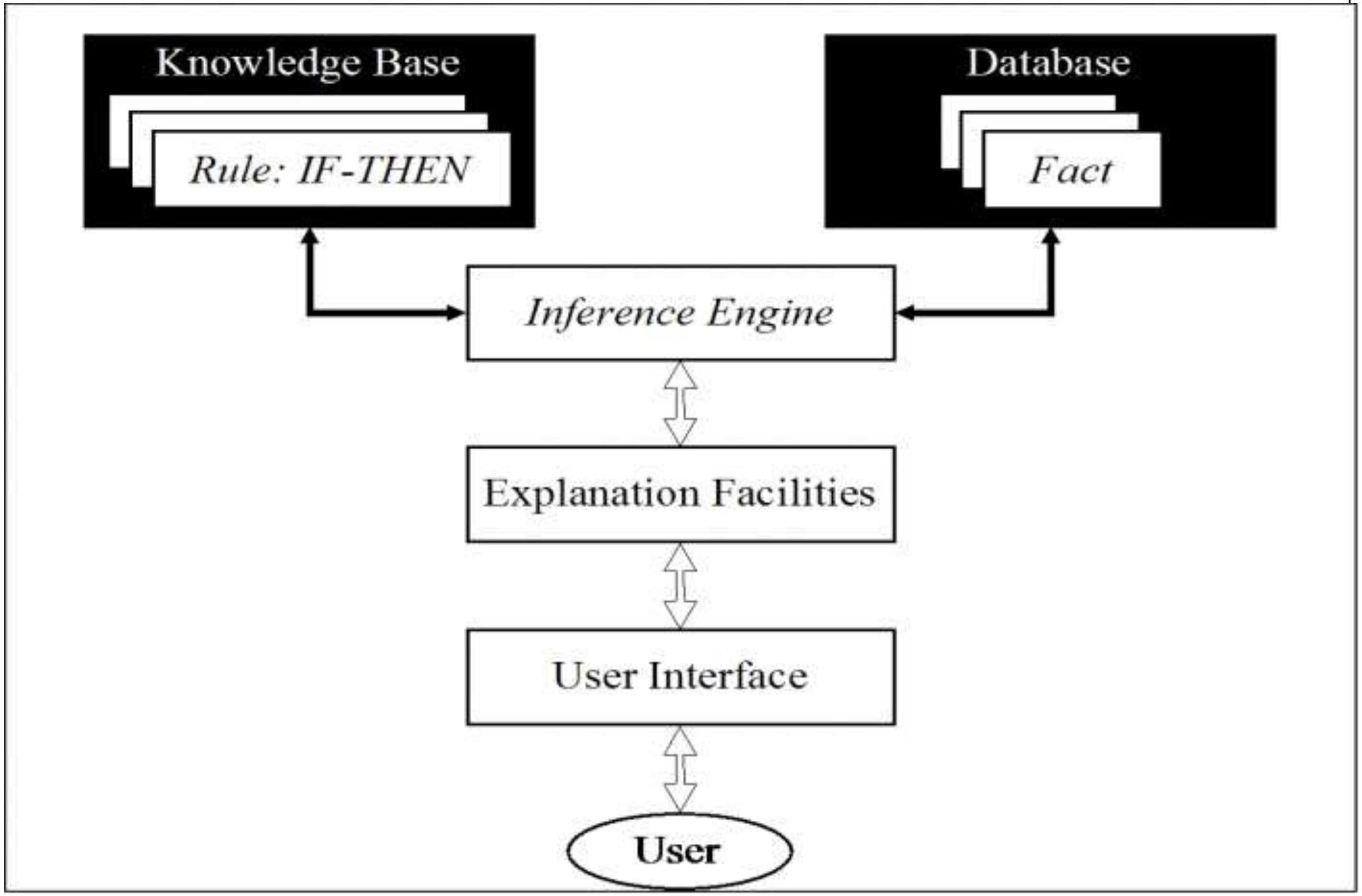
Production System Model dlm ES



Gambar 2 Model Production System

Production Rule

- Berdasarkan gagasan bahwa manusia memecahkan masalah dengan menerapkan pengetahuan mereka (diekspresikan sebagai aturan).
- Production rules disimpan dalam dalam long-term memory, dan informasi terkait *problem-specific* atau fakta disimpan dalam short-term memory.



Gambar 3 Susunan dasar rule-based ES

knowledge base berupa domain knowledge berguna untuk problem solving. Knowledge direpresentasikan dlm sekumpulan rules

database berupa sekumpulan fakta yang akan dicocokkan dengan bagian **IF** (condition) dari rule yang disimpan dalam knowledge base

inference engine menggunakan penalaran dimana ES dapat memberi solusi. Ia menghubungkan knowledge base g fakta dalam database

- ***explanation facilities*** menungkinkan user bertanya kepada ES, bagaimana kesimpulan tertentu tercapai dan mengapa fakta spesifik diperlukan.
- ***user interface*** adalah sarana komunikasi antara pengguna yang mencari solusi untuk masalah dengan ES

