Representación de Conocimiento (2)

Carlos Hurtado L.
Depto de Ciencias de la Computación, Universidad de Chile

Clase Pasada

- Sintaxis y semántica de lógica proposicional
- Interpretación y modelo
- Consecuencia lógica
- Consecuencia sintáctica
  - Reglas de derivación
  - Demostración
  - Completitud y correctitud de un conjunto de reglas de derivación

Clase pasada: Propiedades de Reglas de Inferencia

Un conjunto de reglas de inferencia R es:

- Correcto ssi KB $\vdash_R$ α implica KB $\models$ α
- Completo ssi KB $\models$ α implica KB $\vdash_R$ α
- Ejemplo:
  - ¿qué regla es trivialmente completa para cálculo proposicional?
  - ¿qué regla es trivialmente correcta para CP?

Clase pasada: Ejemplo: Modus Ponens (MP)

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

- ¿Es MP correcto?
  - Sí. Ejercicio: demostrarlo
- ¿Es MP completo?

Clase pasada: Ejemplo: Modus Ponens (MP)

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

- ¿Es MP correcto?
  - Sí. Ejercicio: demostrarlo
- ¿Es MP completo?
  - No. Contraejemplo:
  - A partir de $\alpha \Rightarrow \beta, \beta \Rightarrow \delta$ no podemos demostrar $\alpha \Rightarrow \delta$ usando MP

Resolución

- Introducido por Alan Robin en 1965.
- Procedimiento de prueba correcto y completo para Cálculo de Proposiciones
- Se generaliza a Cálculo de Predicados de Primer Orden
- Base de motor de inferencia en Prolog y demostradores de teoremas, entre otros sistemas.
  - Carine, Gandalf, SPASS.

Ejemplo

Vía resolución podemos verifica si una concusión se deriva de un conjunto de premisas.

If Mary loves Pat, then Mary loves Quincy. If it is Monday, Mary loves Pat or Quincy. Prove that, if it is Monday, then Mary loves Quincy.

Cláusulas

- El procedimiento de Resolución requiere expresar sentencias en forma clausal
- Forma CNF: conjunción de cláusulas.

Cláusulas

- Def.: un literal es un átomo o la negación de un átomo. Por ejemplo P, $\neg P$, Q, $\neg Q$ son literales
- Def.: una cláusula es una disyunción de literales. Por ejemplo, ($\neg P \vee Q \vee$ R) es una cláusula.
  - Notación abreviada: ($\neg P$,Q,R)
  - Cláusula vacía () denota "Falso"

Cláusulas

- Una sentencia en forma CNF (Conjunctive Normal Form) es una conjunción de cláusulas. Ejemplo: $(\neg P \vee Q \vee R) \wedge (\neg Q \vee R) \wedge (\neg R \vee \neg Q)$
  - Notación abreviada: $\{(\neg P, Q, R), (\neg Q, R), (\neg R, \neg Q)\}$.
- Toda sentencia se puede transformar vía procedimiento simple a forma CNF.
  - Más adelante veremos cómo.

Regla de Resolución

$$\frac{\left(p, q_1, \ldots, q_n\right) \quad \left(\neg p, r_1, \ldots, r_n\right)}{\left(q_1, \ldots, q_k, r_1, \ldots, r_n\right)}$$

Resolvente

Otra forma de ver la regla de resolución:

$$\frac{\left(\neg q_1 \wedge \ldots \wedge \neg q_k \Rightarrow p\right) \quad \left(p \Rightarrow r_1 \vee \ldots \vee r_n\right)}{\left(\neg q_1 \wedge \ldots \wedge \neg q_k \Rightarrow r_1 \vee \ldots \vee r_n\right)}$$

Casos Especiales de Regla de Resolución

Modus Ponens

$$\frac{\begin{array}{l} p \Rightarrow q \\ p \end{array}}{q}$$

$$\frac{\begin{array}{l} (\neg p, q) \\ (p) \end{array}}{(q)}$$

Modus Tolens

$$\frac{\begin{array}{l} p \Rightarrow q \\ \neg q \end{array}}{\neg p}$$

$$\frac{\begin{array}{l} (\neg p, q) \\ (\neg q) \end{array}}{(\neg p)}$$

Encadenamiento

$$\frac{\begin{array}{l} p \Rightarrow q \\ q \Rightarrow r \end{array}}{p \Rightarrow r}$$

$$\frac{\begin{array}{l} (\neg p, q) \\ (\neg q, r) \end{array}}{(\neg p, r)}$$

Regla de Resolución: Múltiples resolventes

$$\frac{\left(P, Q, \neg R\right) \quad \left(P, W, \neg Q, R\right)}{\left(P, Q, W, \neg Q\right)}$$

En este caso tenemos dos resolventes, pero ambos son tautologías.

$$\frac{\left(P, Q, \neg R\right) \quad \left(P, W, \neg Q, R\right)}{\left(P, \neg R, W, R\right)}$$

Es decir, si tenemos más de un resolvente, éstos siempre son tautologías (equivalentes a “Verdadero”)

Regla de Resolución

- Es correcta
  - Ejercicio: demostrarlo.
- No es completa
  - Ejemplo: no podemos derivar (P ,R) a partir de (P) usando la regla de resolución.
  - Resolución no perite agregar nuevas variables proposicionales.

Refutación mediante Resolución

- Idea: usar regla de resolución para generar una refutación.
- Refutación: demostración de “Falso” a partir de un conjunto de sentencias.
- Falso se denota como la cláusula vacía: “()”.

Refutación: Ejemplo

- Sea S={ (P, Q), (¬Q, R), (¬P, S), (¬P, ¬S), (¬R)}.
- La siguiente es una refutación para S:

  (P, Q), (¬Q, R), (P,R), (¬P, S), (¬P, ¬S), (¬P), (R), (¬R), ()
- Como regla de resolución es correcta, en esta caso sabemos que S $\models$ (), que equivale a decir que S no es satisfacible.

