Metode Inferensi

Outline

- Trees, Lattice dan Graph
- State dan Ruang Masalah
- AND-OR Tree dan Tujuan
- Penalaran Deduktif dan Syllogisms
- Kaidah Inferensi
- Keterbatasan Logika Proposisi
- Logika Predikat Order Pertama Kali
- Sistem Logika
- Resolusi, Sistem Resolusi dan Deduksi
- Shallow dan Casual Reasoning
- Rangkaian Forward dan Backward
- Metode Lain dari Inferensi

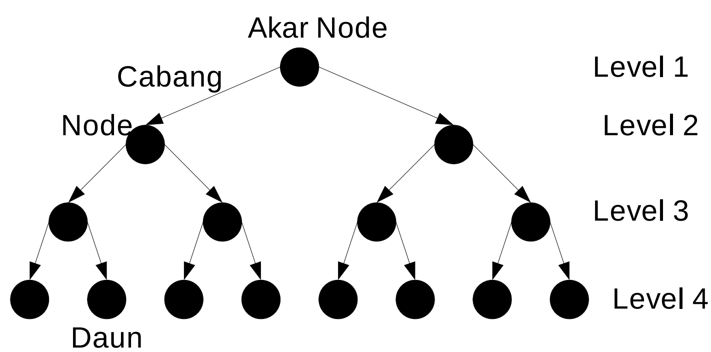
Referensi

Giarrantano, J and G.Riley, *Expert System : Principle and Programming,* 4th ed, PWS Kent, 2004

Tree, Lattice dan Graf (1/3)

- *Tree* (pohon) adalah suatu hierarki struktur yang terdiri dari *Node (simpul/veteks*) yang menyimpan informasi atau pengetahuan dan cabang (*link*/*edge*) yang menghubungkan node.

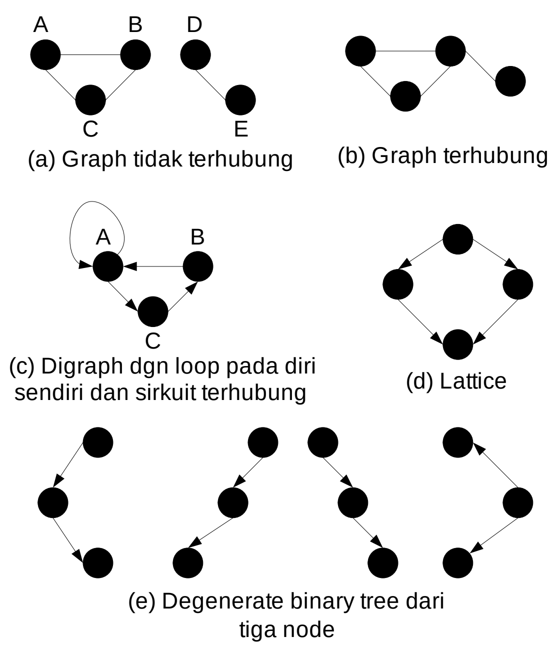
- Binary tree mempunyai 0,1 atau 2 cabang per-node.
  - Node tertinggi disebut root
  - Node terendah disebut daun



- Tree merupakan tipe khusus dari jaringan semantic, yang setiap nodenya kecuali akar, mempunyai satu node orang tua dan mempunyai nol atau lebih node anak.
- Tree adalah kasus khusus dalam Graph
- Graph dapat mempunyai nol atau lebih link di antara node dan tidak ada perbedaan antara orangtua dan anak.

Tree, Lattice dan Graf (2/3)

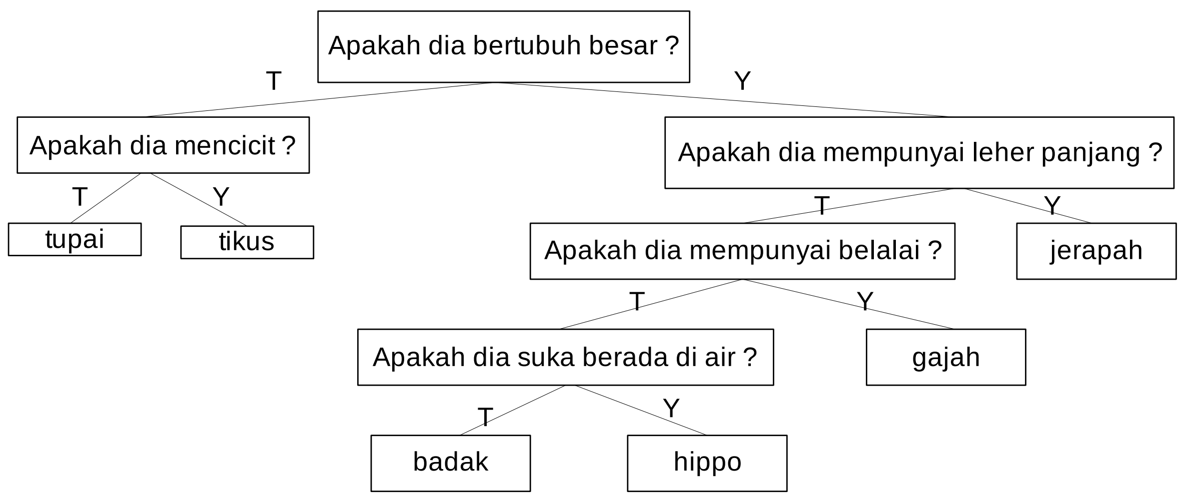
- Dalam graph, link dapat ditunjukkan berupa panah atau arah yang memadukan node dan bobot yang merupakan karakteristik beberapa aspek dari link.
- Beberapa contoh graph sederhana:



- Graph *asiklik* adalah graph yang tidak mengandung siklus.
- Graph dengan link berarah disebut *digraph*.
- Graph asiklik berarah disebut *lattice*.
- Tree yang hanya dengan path tunggal dari akar untuk satu daun disebut *degenerate tree*.

Tree, Lattice dan Graf (3/3)

- Aplikasi tree dan lattice adalah pembuatan keputusan disebut *decision tree* dan *decision lattice*.
- Contoh : decision tree yang menunjukkan pengetahuan tentang hewan.



- Aturan produksi (IF…THEN…) dari contoh di atas :

  JIKA pertanyaan=”Apakah dia bertubuh besar ?”

  DAN jawaban=”Tidak”

  MAKA pertanyaan=”Apakah dia mencicit?”

  JIKA pertanyaan=”Apakah dia bertubuh besar ?”

  DAN jawaban=”Ya”

  MAKA pertanyaan=”Apakah dia mempunyai leher panjang?”

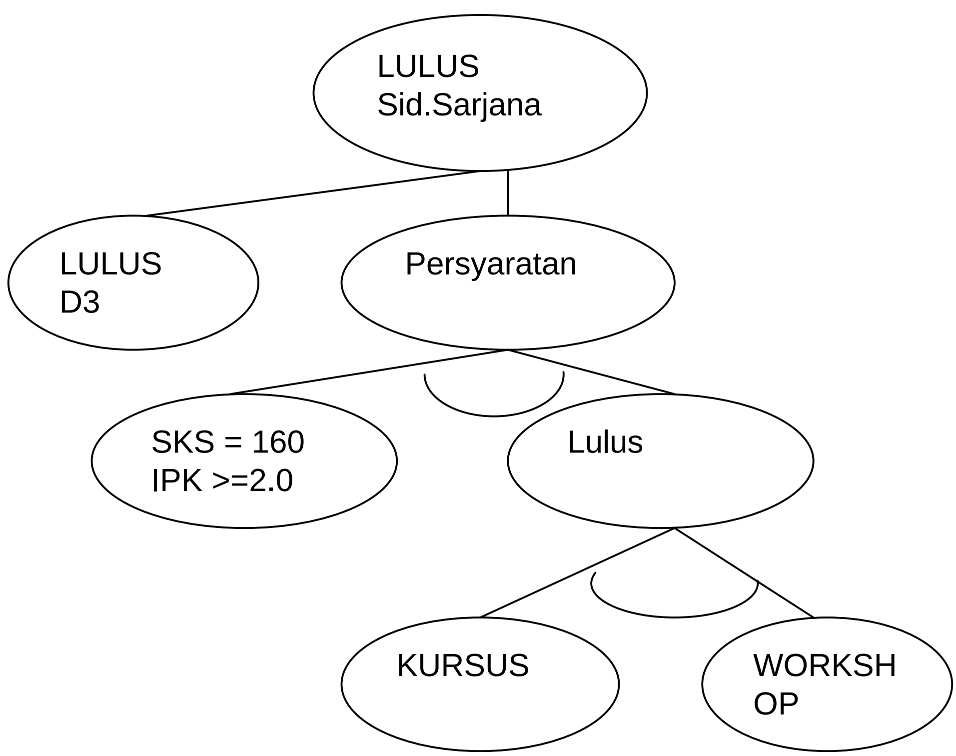
  dst……

Keadaan dan Ruang Masalah

- Graf dapat diaplikasikan untuk memecahkan beragam masalah.
- Metode yang menggambarkan perilaku dari objek yang didefinisikan dengan graf disebut Ruang Keadaan (State Space).
- Suatu Keadaan (State)adalah kumpulan beragam karakteristik yang digunakan untuk mendefinisikan status atau keadaan suatu objek.
- Ruang Keadaan merupakan suatu himpunan keadaan yang menunjukkan transisi antar keadaan dari objek.
- Contoh :
  - Pembelian soft-drink dengan cara memasukkan koin ke dalam Vending Machine dan memilih jenis soft-drink
  - Travelling Salesman Problem