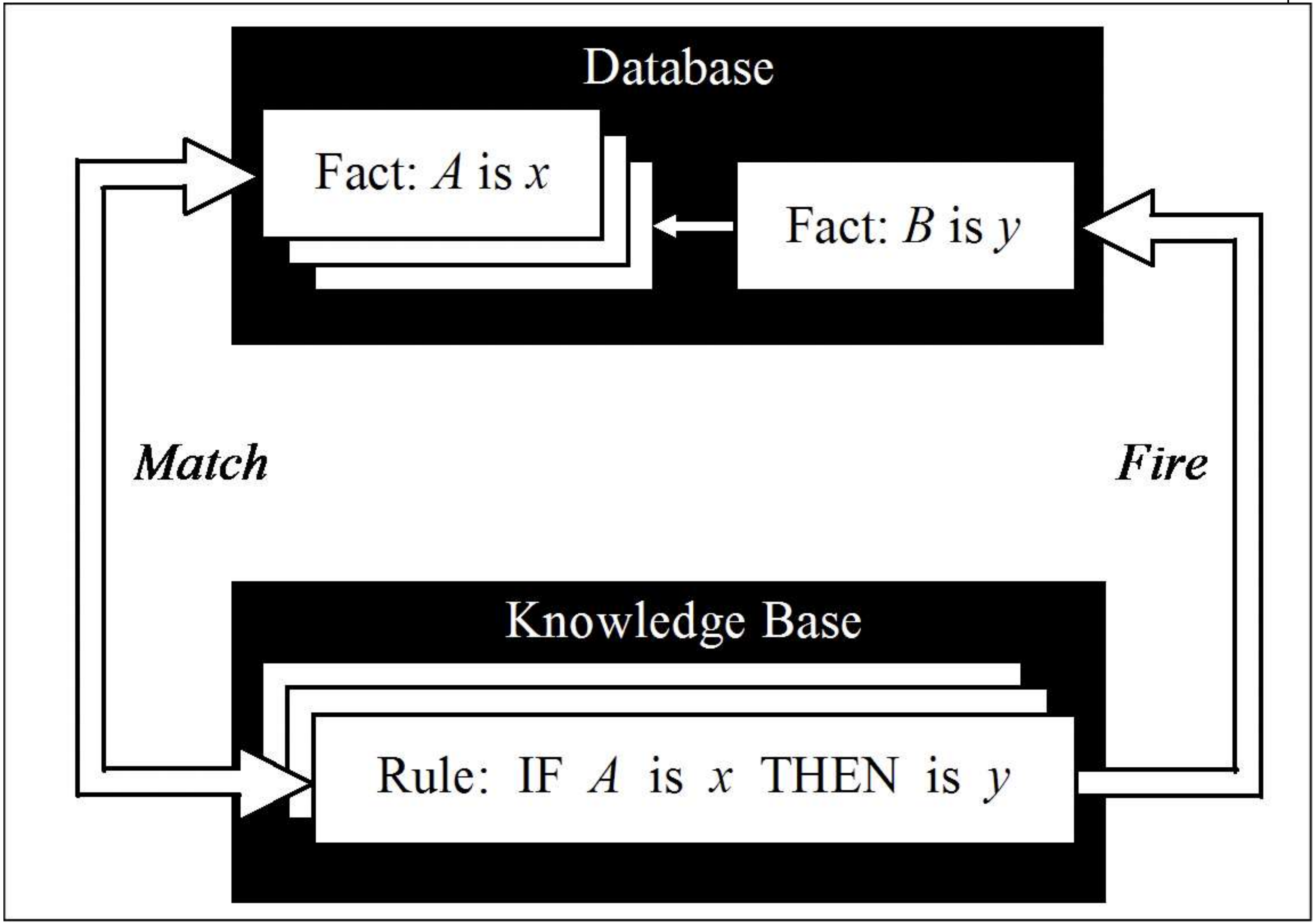
Struktur kendali (Inference)

- **Forward Chaining**

- **D**omain knowledge dinyatakan dengan sekumpulan production rule IF-THEN
- Data dinyatakan dalam sekumpulan fakta mengenai situasi terkini
- Inference engine membandingkan setiap rule yang disimpan dalam knowledge base gan fakta dalam database.

Forward chaining (cont’d)

- Jika bagian **IF** (antecedent) dari rule cocok (matches) dengan fakta, rule di **fire** dan bagian **THEN** (action) dieksekusi.
- Pencocokan antecedent dengan fakta menghasilkan rantai inferensi (inference chain)



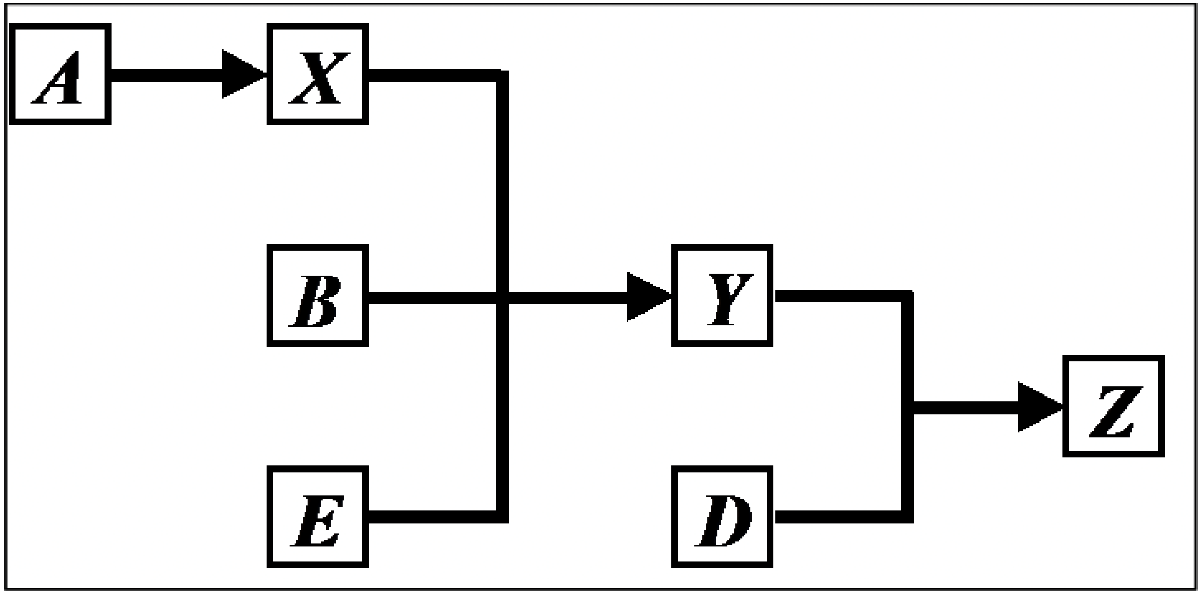
Gambar 4: Siklus inference engine

Contoh

- RULE 1:
  IFY is *true* **AND** D is *true*

  THEN Z is *true* ($Y\&D \rightarrow Z$)
- RULE 2:
  IF X is *true* **AND** B is *true* **AND** E is *true*