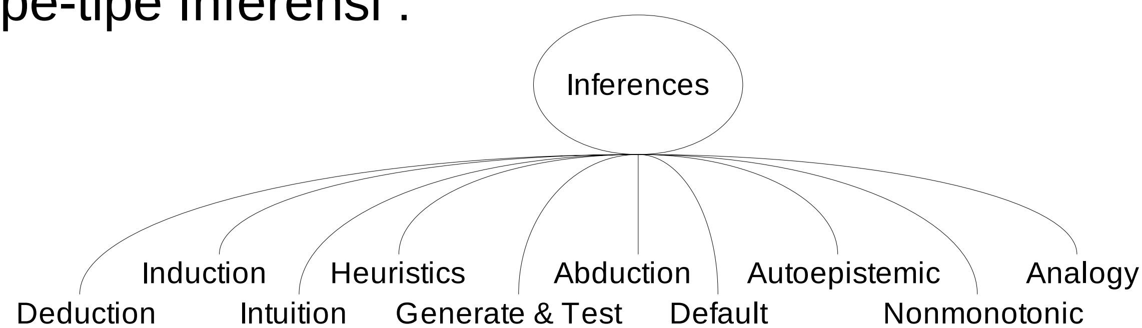
Pohon AND-OR dan Tujuan

- Banyak tipe system pakar menggunakan *backward chaining* untuk mendapatkan solusi dari permasalahan.
- Salah satu tipe dari tree atau lattice yang digunakan dalam masalah representasi *backward chaining* adalah Pohon.
- Contoh :



Penalaran Deduktif dan Silogisme (1/8)

Tipe-tipe Inferensi :



- **Deduction**
  Pemberian alasan logikal dimana kesimpulan harus mengikuti premis
- **Induction**
  Inferensi dari khusus ke umum
- **Intuition**
  Tidak ada teori yg menjamin. Jawabannya hanya muncul, mungkin dengan penentuan pola yg ada secara tidak disadari
- **Heuristic**
  Aturan yg didasarkan pada pengalaman
- **Generate & Test**

  Trial dan error. Digunakan dgn perencanaan

Penalaran Deduktif dan Silogisme (2/8)

- **Abduction**

  Pemberian alasan kembali dari kesimpulan yg benar ke premis
- **Default**

  Diasumsikan pengetahuan umum sebagai default
- **Autoepistemic**

  Self-knowledge
- **Nonmonotonic**

  Pengetahuan yg sebelumnya mungkin tdk benar jika bukti baru didapatkan
- **Analogy**

  Kesimpulan yg berdasarkan pada persamaan untuk situasi yg lainnya.

Penalaran Deduktif dan Silogisme (3/8)

- Suatu logika argument adalah kumpulan dari pernyataan-pernyataan yang dinyatakan untuk dibenarkan sebagai dasar dari rantai penalaran.
- Salah satu jenis logika argunen adalah ***Silogisme***.
- Contoh :

  Premis : Siapapun yang dapat membuat program adalah pintar

  Premis: John dapat membuat program

  Konklusi: Oleh karenanya John adalah pintar

  Proses deduktif pada contoh di atas bergerak dari prinsip umum menuju konklusi khusus.
- Penalaran deduktif umumnya terdiri dari tiga bagian : *premis mayor, premis minor* dan *konklusi.*
- Premis disebut juga *antecedent*
- Konklusi/kesimpulan disebut juga *consequent*

Penalaran Deduktif dan Silogisme (4/8)

- Silogisme dapat direpresentasikan ke dalam bentuk aturan JIKA…..MAKA….. (IF…THEN…..), contoh :

  JIKA siapapun yang dapat membuat program adalah pintar
  DAN John dapat membuat program
  MAKA John adalah pintar

- Silogisme klasik disebut *categoricall syllogism* (silogisme yang pasti)
- Premis dan konklusi didefinisikan sebagai statement yang pasti dari empat bentuk berikut :

Bentuk	Skema	Arti
A	Semua S adalah P	Universal Afirmative
E	Tidak S adalah P	Universal Negative
I	Beberapa S adalah P	Particular Afirmative
O	Beberapa S bukan P	ParticularNegative

  – Subjek dari konklusi S disebut bagian minor bila predikat konklusi P adalah bagian mayor.
  – Premis terdiri dari premis mayor dan premis minor.

Penalaran Deduktif dan Silogisme (5/8)

- Contoh :

  Premis mayor : Semua M adalah P
  Premis minor : Semua S adalah M
  Konklusi : Semua S adalah P

- Contoh :

  "Semua mikrokomputer adalah computer"
  Subjeknya (objek yang digambarkan) adalah mikrokomputer.
  Predikatnya (beberapa sifat subjek) adalah computer

- M (middle term) adalah hal yang penting karena silogisme didefinisikan sedemikian sehingga konklusi tidak dapat disimpulkan dengan mengambil salah satu premis.

- Q (quantifier) menggambarkan porsi dari kelas yang diketahui.
  - Quantifier "semua" dan "tidak" adalah universal karean menunjukkan keseluruhan kelas.
  - "beberapa" adalah khusus (particular) karena hanya menunjukkan satu bagian dari kelas yang diketahui.

Penalaran Deduktif dan Silogisme (6/8)

- Mood dari silogisme didefinisikan sebagai tiga huruf yang memberikan bentuk masing-masing premis mayor, minor dan konklusi.
- Contoh :

  Semua M adalah P

  <u>Semua S adalah M</u>

  ∴Semua S adalah P

  menunjukkan suatu mood AAA-1

- Ada 4 kemungkinan pola susunan istilah S, P dan M :

	Figure 1	Figure 2	Figure 3	Figure 4
Premis Mayor	MP	PM	MP	PM
Premis Minor	SM	SM	MS	MS

Penalaran Deduktif dan Silogisme (7/8)

- Tidak selalu argument yang mempunyai bentuk silogisme merupakan silogisme yang valid.
- Contoh : Silogisme tidak valid berbentuk AEE-1

  Semua M adalah P

  Tidak S adalah M

  ∴Tidak S adalah P

  Semua mikrokomputer adalah computer

  Bukan mainframe adalah mikrokomputer

  ∴Bukan mainframe adalah computer
- Diperlukan prosedur keputusan (*decision procedure)* untuk pembuktian validitas.
- Prosedur keputusan untuk silogisme dapat dilakukan menggunakan diagram venn tiga lingkaran yang saling berpotongan yang merepresentasikan S,P, M.

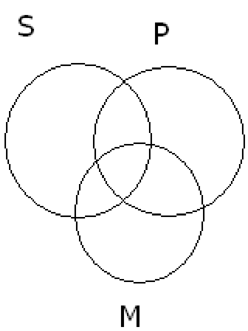
Penalaran Deduktif dan Silogisme (8/8)

- Contoh : Prosedur Keputusan untuk AEE-1

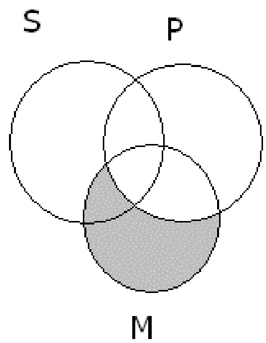
Semua M adalah P

<u>Tidak S adalah M</u>

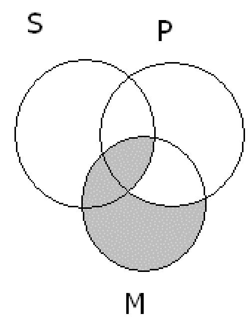
∴Tidak S adalah P



a. Diagram Venn



b. Setelah Premis Mayor



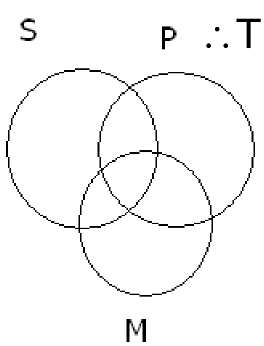
c. Setelah Premis Minor

- Contoh : Prosedur Keputusan untuk EAE-1

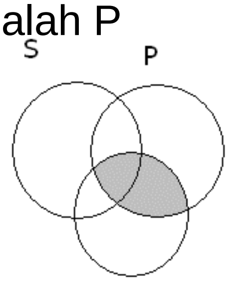
Tidak M adalah P

<u>Semua S adalah M</u>

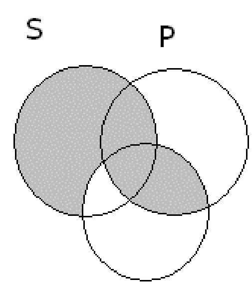
∴Tidak S adalah P



a. Diagram Venn



b. Setelah Premis Mayor



c. Setelah Premis Minor

Kaidah dari Inferensi (1/6)

- Diagram Venn tidak sesuai untuk argumen yang lebih kompleks karena sulit dibaca pada decision tree untuk silogisme.
- Logika proposisi memberikan pengertian lain dari penggambaran argumen.
- Contoh :

  Jika ada daya listrik, komputer akan bekerja

  Ada daya

  ∴ Komputer akan bekerja

Jika : A = ada daya listrik

B = komputer akan bekerja

Sehingga dapat ditulis :

$$\begin{array}{l} A \rightarrow B \\ \underline{A \quad\quad} \\ \therefore B \end{array}$$

Kaidah dari Inferensi (2/6)

- Bentuk umum ***Ponens*** / direct reasoning / law of detachment / assuming the antecedent

$$\begin{array}{l} p \rightarrow q \\ \underline{p \qquad\qquad} \\ \therefore q \end{array} \qquad \text{atau} \quad p \rightarrow q,\ \ p;\ \ \therefore q$$

Bentuk tersebut valid, karena argumen tersebut dapat ditunjukkan sebagai suatu *tautologi*.

$$((p \rightarrow q) \wedge p) \rightarrow q$$

Tabel Kebenaran Ponens :

p	q	$p \rightarrow q$	$((p \rightarrow q) \wedge p)$	$((p \rightarrow q) \wedge p) \rightarrow q$
T	T	T	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	T

Kaidah dari Inferensi (3/6)

- Terdapat argumen yang menyerupai ponens namun perlu dibuktikan validitasnya.

Contoh :

Jika tidak kesalahan maka program dapat mengkompile

Program dapat mengkompile

$\therefore$ Tidak ada kesalahan

$$\begin{array}{l} p \rightarrow q \\ q \\ \hline \therefore p \end{array} \qquad \text{atau} \quad p \rightarrow q, \; q; \; \therefore p$$

Tabel Kebenaran:

p	q	$p \rightarrow q$	$((p \rightarrow q) \wedge q)$	$((p \rightarrow q) \wedge q) \rightarrow p$
T	T	T	T	T
T	F	F	F	T
F	T	T	T	F
F	F	T	F	T

(Bukan Pones karena tidak bersifat Tautology)

Kaidah dari Inferensi (4/6)

- Skema argumen lain :

$$\begin{array}{l} p \rightarrow q \\ \sim q \\ \hline \therefore \sim p \end{array}$$

Tabel Kebenaran:

p	q	p→q	~q	(p→q)∧~q)	~p	((p→q)∧~q) →~p
T	T	T	F	F	F	T
T	F	F	T	F	F	T
F	T	T	F	F	T	T
F	F	T	T	T	T	T

Argumen di atas disebut ***Tollens*** / indirect reasoning / law of contraposition.

Kaidah dari Inferensi (5/6)

Hukum Inferensi	Skema
1. Hukum Detasemen	$p \rightarrow q$ p ――― $\therefore q$
2. Hukum Kontrapositif	$p \rightarrow q$ ――― $\therefore \sim q \rightarrow \sim p$
3. Hukum Modus Tollens	$p \rightarrow q$ $\sim q$ ――― $\therefore \sim p$
4. Aturan Rantai (hukum silogisme)	$p \rightarrow q$ $q \rightarrow r$ ――― $\therefore p \rightarrow r$
5. Hukum Disjungsi	$P \vee q$ $\sim p$ ――― $\therefore q$ $p \vee q$ $\sim q$ ――― $\therefore p$
6. Hukum Negasi Ganda	$\sim(\sim p)$ ――― $\therefore p$
7. Hukum De Morgan	$\sim(p \wedge q)$ ――― $\therefore \sim p \vee \sim q$ $\sim(p \vee q)$ ――― $\therefore \sim p \wedge \sim q$

Kaidah dari Inferensi (6/6)

Hukum Inferensi	Skema
8. Hukum Simplifikasi	$\dfrac{p \wedge q}{\therefore p}$ $\quad$ $\dfrac{p \wedge q}{\therefore q}$
9. Hukum Konjungsi	$\dfrac{\begin{matrix} p \\ q \end{matrix}}{\therefore p \wedge q}$
10. Hukum Penambahan Disjungtif	$\dfrac{p}{\therefore p \vee q}$
11. Hukum Argumen Konjungtif	$\dfrac{\begin{matrix} \sim(p \wedge q) \\ p \end{matrix}}{\therefore \sim q}$ $\quad$ $\dfrac{\begin{matrix} \sim(p \wedge q) \\ q \end{matrix}}{\therefore \sim p}$

Kaidah dari Inferensi (6/)

- Kaidah inferensi dapat digunakan untuk argumen yang mempunyai lebih dari dua premis.
- Contoh :

Harga chip naik hanya jika yen naik
Yen naik hanya jika dollar turun dan
jika dollar turun maka yen naik
Karena harga chip telah naik
$\therefore$ Dollar harus turun
Misal : C = harga chip naik
Y = Yen naik
D = Dollar turun

1. $C \rightarrow Y$
2. $(Y \rightarrow D) \wedge (D \rightarrow Y)$
3. C

$\therefore D$

- Kondisional $p \rightarrow q$ mempunyai converse, inverse dan kontrapositif

Kondisional	$p \rightarrow q$
Converse	$q \rightarrow p$
Inverse	$\sim p \rightarrow \sim q$
Kontrapositif	$\sim q \rightarrow \sim p$

Jika $p \rightarrow q$ dan $q \rightarrow p$ bernilai benar, maka keduanya adalah ekuivalen.
$p \rightarrow q \wedge q \rightarrow p$ ekuivalen dengan $p \leftrightarrow q$ atau $p \equiv q$.

sehingga argumen untuk contoh di atas, menjadi :

1.	$C \rightarrow Y$	
2.	$(Y \rightarrow D) \wedge (D \rightarrow Y)$	
3.	C	$/\therefore D$
4.	$Y \equiv D$	2 ekuivalen
5.	$C \rightarrow D$	1 substitusi
6.	D	3,5 modus ponens

KETERBATASAN LOGIKA PROPOSISI (1/2)

- Perhatikan contoh berikut :
  All men are mortal
  Socrates is a man
  Therefore, Socrates is mortal

Misal : p = All men are mortal
q = Socrates is a man
r = Socrates is mortal

Skema argumennya menjadi : p, q; ∴ r

p
q
∴ r

Bila dibuat tabel kebenaran, hasilnya invalid.

- Argumen invalid sering diinterpretasikan sebagai konklusi yang salah (walaupun beberapa orang berpendapat argumen itu dapat saja bernilai benar).

- Argumen yang invalid berarti argumen tersebut tidak dapat dibuktikan dengan logika proposisi.

KETERBATASAN LOGIKA PROPOSISI (2/2)

- Keterbatasan logika proposisi dapat diatasi melalui logika predikat sehingga argumen tersebut menjadi valid.
- Kenyataannya, semua logika silogistik adalah subset yang valid dari logika proposisi urutan pertama.
- Contoh :

  If Socrates is a man, then Socrates is mortal
  Socrates is a man
  Therefore, Socrates is mortal

Misal: p = Socrates is a man
q = Socrates is mortal

Argumennya menjadi :

$$\begin{array}{l} p \rightarrow q \\ p \\ \hline q \end{array}$$

Argumen di atas adalah silogistik yang valid, yaitu bentuk *modus ponens*.