  THEN Y is *true* ($X\&B\&E \rightarrow Y$)
- RULE 3: **IF** A is *true* **THEN**X is *true*($A\rightarrow X$)



Gambar 5: Inference chain

- Forward chaining: data-driven reasoning. (Banyak aturan dieksekusi termasuk yang tidak ada hubungannya dengan tujuan yang ditetapkan.) Anggaplah ada aturan lain, A → Q, yang tidak terkait dengan tujuan untuk menentukan Z, itu juga akan di fire.
- Oleh karenanya jika tujuan hanya untuk menyimpulkan satu fakta tertentu, metode forward chaining menjadi kurang efektif

Summary (Forward Chaining)

- Strategi pencocokan yang dimulai dari satu set fakta yang diketahui
- mengurutkan fakta baru menggunakan *rule* yang premisnya sesuai dengan fakta yang diketahui
- proses ini berlanjut sampai
  - gol ditemui
  - Tidak ada lagi rule yang premisnya cocok dengan fakta diketahui atau fakta yang diurutkan

Forward Chaining (cont’d)

- Data Driven
- menganalisis masalah dengan melihat fakta yang sesuai/cocok dengan bagian IF dan IF-THEN rule.
- Penggunaan: untuk mengawasi dan mendiagnosis sistem pengendalian proses pada *real time* dimana data yang didapat selalu di-*up date*

Ilustrasi Fwd Chaining dlm rule based system

- Contoh:

Diberi fakta sbb:

Fakta-1 : ibu dari Budi adalah Siti

Fakta-2 : ibu dari Anna adalah Siti

Rule-1:

IF ibu_bapa dari X AND ibu_bapa dari Y adalah P

THEN X adik_beradik Y

Rule-2:

IF ibu dari X ialah P

THEN ibu_bapa dari X ialah P

Gunakan Forward Chaining

Rule-1 sesuai??

TIDAK sebab tidak ada fakta ibu_bapa

Rule-2 sesuai??

YA fakta-1 dan 2 sesuai

Contoh Lain

- **Rule 1:**

  IF Y is true.

  AND D is true

  THEN Z is true.

Rule 2:

IF X is true

AND B is true

AND E is true

THEN Y is true

- **Rule 3:**

  IF A is true

  THEN X is true

Rule 4:

IF C is true

THEN L is true

- **Rule 5:**

  IF L is true

  AND M is true

  THEN N is true

Rule 1: $Y \ \& \ D \rightarrow Z$

Rule 2: $X \ \& \ B \ \& \ E \rightarrow Y$

Rule 3: $A \rightarrow X$

Rule 4: $C \rightarrow L$

Rule 5: $L \ \& \ M \rightarrow N$

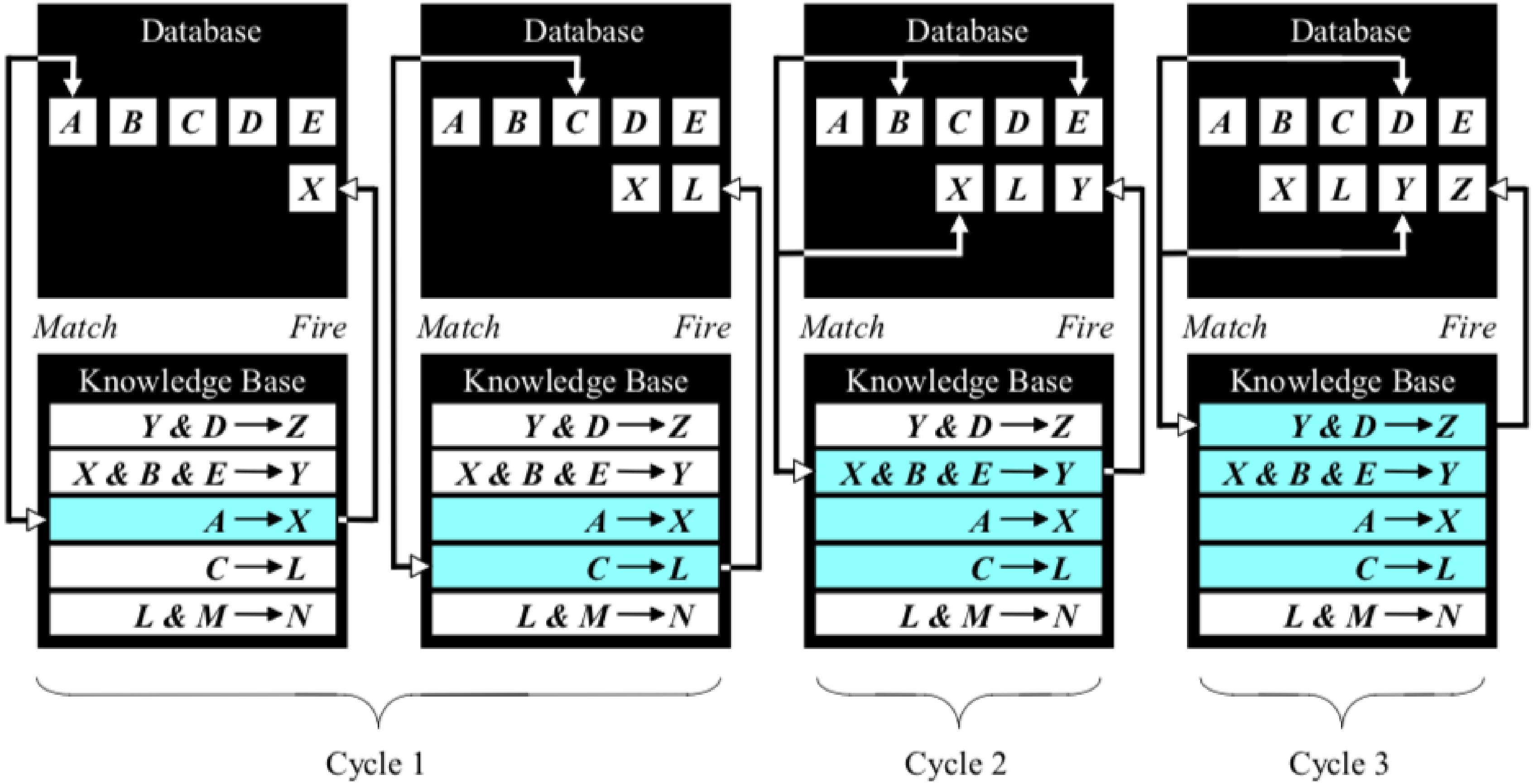
First cycle:

terdapat dua rules, yaitu:

Rule 3: *A* → *X*, dan Rule 4: *C* → *L*

Rules tsb cocok dengan fakta di database, maka Rule 3 diterima (sebagai topmost one) dan bagian THEN dieksekusi, fakta baru mengenai *X* ditambahkan pada database.

(begitu jg dgn rule 4)


Database
A
B
C
D
E
X
L
Y
Z
Match
Fire
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Cycle 1
Cycle 2
Cycle 3

- Berikut akan diberikan contoh lain, dimana Forward Chaining digunakan dalam penyelesaian 8-puzzle

8-puzzle as a production system

Start state:

2	8	3
1	6	4
7		5

Goal state:

1	2	3
8		4
7	6	5

1 **Production set:**

Condition		Action
goal state in working memory	→	halt
blank is not on the left edge	→	move the blank left
blank is not on the top edge	→	move the blank up
blank is not on the right edge	→	move the blank right
blank is not on the bottomedge	→	move the blank down

2 **Working memory is the present board state and goal state.**

3 **Control regime:**

1. Try each production in order.
2. Do not allow loops.
3. Stop when goal is found.

8-puzzle as a production system

Start state:

2	8	3
1	6	4
7		5

Goal state:

1	2	3
8		4
7	6	5

1 **Production set:**

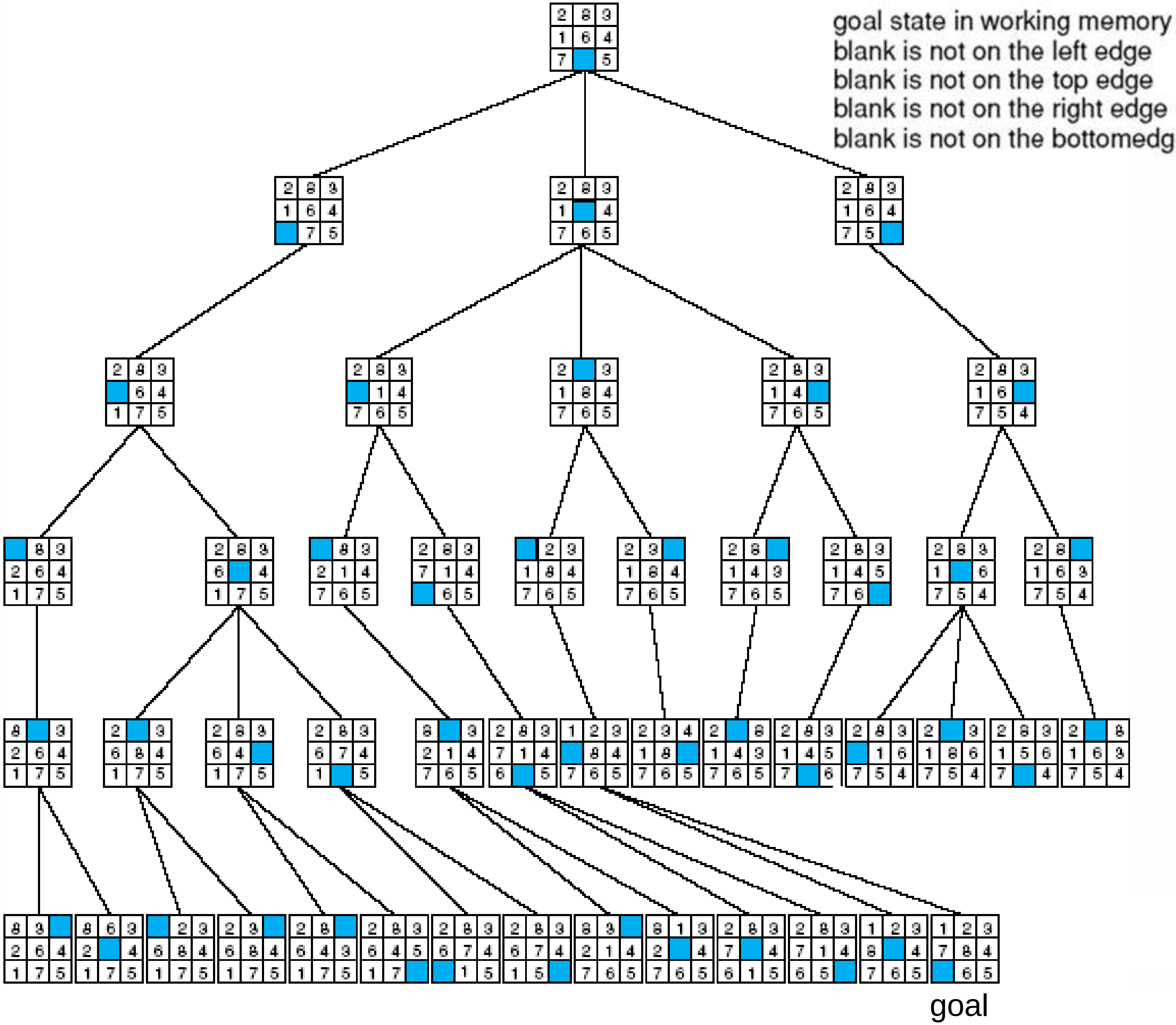
Condition		Action
goal state in working memory	→	halt
blank is not on the left edge	→	move the blank left
blank is not on the top edge	→	move the blank up
blank is not on the right edge	→	move the blank right
blank is not on the bottomedge	→	move the blank down