LOGIKA PREDIKAT URUTAN PERTAMA (First Order Predicate Logic) (1/2)

- Representasi 4 kategori silogisme menggunakan logika predikat

Bentuk	Skema	Representasi Predikat
A	Semua S adalah P	$(\forall x)\ (S(x) \rightarrow P(x))$
E	Tidak S adalah P	$(\forall x)\ (S(x) \rightarrow \sim P(x))$
I	Beberapa S adalah P	$(\exists x)\ (S(x) \rightarrow P(x))$
O	Beberapa S bukan P	$(\exists x)\ (S(x) \rightarrow \sim P(x))$

- Kaidah Universal Instatiation merupakan state dasar, dimana suatu individual dapat digantikan (disubsitusi) ke dalam sifat universal.

LOGIKA PREDIKAT URUTAN PERTAMA
(First Order Predicate Logic) (2/2)

- Contoh :

  Misal, ϕ merupakan fungsi proposisi :

$$\frac{(\forall x)\ \phi(x)}{\therefore\ \phi(a)}$$

  merupakan bentuk yang valid, dimana a menunjukkan spesifik individual, sedangkan x adalah suatu variabel yang berada dalam jangkauan semua individu (universal)

- Contoh lain :

$$\frac{(\forall x)\ H(x)}{\therefore\ H(Socrates)}$$

- Berikut ini adalah contoh pembuktian formal silogisme

  All men are mortal

  Socrates is a man

  Therefore, Socrates is mortal

Misal : H = man, M = mortal, s = Socrates

1. $(\forall x)\ (H(x) \rightarrow M(x))$	
2. $H(s)$	/ $\therefore$ $M(s)$
3. $H(s) \rightarrow M(s)$	1 Universal Instatiation
4. $M(s)$	2,3 Modus Ponens

SISTEM LOGIKA (1/3)

- Sistem logika adalah kumpulan objek seperti kaidah (rule), aksioma, statement dan lainnya yang diatur dalam cara yang konsisten.

- Sistem logika mempunyai beberapa tujuan :

1. Menentukan bentuk argumen.

   Awalnya argumen logika tidak memiliki arti dalam semantic sense, bentuk yang valid pada dasarnya dapat dicapai jika validitas dari argumen tersebut dapat ditentukan.

   Fungsi terpenting dari logika sistem adalah menentukan **well formed formulas (wffs)** dari argumen yang digunakan.

   Contoh :

   All S is P . merupakan wffs

   tapi…. All
   All is S P ….. bukan wffs
   Is S all

2. Menunjukkan kaidah inferensi yang valid.
3. Mengembangkan dirinya sendiri dengan menemukan kaidah baru inferensi dan memperluas jangkauan argumen yang dapat dibuktikan.

SISTEM LOGIKA (2/3)

- Sistem logika dibangun melalui Sentential atau kalkulus proposisi, kalkulus predikat dst.
- Setiap sistem disandarkan pada **aksioma** atau **postulat**, yang merupakan definisi mendasar dari sistem.
- Suatu **aksioma** merupakan fakta sederhana atau assertion yang tidak dapat dibuktikan dalam sistem. Terkadang, kita menerima aksioma dikarenakan ada sesuatu yang menarik atau melalui pengamatan.
- Sistem formal membutuhkan :
  1. simbol alfabet.
  2. suatu set finite string dari simbol tertentu, wffs
  3. aksioma, definisi dari sistem
  4. kaidah inferensi, yang memungkinkan wffs, A untuk dikurangi sebagai kesimpulan dari set finite Γ wff lain dimana Γ = {A1,A2,…An}. Wffs harus berupa aksioma atau teori lain dari sistem logis.

Sebagai contoh : sistem logika dapat didefinisikan menggunakan modus pones untuk diturunkan menjadi teorema baru.

SISTEM LOGIKA (3/3)

- Jika terdapat argumen :

  A1, A2, ……., AN; ∴ A

  yang valid, maka A disebut teorema dari sistem logika formal dan ditulis dengan simbol (metasymbol) yang menunjukkan wff adalah suatu **teorema** .

  A1, A2, ……., AN |- A

- Contoh : teorema silogisme tentang Socrates yang ditulis dalam bentuk logika predikat.

  $(\forall x)$ (H (x)→M(x)), H(s) |- M(s)

  M(s) dapat dibuktikan dari aksioma di sisi kiri, hal tersebut menunjukkan aksioma

- Suatu teorema merupakan **tautology**, ditunjukkan melalui Γ sebagai set null dimana wff selalu bernilai null dan tidak tergantung dari aksioma atau teorema yang lain.
- Teorema dengan tautology ditulis dengan simbol |=, misalnya |= A.
- Contoh : Jika A $\equiv$ p $\vee$ ~p maka |= p $\vee$ ~p

- Suatu model adalah interpretasi wff bernilai benar.

  Suatu wff disebut **konsisten** atau **satifiable** jika interpretasi yang dihasilkan benar, dan disebut **inkonsisten** atau **unsatisfiable** jika wff menghasilkan nilai yang salah pada semua interpretasi.

RESOLUSI (1/3)

- Diperkenalkan oleh Robinson (1965).
- Resolusi merupakan kaidah inferensi utama dalam bahasa PROLOG.
- PROLOG menggunakan notasi “quantifier-free”.
- PROLOG didasarakan pada logika predikat urutan pertama.
- Sebelum resolusi diaplikasikan, wff harus berada dalam **bentuk normal** atau **standard**.
- Tiga tipe utama bentuk normal : **conjunctive normal form, clausal form** dan **subset Horn clause**.
- Resolusi diaplikasikan ke dalam bentuk normal wff dengan menghubungkan seluruh elemen dan quantifier yang dieliminasi.
- Contoh :

  $(A \vee B) \wedge (\sim B \vee C)$ ………… conjunctive normal form

  Dimana $A \vee B$ dan $\sim B \vee C$ adalah clause.

RESOLUSI (2/3)

- Logika proposional dapat ditulis dalam bentuk clause.
- **Full clause form** yang mengekspresikan formula logika predikat dapat ditulis dalam **Kowalski clause form**.

  A1, A2, ……., AN $\rightarrow$ B1, B2, ……., BM

  Clause yang ditulis dalam notasi standard :

  A1$\wedge$ A2, ……., AN $\rightarrow$ B1 $\vee$ B2, ……., BM

  Bentuk disjungsinya merupakan disjungsi dari literal menggunakan equivalence :

  $p \rightarrow q \equiv \sim p \vee q$

  sehingga

  A1$\wedge$ A2, ..., AN $\rightarrow$ B1 $\vee$ B2, ……., BM

  $\equiv$ ~(A1$\wedge$ A2, …, AN) $\vee$ (B1 $\vee$ B2, …, BM)

  $\equiv$ ~A1$\vee$ ~A2, …, ~AN $\vee$ B1 $\vee$ B2, …., BM

  Yang merupakan hukum de Morgan :

  $\sim(p \wedge q) \equiv \sim p \vee \sim q$

  Dengan Horn clause dapat ditulis :

  A1, A2, ……., AN $\rightarrow$ B

  Dalam bahasa PROLOG ditulis :

  B :- A1, A2, ……., AN

RESOLUSI (3/3)

- Untuk membuktikan teorema di atas benar, digunakan metode klasik **reductio ad absurdum** atau **metode kontradiksi**.
- Tujuan dasar resolusi adalah membuat infer klausa baru yang disebut "**revolvent**" dari dua klausa lain yang disebut **parent clause**.
- Contoh :

  $A \vee B$

  $A \vee \sim B$

  $\therefore A$

  Premis dapat ditulis : $(A \vee B) \wedge (A \vee \sim B)$

  Ingat Aksioma Distribusi :

  $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee q)$

  Sehingga premis di atas dapat ditulis :

  $(A \vee B) \wedge (A \vee \sim B) \equiv A \vee (B \wedge \sim B) \equiv A$

  dimana $B \wedge \sim B$ selalu bernilai salah.

- Tabel Klausa dan Resolvent

Parent Clause	Resolvent	Arti
$p \rightarrow q$, p atau $\sim p \vee q$, p	q	Modus Pones
$p \rightarrow q$, $q \rightarrow r$ atau $\sim p \vee q$, $\sim q \vee r$	$p \rightarrow r$ atau $\sim p \vee r$	Chaining atau Silogisme Hipotesis
$\sim p \vee q$, $p \vee q$	q	Penggabungan
$\sim p \vee \sim q$, $p \vee q$	$\sim p \vee p$ atau $\sim q \vee q$	TRUE (tautology)
~p, p	Nill	FALSE (kontradiksi)

SISTEM RESOLUSI DAN DEDUKSI (1/2)

- **Refutation** adalah pembuktian teorema dengan menunjukkan negasi atau pembuktian kontradiksi melalui reductio ad absurdum.

Melakukan refute berarti membuktikan kesalahan.

Contoh :

$$\begin{array}{l} A \rightarrow B \\ B \rightarrow C \\ C \rightarrow D \\ \hline \therefore A \rightarrow D \end{array}$$

Untuk membuktikan konklusi $A \rightarrow D$ adalah suatu teorema melalui resolusi refutation, hal yang dilakukan :

$$p \rightarrow q \quad \equiv \quad \sim p \vee q$$

sehingga

$$A \rightarrow D \quad \equiv \quad \sim A \vee D$$

dan langkah terakhir adalah melakukan negasi

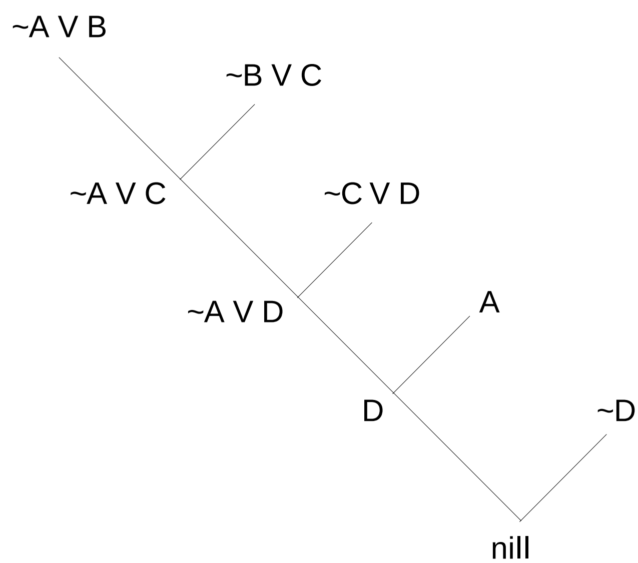
$$\sim(\sim A \vee D) \equiv \quad A \wedge \sim D$$

Penggunaan konjungsi dari disjunctive form pada premis dan negasi pada konsklusi, memberikan conjuctive normal form yang cocok untuk **resolusi refutation.**

SISTEM RESOLUSI DAN DEDUKSI (2/2)

- **Dari contoh di atas, penulisannya menjadi :**

  $(\sim A \vee B) \wedge (\sim B \vee C) \wedge (\sim C \vee D) \wedge A \wedge \sim D$



Pohon Resolusi Refutation

Akar bernilai nill, menunjukkan kontradiksi. Sehingga melalui refutation dapat ditunjukkan konklusi asli (awal) adalah teorema dengan peran kontradiksi.

RESOLUSI LOGIKA PROPOSISI (1/4)

- Dalam proposisi, resolusi merupakan aturan untuk melakukan inferensi yang dapat berjalan secara efisien dalam suatu bentuk khusus, yaitu *conjunctive normal form* (CNF)
- Bentuk CNF memiliki ciri-ciri :
  - Setiap kalimat merupakan disjungsi literal
  - Semua kalimat terkonjungsi secara implisit.
- Untuk mengubah suatu kalimat ke dalam bentuk CNF, dapat digunakan langkah-langkah sebagai berikut :
  - Hilangkan implikasi dan euivalensi
    - $x \rightarrow y$ menjadi $\sim x \vee y$
    - $x \leftrightarrow y$ menjadi $(\sim x \vee y) \wedge (\sim y \vee x)$
- Kurangi lingkup semua negasi menjadi satu negasi saja
    - $\sim(\sim x)$ menjadi x
    - $\sim(x \vee y)$ menjadi $(\sim x \wedge \sim y)$
    - $\sim(x \wedge y)$ menjadi $(\sim x \vee \sim y)$
- Gunakan aturan asosiatif dan distributif untuk mengkonversi menjadi conjuction of disjunction.
    - Assosiatif : $(A \vee B) \vee C = A \vee (B \vee C)$
    - Distributif : $(A \wedge B) \vee C = (A \vee C) \wedge (B \vee C)$
- Buatu satu kalimat terpisah untuk tiap-tiap konjungsi.

RESOLUSI LOGIKA PROPOSISI (2/4)

- Pada logika proposisi, prosedur untuk membuktikan proposisi P dengan beberapa aksioma F yang telah diketahui, dengan menggunakan resolusi, dapat dilakukan melalui algoritma sebagai berikut :
- Konversikan semua proposisi F ke bentuk CNF
- Negasikan P, dan konversikan hasil negasi tersebut ke bentuk klausa. Tambahkan ke himpunan klausa yang telah ada pada langkah 1.
- Kerjakan hingga terjadi kontradiksi atau proses tidak mengalami kemajuan :
  - Seleksi 2 klausa sebagai klausa parent.
  - Bandingkan (*resolve*) secara bersama-sama. Klausa hasil *resolve* tersebut dinamakan *resolvent.* Jika ada pasangan literal L dan ~L, eliminir dari *resolvent*.
  - Jika *resolvent* berupa klausa kosong, maka ditemukan kontradiksi. Jika tidak, tambahkan ke himpunan klausa yang telah ada.

RESOLUSI LOGIKA PROPOSISI (3/4)

- Contoh : Diketahui basis pengetahuan sebgai berikut :
  1. P
  2. $(P \wedge Q) \rightarrow R$
  3. $(S \vee T) \rightarrow Q$
  4. T

  Buktikan kebenaran R !

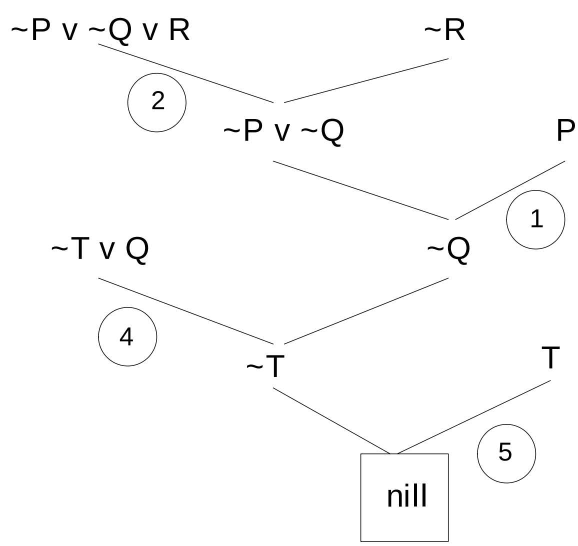
  Apabila kita ingin membuktikan kebenaran R dengan menggunakan resolusi, maka pertama-tama kita harus ubah dulu keempat fakta di atas menjadi bentuk CNF. Konversi ke CNF dapat dilakukan sebagai berikut :

Kalimat	Langkah-langkah	CNF
1. P	Sudah merupakan bentuk CNF	P
2. $(P \wedge Q) \rightarrow R$	- Menghilangkan implikasi $\sim(P \wedge Q) \vee R$ - Mengurangi lingkup negasi $(\sim P \vee \sim Q) \vee R$ - Gunakan assosiatif $\sim P \vee \sim Q \vee R$	$\sim P \vee \sim Q \vee R$
3. $(S \vee T) \rightarrow Q$	- Menghilangkan implikasi $\sim(S \vee T) \vee Q$ - Mengurangi lingkup negasi $(\sim S \wedge \sim T) \vee Q$ - Gunakan distributif $(\sim S \vee Q) \wedge (\sim T \vee Q)$	$(\sim S \vee Q)$ $(\sim T \vee Q)$
4. T	Sudah merupakan bentuk CNF	T

RESOLUSI LOGIKA PROPOSISI (4/4)

- Kemudian kita tambahkan kontradiksi pada tujuannya, R menjadi ~ R, sehingga fakta-fakta (dalam bentuk CNF) dapat disusun menjadi :
  1. P
  2. ~P ∨ ~Q ∨ R
  3. (~S ∨ Q)
  4. (~T ∨ Q)
  5. T
  6. ~R

  Dengan demikian resolusi dapat dilakukan untuk membuktikan R, sebagaimana terlihat pada gambar di bawah ini.