2 **Working memory is the present board state and goal state.**

3 **Control regime:**

1. Try each production in order.
2. Do not allow loops.
3. Stop when goal is found.

8-puzzle searched by a production system



Condition		Action	
goal state in working memory	→	halt	
blank is not on the left edge	→	move the blank left	←
blank is not on the top edge	→	move the blank up	↑
blank is not on the right edge	→	move the blank right	→
blank is not on the bottomedge	→	move the blank down	↓

Data-driven search: forward chaining

1

Production set:

1. $p \wedge q \rightarrow goal$
2. $r \wedge s \rightarrow p$
3. $w \wedge r \rightarrow q$
4. $t \wedge u \rightarrow q$
5. $v \rightarrow s$
6. $start \rightarrow v \wedge r \wedge q$

Trace of execution:

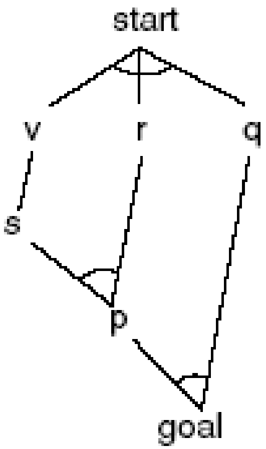
2

3 Fire the last rule in the set.

Iteration #	Working memory	Conflict set	Rule fired
0	start	6	6
1	start, v, r, q	6, 5	5
2	start, v, r, q, s	6, 5, 2	2
3	start, v, r, q, s, p	6, 5, 2, 1	1
4	start, v, r, q, s, p, goal	6, 5, 2, 1	halt

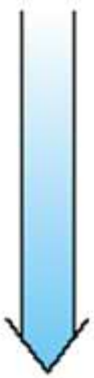
Space searched by execution:

Working memory contains true states.



Forward chaining
Direction of search

start

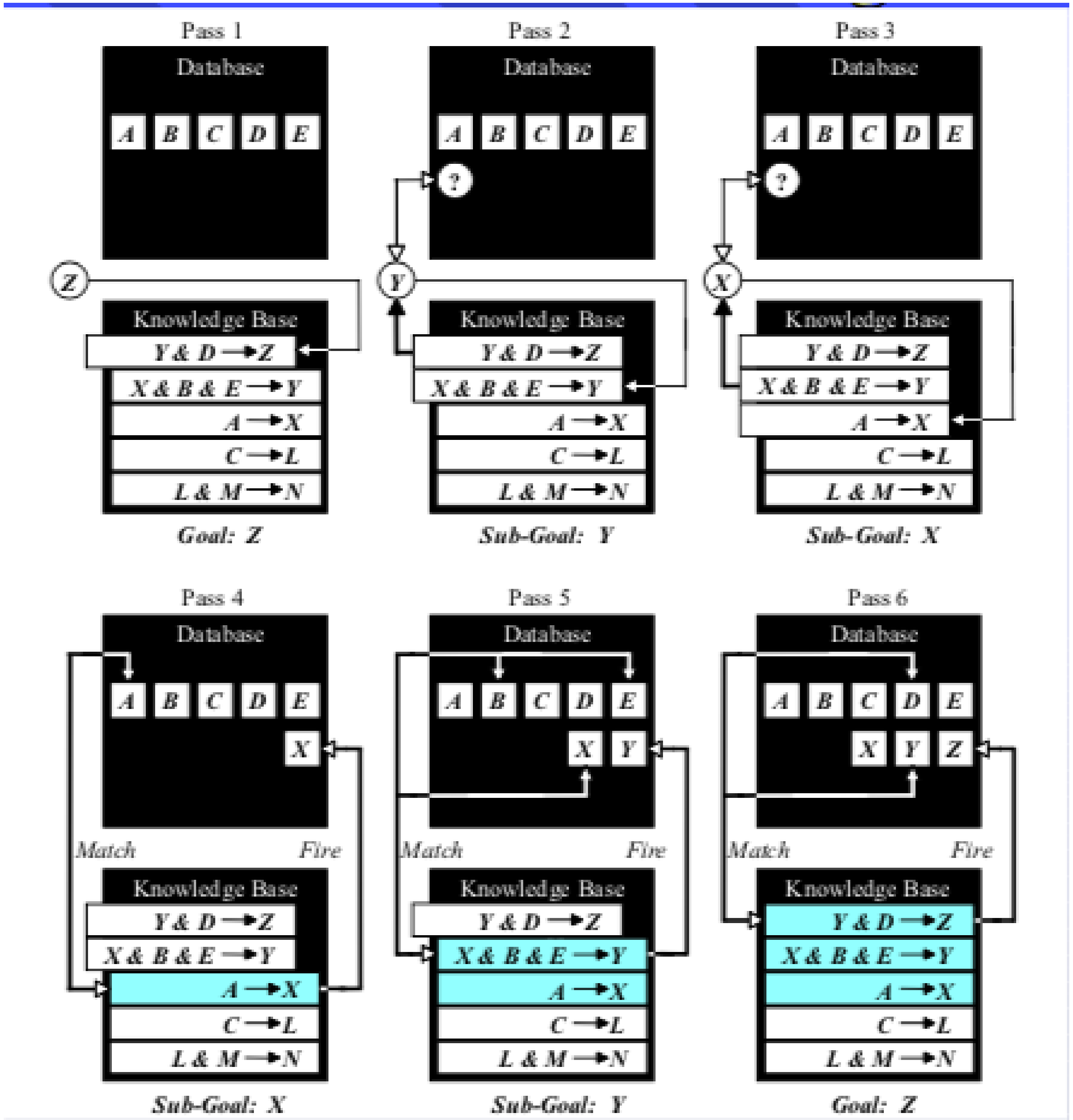


goal

BACKWARD CHAINING

- Disebut juga **Goal-driven** reasoning
- Dalam backward chaining, ES mempunyai goal dan inference engine berusaha mencari fakta-fakta untuk membuktikannya.
- Rule(s) mempunyai goal pada bagian THEN (action). Jika rule tersebut menemukan bagian IF yang cocok dengan data dalam database, maka rule tersebut **fire** dan goal terbukti (dieksekusi). Tetapi hal ini jarang terjadi

- Sehingga inference engine mengesampingkan aturan yang tadi digunakan (stack = ditumpuk) dan menetapkan tujuan baru, sebuah sub goal, untuk membuktikan bagian IF dari rule
- Kemudian engine akan mencari rule yang dapat membuktikan sub goal dari knowledge based
- Inference engine akan mengulang proses stacking (tumpukan) rule sampai tidak ada lagi rules yang didapatkan dalam knowledge base untuk membuktikan sub goal.


Pass 1
Database
A B C D E
Z
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Goal: Z
Pass 2
Database
A B C D E
?
Y
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Sub-Goal: Y
Pass 3
Database
A B C D E
?
X
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Sub-Goal: X
Pass 4
Database
A B C D E
X
Match
Fire
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Sub-Goal: X
Pass 5
Database
A B C D E
X Y
Match
Fire
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Sub-Goal: Y
Pass 6
Database
A B C D E
X Y Z
Match
Fire
Knowledge Base
Y & D → Z
X & B & E → Y
A → X
C → L
L & M → N
Goal: Z

Backward Chaining (*Goal-driven*)

Goal-driven fokus kepada goal, kemudian menentukan rules yang menghasilkan goal, kemudian ‘dihubungkan’ ke belakang (chain backward) menggunakan urutan-urutan rules dan subgoal untuk mendapatkan fakta dari problem.

Goal-driven search: backward chaining

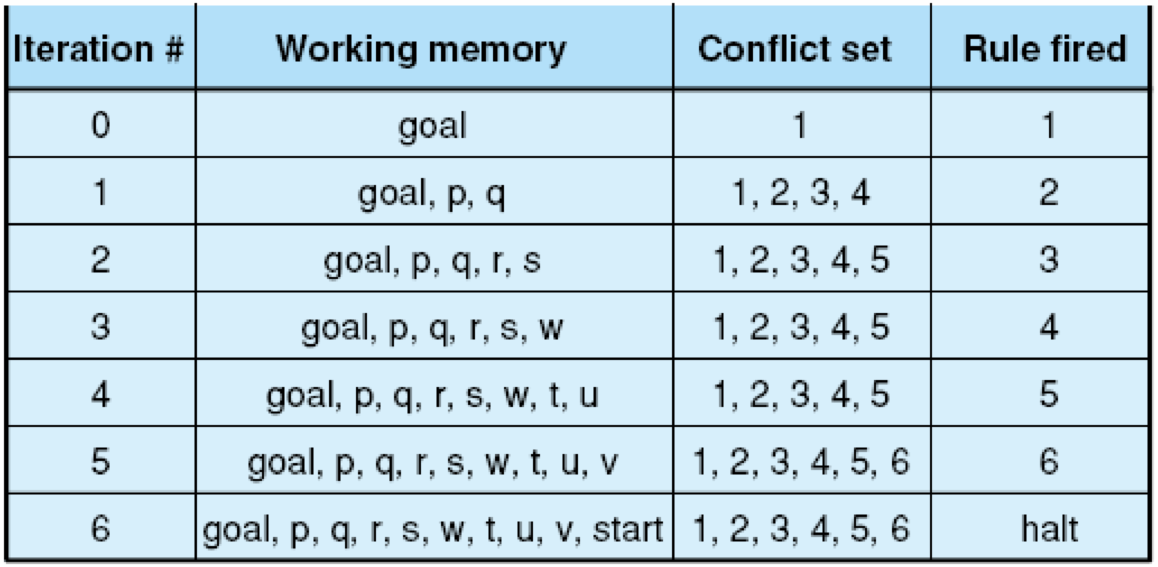
Production set:

1. $p \wedge q \rightarrow goal$
2. $r \wedge s \rightarrow p$
3. $w \wedge r \rightarrow p$
4. $t \wedge u \rightarrow q$
5. $v \rightarrow s$
6. $start \rightarrow v \wedge r \wedge q$

Trace of execution:

subgoals

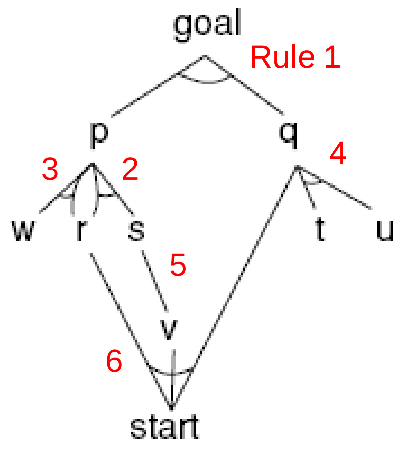
oldest untried rule

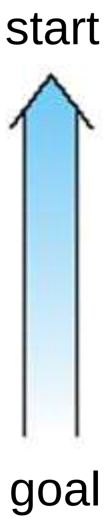


Iteration #	Working memory	Conflict set	Rule fired
0	goal	1	1
1	goal, p, q	1, 2, 3, 4	2
2	goal, p, q, r, s	1, 2, 3, 4, 5	3
3	goal, p, q, r, s, w	1, 2, 3, 4, 5	4
4	goal, p, q, r, s, w, t, u	1, 2, 3, 4, 5	5
5	goal, p, q, r, s, w, t, u, v	1, 2, 3, 4, 5, 6	6
6	goal, p, q, r, s, w, t, u, v, start	1, 2, 3, 4, 5, 6	halt