RESOLUSI LOGIKA PREDIKAT (1/4)

- Resolusi predikat merupakan suatu teknik pembuktian yang lebih efisien sebab fakta-fakta yang akan dioperasikan terlebih dahulu dibawa ke bentuk standar yang sering disebut dengan nama klausa.
- Pembuktian suatu pernyataan menggunakan resolusi ini dilakukan dengan cara menegasikan pernyataan-pernyataan tersebut, kemudian dicari kontradiksinya dari pernyataan-pernyataan yang sudah ada.
- Algoritma konversi ke bentuk klausa :
  1. Eliminir a → b menjadi $\sim a \vee b$
  2. Reduksi skope dari ~ sebagai berikut :
     $\sim(\sim a \vee b) \equiv \sim a \wedge \sim b$
     $\sim(\sim a \wedge b) \equiv \sim a \vee \sim b$
     $\sim\forall x : P(x) \equiv \exists x : \sim P(x)$
     $\sim\exists x : P(x) \equiv \forall x : \sim P(x)$
  3. Standarisasi variabel sehingga semua qualifier ($\forall$ dan $\exists$) terletak pada satu variabel yang unik.
     $\forall x : P(x) \vee \forall x : Q(x)$ menjadi
     $\forall x : P(x) \vee \forall y : Q(y)$
  4. Pindahkan semua qualifier ke depan tanpa mengubah urutan relatifnya.
  5. Eliminasi qualifier "$\exists$"
     $\forall x : \exists y : P(y,x)$ menjadi
     $\forall x : P(S(x),x)$
  6. Buang semua prefiks qualifier "$\forall$"
  7. Ubah menjadi *conjuction of disjunction*
     $(a \wedge b) \vee c \equiv (a \vee b) \wedge (a \vee b)$
  8. Bentuk klausa untuk tiap-tiap bagian konjungsi
  9. Standarisasi variabel di tiap klausa.

RESOLUSI LOGIKA PREDIKAT (2/4)

- Resolusi pada logika predikat pada dasarnya sama dengan resolusi pada logika proposisi, hanya saja ditambahkan dengan unifikasi. Pada logika predikat, prosedur untuk membuktikan pernyataan P dengan beberapa pernyataan F yang telah diketahui, dengan menggunakan **resolusi** dapat dilakukan algoritma sebagai berikut :
- Konversikan semua proposisi F ke bentuk klausa
- Negasikan P dan konversikan hasil negasi tersebut ke bentuk klausa. Tambahkan ke himpunan klausa yang telah ada pada langkah 1.
- Kerjakan hingga terjadi kontradiksi atau proses tidak mengalami kemajuan :
  1. Seleksi 2 klausa sebagai klausa parent.
  2. Bandingkan (*resolve*) secara bersama-sama. Klausa hasil resolve tersebut dinamakan *resolvent*. Jika ada pasangan literal T1 dan ~T2 sedemikian sehingga keduanya dapat dilakukan unifikasi, maka salah satu T1 atau T2 tidak muncul lagi dalam *resolvent*. T1 dan T2 disebut sebagai *complementary literal*. Jika ada lebih dari 1 *complementary literal*, maka ahnya sepasang yang dapat meninggalkan *resolvent*.
  3. Jika *resolvent* berupa klausa kosong, maka ditemukan kontradiksi. Jika tidak, tambahkan ke himpunan klausa yang telah ada.

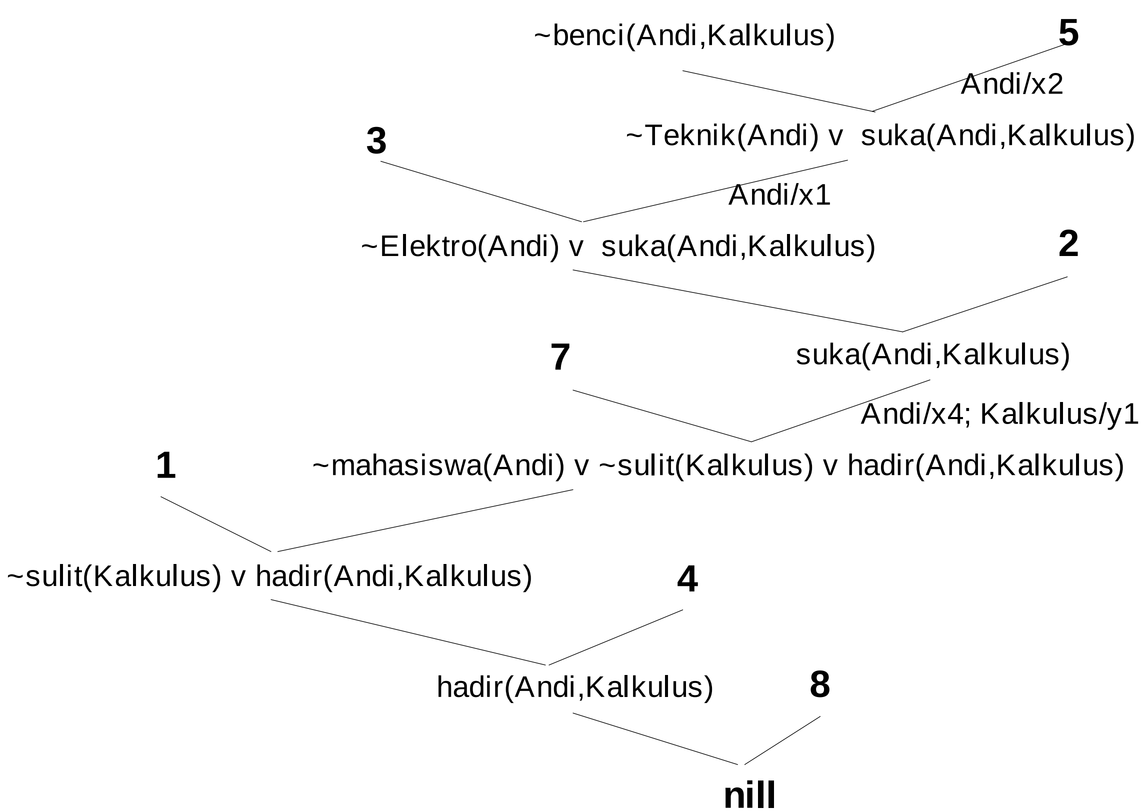
RESOLUSI LOGIKA PREDIKAT (3/4)

- Contoh : terdapat pernyataan-pernyataan sebagai berikut :
  - Andi adalah seorang mahasiswa
  - Andi masuk Jurusan Elektro
  - Setiap mahasiswa elektro pasti mahasiswa teknik
  - Kalkulus adalah matakuliah yang sulit
  - Setiap mahasiswa teknik pasti akan suka kalkulus atau akan membencinya
  - Setiap mahasiswa pasti akan suka terhadap suatu matakuliah
  - Mahasiswa yang tidak pernah hadir pada kuliah matakuliah sulit, maka mereka pasti tidak suka terhadap matakuliah tersebut.
  - Andi tidak pernah hadir kuliah matakuliah kalkulus.
- Kedelapan pernyataan di atas dapat dibawa ke bentuk logika predikat :
  1. mahasiswa(Andi)
  2. Elektro(Andi)
  3. $\forall$x: Elektro(x) → Teknik(x)
  4. sulit(Kalkulus)
  5. $\forall$x: Teknik(x) → suka(x, Kalkulus) $\vee$ benci(x, Kalkulus)
  6. $\forall$x:$\exists$y : suka(x,y)
  7. $\forall$x: $\forall$y: mahasiswa(x) $\wedge$ sulit(y) $\wedge$ ~hadir(x,y) → ~suka(x,y)
  8. ~hadir(Andi, Kalkulus)
- Kemudian dibuat dalam bentuk klausa :
  1. mahasiswa(Andi)
  2. Elektro(Andi)
  3. ~Elektro(x1) $\vee$ Teknik(x1)
  4. sulit(Kalkulus)
  5. ~Teknik(x2) $\vee$ suka(x2, Kalkulus) $\vee$ benci(x2, Kalkulus)
  6. suka(x3,f1(x3))
  7. ~mahasiswa(x4) $\vee$ ~sulit(y1) $\vee$ hadir(x4,y1) $\vee$ ~suka(x4,y1)
  8. ~hadir(Andi,Kalkulus)
- Akan dibuktikan apakah “Andi benci kalkulus” atau dapat ditulis :

  benci(Andi,Kalkulus)

RESOLUSI LOGIKA PREDIKAT (4/4)

- Pohon resolusi pada logika predikat untuk contoh di atas adalah :



PENALARAN SHALLOW dan CAUSAL (1/5)

- Sistem pakar menggunakan rantai inferensi, dimana *rantai yang panjang* merepresentasikan lebih banyak **causal** atau pengetahuan yang mendalam. Sedangkan **penalaran shallow** umumnya menggunakan kaidah tunggal atau inferensi yang sedikit.
- Kualitas inferensi juga faktor utama dalam penentuan kedalaman dan pendangkalan dari penalaran.
- **Shallow knowledge** disebut juga **experiment knowledge.**
- Contoh : Penalaran shallow

IF a car has
a good battery
good sparkplugs conditional elements
gas
good tires
THEN the car can move

PENALARAN SHALLOW dan CAUSAL (2/5)

- Pada penalaran shallow, tidak ada atau hanya terdapat sedikit pemahaman dari subjek, dikarenakan tidak ada atau hanya terdapat sedikit rantai inferensi.
- Keuntungan dari penalaran shallow adalah kemudahan dalam pemograman, yang berarti waktu pengembangan program menjadi singkat, program menjadi lebih kecil, lebih cepat dan biaya pengembangan menjadi murah.
- Penalaran causal disebut juga penalaran mendalam (**deep reasoning**), karena pemahaman yang mendalam diperoleh dari pemahaman rantai causal kejadian yang terjadi, atau dengan kata lain kita dapat memahami proses dari suatu abstrak yang disajikan.
- Frame dan jaringan semantik adalah contoh model yang menggunakan penalaran causal.

PENALARAN SHALLOW dan CAUSAL (3/5)

- Contoh :
  IF the battery is good
  THEN there is electricity
  IF there is electricity
  and the sparkplugs are good
  THEN the sparkplugs will fire
  IF the sparkplugs fire
  and there is gas
  THEN the engine will run
  IF the engine runs
  and there are is gas
  THEN the engine will run
  IF the engine runs
  and there are good tires
  THEN the car will move

- Penalaran causal cocok digunakan untuk operasi yang berubah-ubah dari sistem yang dibatasi oleh kecepatan eksekusi, memori dan peningkatan biaya pengembangan.
- Penalaran causal dapat digunakann untuk membangun model sistem nyata, seperti model yang dipakai untuk simulasi penggalian hipotesa penalaran pada tipe query "what if".
- Contoh : Dalam mengobati pasien, dokter dihadapkan pada jangkauan yang lebar dalam melakukan tes diagnosa untuk memverifikasi kejadian/penyakit secara cepat dan tepat.
- Karena kebutuhan akan penalaran causal meningkat, diperlukan kombinasi dengan kaidah penalaran satu shallow.
- Metode resolusi dengan refutation dapat digunakan untuk membuktikan apakah kaidah tunggal konklusi bernilai benar dari banyak kaidah (multiple rule).

PENALARAN SHALLOW dan CAUSAL (4/5)

- Contoh :

B=battery is good
C= car will move
E=there is electricity
F=sparkplugs will fire
G=there is gas
R=engine will run
S=sparkplugs are good
T=there are good tires

1. $B \wedge S \wedge G \wedge T \rightarrow C$
2. $B \rightarrow E$
3. $E \wedge S \rightarrow F$
4. $F \wedge G \rightarrow R$
5. $R \wedge T \rightarrow C$

- Langkah pertama di atas diaplikasikan pada resolusi refutation dengan menegasikan konklusi atau kaidah tujuan.

(1') $\sim(B \wedge S \wedge G \wedge T \rightarrow C) = \sim[\sim(B \wedge S \wedge G \wedge T) \vee C]$

Selanjutnya, setiap kaidah yang lain diekspresikan dalam disjunctive form menggunakan equivalesi seperti :

$p \rightarrow q \equiv \sim p \vee q$ dan $\sim(p \wedge q) \equiv \sim p \vee \sim q$

sehingga versi baru dari (2)-(5) menjadi :

(2') $\sim B \vee E$

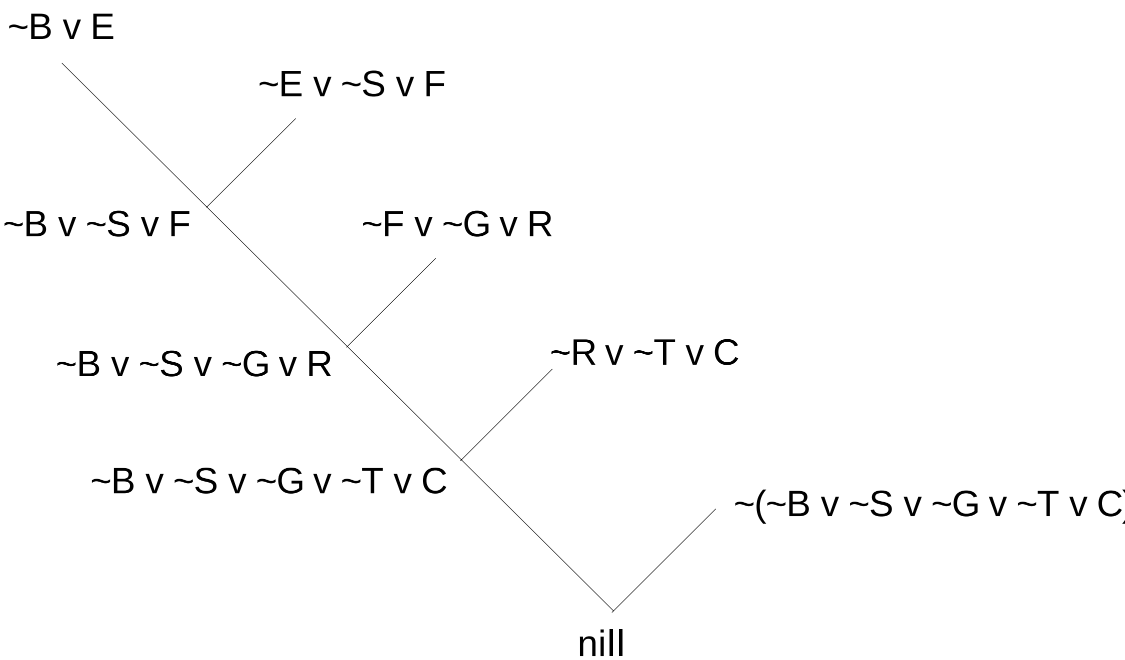
(3') $\sim(E \wedge S) \vee F = \sim E \vee \sim S \vee F$

(4') $\sim(F \wedge G) \vee R = \sim F \vee \sim G \vee R$

(5') $\sim(R \wedge T) \vee C = \sim R \vee \sim T \vee C$

PENALARAN SHALLOW dan CAUSAL (5/5)

- Pohon Resolusi Refutation-nya :



- Akar bernilai nill, menunjukkan kontradiksi. Sehingga melalui refutation dapat ditunjukkan konklusi asli (awal) :

$$B \wedge S \wedge G \wedge T \rightarrow C$$

adalah **teorema** dengan peran kontradiksi.

FORWARD CHAINING DAN BACKWARD CHAINING (1/)

- Chain (rantai) : perkalian inferensi yang menghubung-kan suatu permasalahan dengan solusinya.
- Forward chaining :
  - Suatu rantai yang dicari atau dilewati/dilintasi dari suatu permasalahn untuk memperoleh solusi.
  - Penalaran dari fakta menuju konklusi yang terdapat dari fakta.
- Backward chaining :
  - Suatu rantai yang dilintasi dari suatu hipotesa kembali ke fakta yang mendukung hipotesa tersebut.
  - Tujuan yang dapat dipenuhi dengan pemenuhan sub tujuannya.