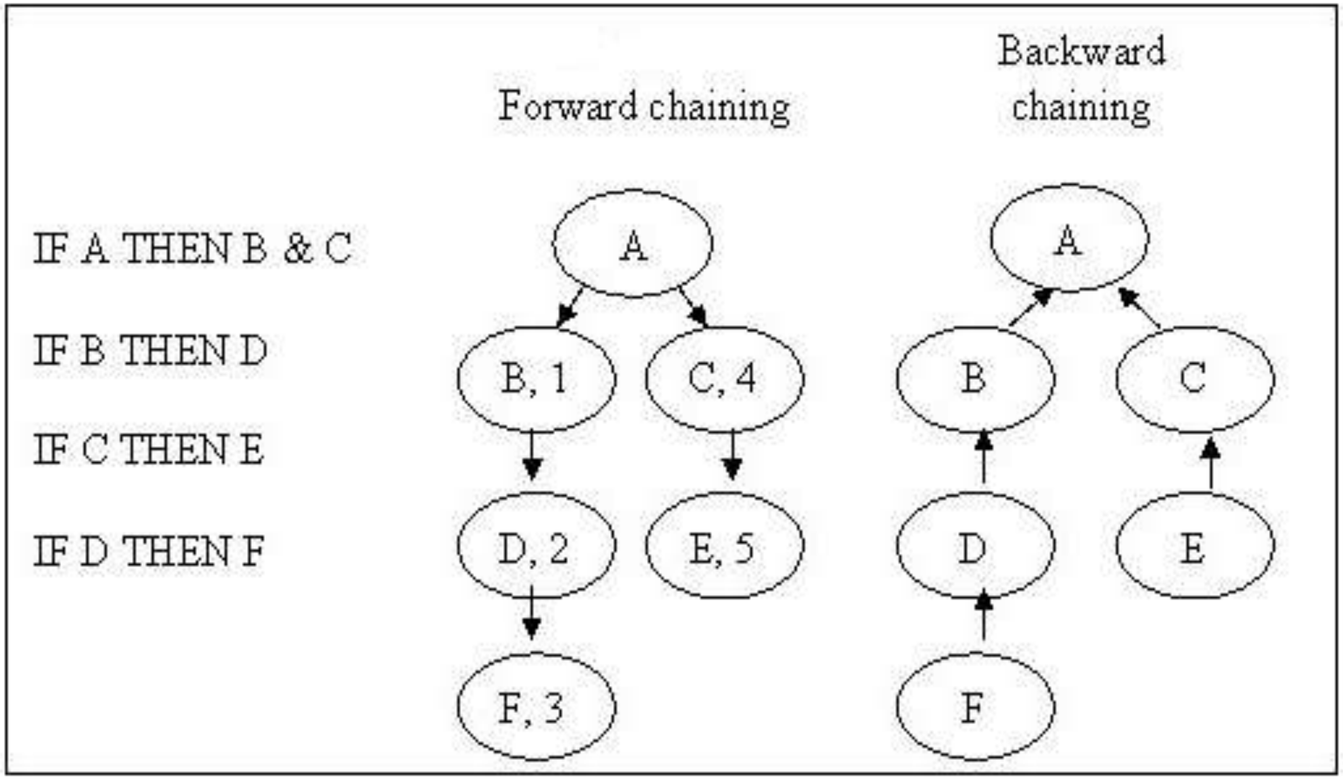
Working memory contains goal and sub-goal states waiting to be satisfied (shown true).

Space searched by execution:





Backward chaining
Direction of search



IF A THEN B & C

IF B THEN D

IF C THEN E

IF D THEN F

Contoh lain:

If corn is grown on poor soil, then it will get blackfly.

If soil hasn't enough nitrogen, then it is poor soil.

- Forward chaining: This soil is low in nitrogen; therefore this is poor soil; therefore corn grown on it will get blackfly.
- Backward chaining: This corn has blackfly; therefore it must have been grown on poor soil; therefore the soil must be low in nitrogen.

Forward chaining vs. backward chaining

- Data-driven, forward chaining
  - Starts with the initial given data and search for the goal.
  - At each iteration, new conclusion (RHS) becomes the pattern to look for next
  - Working memory contains true sentences (RHS's).
  - Stop when the goal is reached.
- Goal-driven is the reverse.
  - Starts with the goal and try to search for the initial given data.
  - At each iteration, new premise (LHS) becomes the new subgoals, the pattern to look for next
  - working memory contains subgoals (LHS's) to be satisfied.
  - Stop when all the premises (subgoals) of fired productions are reached.
- Sense of the arrow is in reality reversed.
- Both repeatedly pick the next rule to fire.

condition → action
premise → conclusion

	Forward chaining	**Backward chaining**
Starts with	premise	conclusion
Search for	conclusion	premise
Working memory	true statements	subgoals to be proved
Stopping criteria	goal is reached	Initial data are reached
	Data-driven	**Goal-driven**

Combining forward- and backward-chaining

- Begin with data and search forward until the number of states becomes unmanageably large.
- Switch to goal-directed search to use subgoals to guide state selection.

1. Show how to use backward chaining, forward chaining for the following example and construct an Inference Tree

Here is an example involving an investment decision: whether to invest in IBM stock. The following varibles are used: A= Have $10,000, B= Younger than 30, C = Education at college level, D= Annual income of at least $40,000, E= Invest in securities, F= Invest in growth stocks, G= Invest in IBM stock

Each of these variables can be answered as true or false.

The facts: We assume that an investor has $10,000(A is true) and that she is 25 years old (B is true). She would like advice on investing in IBM stock (Yes or no for the goal).