FORWARD CHAINING DAN BACKWARD CHAINING (2/)

- Contoh rantai inferensi :

  gajah(x) → mamalia (x)

  mamalia(x) → binatang(x)

- Causal (sebab-akibat) Forward chain

gajah(clyde)

gajah(x) → mamalia(x)

mamalia(x) → binatang(x)

binatang(clyde)

- Explicit Causal chain

gajah(clyde)

unifikasi

implikasi gajah(clyde) → mamalia(clyde)

unifikasi

implikasi mamalia(clyde) →

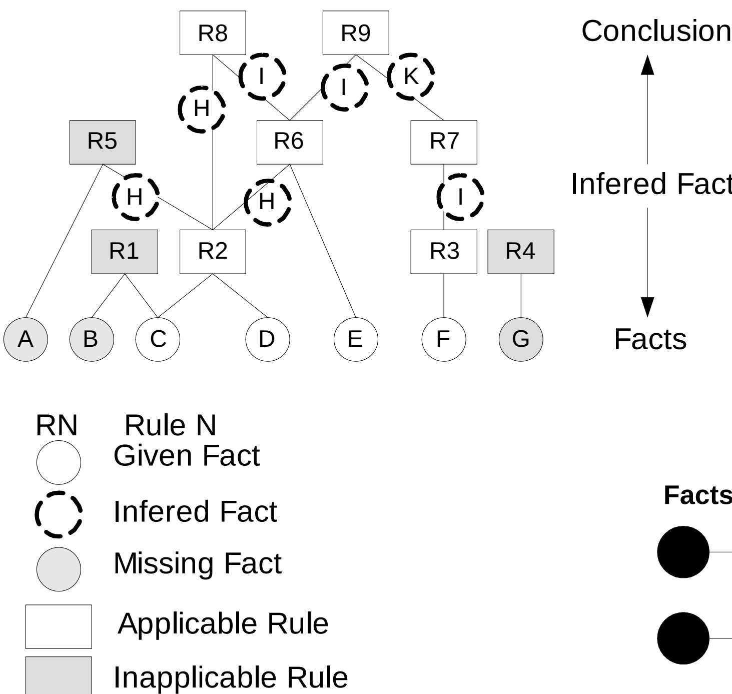
FORWARD CHAINING DAN BACKWARD CHAINING (3/)

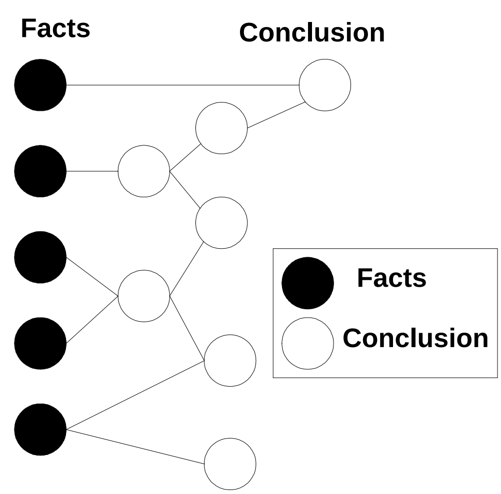
- Karakteristik Forward dan Backward chaining

Forward chaining	Backward chaining
Perencanaan, monitoring, kontrol	Diagnosis
Disajkan untuk masa depan	Disajikan untuk masa lalu
Antecedent ke konsekuen	Konsekuen ke antecedent
Data memandu, penalaran dari bawah ke atas	Tujuan memandu, penalaran dari atas ke bawah
Bekerja ke depan untuk mendapatkan solusi apa yang mengikuti fakta	Bekerja ke belakang untuk mendapatkan fakta yang mendukung hipotesis
Breadth first search dimudahkan	*Depth first search* dimudahkan
Antecedent menentukan pencarian	*Konsekuen* menentukan pencarian
Penjelasan tidak difasilitasi	Penjelasan difasilitasi

FORWARD CHAINING DAN BACKWARD CHAINING (4/)

- Forward Chaining

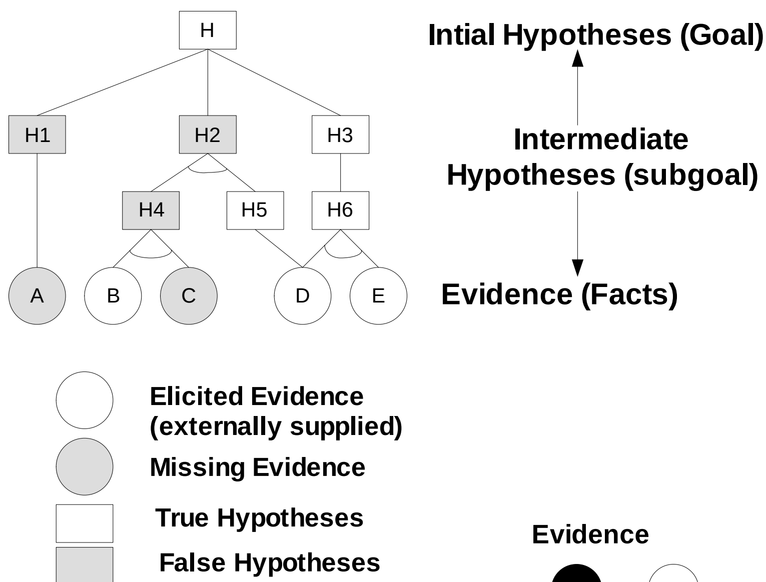


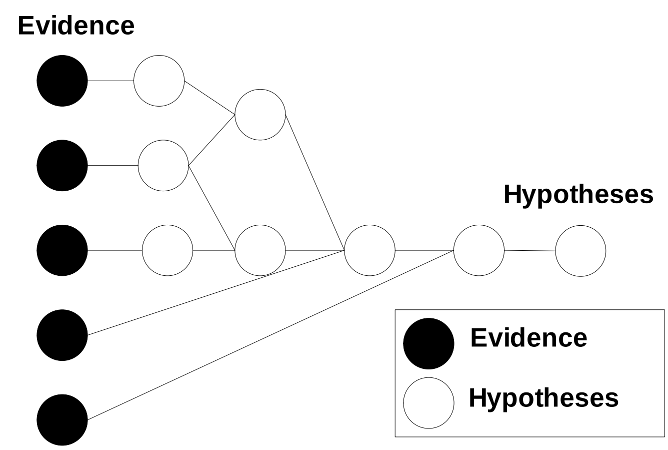


Melebar dan Tidak Dalam

FORWARD CHAINING DAN BACKWARD CHAINING (5/)

- Backward Chaining





Sempit dan Dalam

METODE LAIN DARI INFERENSI (1/2)

- ***ANALOGI***
  - Mencoba dan menghubungkan situasi lama sebagai penuntun ke situasi baru.
  - Contoh : diagnosis medical (gejala penyakit yang diderita oleh seorang pasien ternyata sama dengan gejala yang dialami pasien lain).
  - Pemberian alasan analogis berhubungan dgn induksi. Bila induksi membuat inferensi dari spesifik ke umum pada situasi yang sama, maka analogy membuat inferensi dari situasi yang tidak sama.
- ***GENERATE AND TEST***
  - Pembuatan solusi kemudian pengetesan untuk melihat apakah solusi yg diajukan memenuhi semua persyaratan. Jika solusi memenuhi maka berhenti yg lain membuat sollusi yg baru kemudian test lagi dst.
  - Contoh : Dendral, prog AM (artificial Mathematician), Mycin
- ***ABDUCTION/PENGAMBILAN***
  - Metodenya mirip dengan modus ponens

Abduction	Modus ponens
$p \rightarrow q$	$p \rightarrow q$
q	p
$\therefore p$	$\therefore q$

  - Bukan argument deduksi yang valid
  - Berguna untuk kaidah inferensi heuristik
  - Analogi,generate and test, abduction adalah metode bukan deduksi. Dari premise yg benar, metode ini tidak dapat membuktikan kesimpulan yg benar

METODE LAIN DARI INFERENSI (2/2)

Perbedaan Forward Chaining,

Backward Chaining dan Abduction

Inference	Start	Tujuan
FORWARD	Fakta	Kesimpulan yg harus mengikuti
BACKWARD	Kesimpulan tdk pasti	Fakta pendukung kesimpulan
ABDUCTION	Kesimpulan benar	Fakta yang dapat mengikuti

- ***NONMONOTONIC REASONING***
  - Adanya tambahan aksioma baru pada sistem logika berarti akan banyak teorema yang dapat dibuktikan.
  - Peningkatan teorema dengan peningkatan aksioma dikenal dengan ***sistem monotonik***
  - Suatu masalah dapat terjadi, jika diperkenalkan aksioma parsial atau komplit baru yang kontradikasi dengan aksioma sebelumnya.
  - Pada **sistem nonmonotonik**, tidak perlu adanya peningkatan teorema yang sejalan dengan peningkatan aksioma.