The rules: Our knowledge base contains five rules:

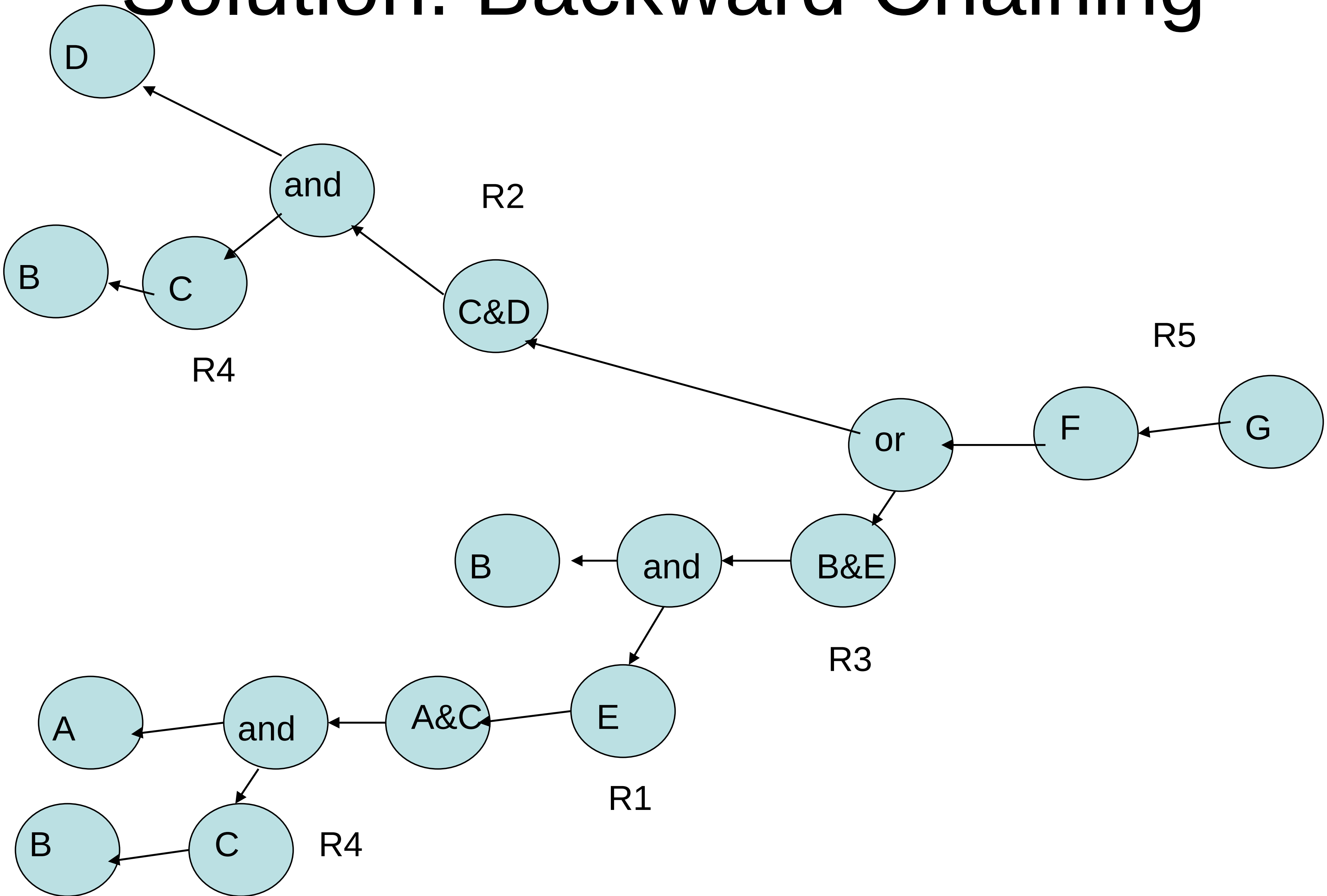
- R1: If A and C, Then E.
- R2: If D and C, Then F.
- R3: If B and E, Then F.
- R4: If B Then C
- R5: If F, Then G

Our goal is to determine whether to invest in IBM Stock.

Solution: Backward chaining

- Start: We start by looking for a rule that includes the goal(G) in its conclusion (THEN part). Because, R5 is the only rule that qualifies, we start with it.
- Step1: Try to accept or reject G. The ES goes to the assertion base to see whether G is there. Since we have in the assertion base : A is true, B is true, ES proceeds to step 2
- Step 2: R5 traces G to F. F is a premise of R5 is the conclusion of R2 and R3. There to check whether F is true, we need to check either of these two rules.
- Step 3: We try R2 first; if both D and C are true, then F is true. Since D is not a conclusion of any rule, ES tries to find out the whether D is true by asking a question to the investor.
- Step 4: ES does a backtracking and goes to R3: test B and E. We know that B is true because it is a given fact. To prove E, we go to R1, where E is the conclusion.
- Step 5: Examine R1. It is necessary to determine whether A and C are true.
- Step 6: A is true because it is a given fact. To test C, it is necessary to test R4.
- Step 7: R4 tells us that C is true. Therefore C becomes a fact . Now E is true, which validates F which validates the goal.

Solution: Backward Chaining



CONFLICT RESOLUTION

- RULE 1:

IF the traffic light is green

THEN the action is go

- RULE 2:

IF the traffic light is red

THEN the action is stop

- RULE 3:

IF the traffic light is red

THEN the action is go

Bagaimana menyelesaikan conflict?

1. Fire the rule with the highest priority

Contoh:

RULE 1: Meningitis Prescription1 (Priority 100)
IF Infection is Meningitis
AND The patient is a child

THEN Prescription in Number_1

AND Drug recommendation is Ampicillin

AND Drug recommendation is Gentamicin

AND Display Meningitis Prescription 1

- RULE 2 Meningitis Prescription 2 (Priority 90)

IF Infection is Meningitis

AND The patient is an Adult

THEN Prescription in Number_2

AND Drug recommendation is Penicillin

AND Display Meningitis Prescription 2

2. Fire the most specific rule

Contoh:

RULE 1:
IF the season is autumn
AND the sky is cloud
AND the forecast is rain
THEN the advice is stay at home

RULE 2:
IF the season is autumn
THEN the advice is bring umbrella.

Kesimpulan

- Data-driven, forward chaining
  - conditions first, then actions
  - working memory contains true statements describing the current environment
- Goal-driven, backward chaining
  - actions first, then conditions (subgoals)
  - working memory contains subgoals to be shown as true
- Mixed approach
  - Start with data and go forward until frontier is too big
  - Then start with goal and go backward
  - Try to connect the two in the middle of the state space
- Prolog implementation of production systems requires infinite loop detection.