Refutación mediante resolución

Recordar Teorema de Deducción:

$KB \models \alpha$ ssi $(KB \wedge \neg \alpha) \models$ Falso

Luego, verificaremos la primera consecuencia lógica anterior encontrando una refutación para $(KB \wedge \neg \alpha)$

.... y después demostraremos que este procedimiento es correcto y completo.

Refutación Mediate Resolución

¿ KB $\models$ α ?

2. S:= (KB $\wedge \neg \alpha$)
3. Convertir S en CNF
4. En forma iterativa, aplicar regla de resolución a cláusulas de S e ir añadiendo los resolventes a S. Parar cuando:
   - No tengamos nuevos resolventes, en este caso return(false)
   - Se generó la clásula vacía, en esta caso return(true)

Ejercicio

If Mary loves Pat, then Mary loves Quincy. If it is Monday, Mary loves Pat or Quincy. Prove that, if it is Monday, then Mary loves Quincy.

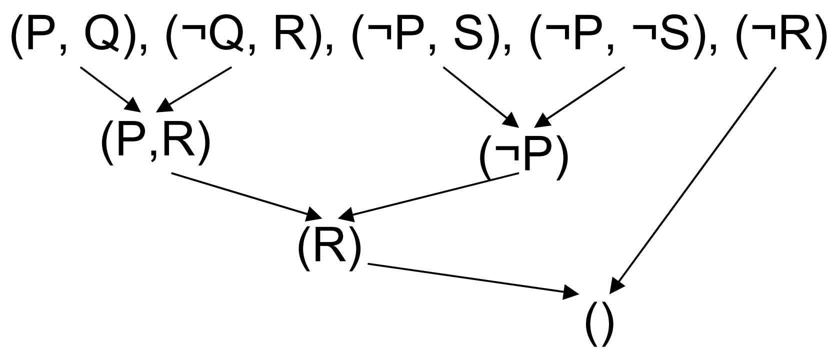
Ejemplo

If Mary loves Pat, then Mary loves Quincy. If it is Monday, Mary loves Pat or Quincy. Prove that, if it is Monday, then Mary loves Quincy.

1.	$\{\neg p, q\}$	Premisa
2.	$\{\neg m, p, q\}$	Premisa
3.	$\{m\}$	Objetivo Negado
4.	$\{\neg q\}$	Objetivo Negado
5.	$\{p, q\}$	3,2
6.	$\{q\}$	5,1
7.	$\{\}$	6,4

Ejemplo

- KB={ (P, Q), (¬Q, R), (¬P, S), (¬P, ¬S)}, α=R.
- Agregamos ($\neg\alpha$) a KB
- Aplicamos regla de resolución para construir una refutación:



- Lo que genera la siguiente refutación:
  (P, Q), (¬Q, R), (P,R), (¬P, S), (¬P, ¬S), (¬P), (R), (¬R),()
- Por lo tanto, KB $\models$ α.

Propiedades de Refutación Mediante Resolución

- Teorema: Regla de resolución es completa y correcta para generar refutaciones:
  - Resolución de $(KB \wedge \neg\alpha)$ genera la cláusula vacía ssi $KB \models \alpha$
  - Resolución de $(KB \wedge \neg\alpha)$ genera la cláusula vacía ssi $(KB \wedge \neg\alpha) \models$ Falso
  - La sentencia $(KB \wedge \neg\alpha)$ no es satisfacible ssi existe una refutación (mediante resolución) desde $(KB \wedge \neg\alpha)$.

¿Por qué funciona?

- Supongamos que KB $\models\ \alpha$
- La regla de resolución aplicada directamente no siempre genera α:

  KB = { (a,b), ($\neg$a,b), (b,c) }

  α = (b,$\neg$c)

¿Por qué funciona?

- Supongamos que KB $\models$ α
- La regla de resolución aplicada directamente no siempre genera α:

  KB = { (a,b), ($\neg$a,b), (b,c) }

  α = (b,$\neg$c)
- Sin embargo, siempre genera una cláusula contenida en α.
- Ejemplo: Al aplicar regla de resolución al KB anterior obtenemos b

Algoritmos de Resolución

- Existen distintas formas de implementar el proceso de resolución.
- Estrategias de ordenamiento: definen el orden en que se realizan las resoluciones
    - Primero en Anchura
    - Primero en Profundidad
- Estrategias de refinamiento:
  - Imponen restricciones en las resoluciones a realizar:
    - Conjunto Soporte
    - Resolución Lineal
    - Resolución Unitaria

Estrategia Primero en Anchura

- Res(a,b):= cláusula que se obtienen de resolver a y b.
- Resolventes nivel 0,
  - R(0):= KB
- Resolventes nivel 1,
  - R(1):={Res(a,b) | a,b in R(0)} – R(0)
- …
- Resolventes nivel i+1,
  - R(i+1):={Res(a,b) | a en R(i) y b en R(k), k <= i} – (Union$_{j<i}$ R(j))

Estrategia Primero en Anchura (cont.)

- R(0):=KB, i:=1;
- Loop
  - Calcular R(i),
  - Si (() en R(i)) return(true)
  - Si (R(i)={}) return(false)
  - i:=i+1

Ejemplo: Primero en Anchura

$$\frac{\dfrac{\dfrac{(P,Q),(\neg Q,R),(\neg P,S),(\neg P,\neg S),(\neg R)}{(P,R),(Q,S),(Q,\neg S),(\neg Q),(\neg P)}}{(R),(Q),(S),(\neg S),(R,S)(R,\neg S),(P),(Q,\neg P)}}{....()}$$

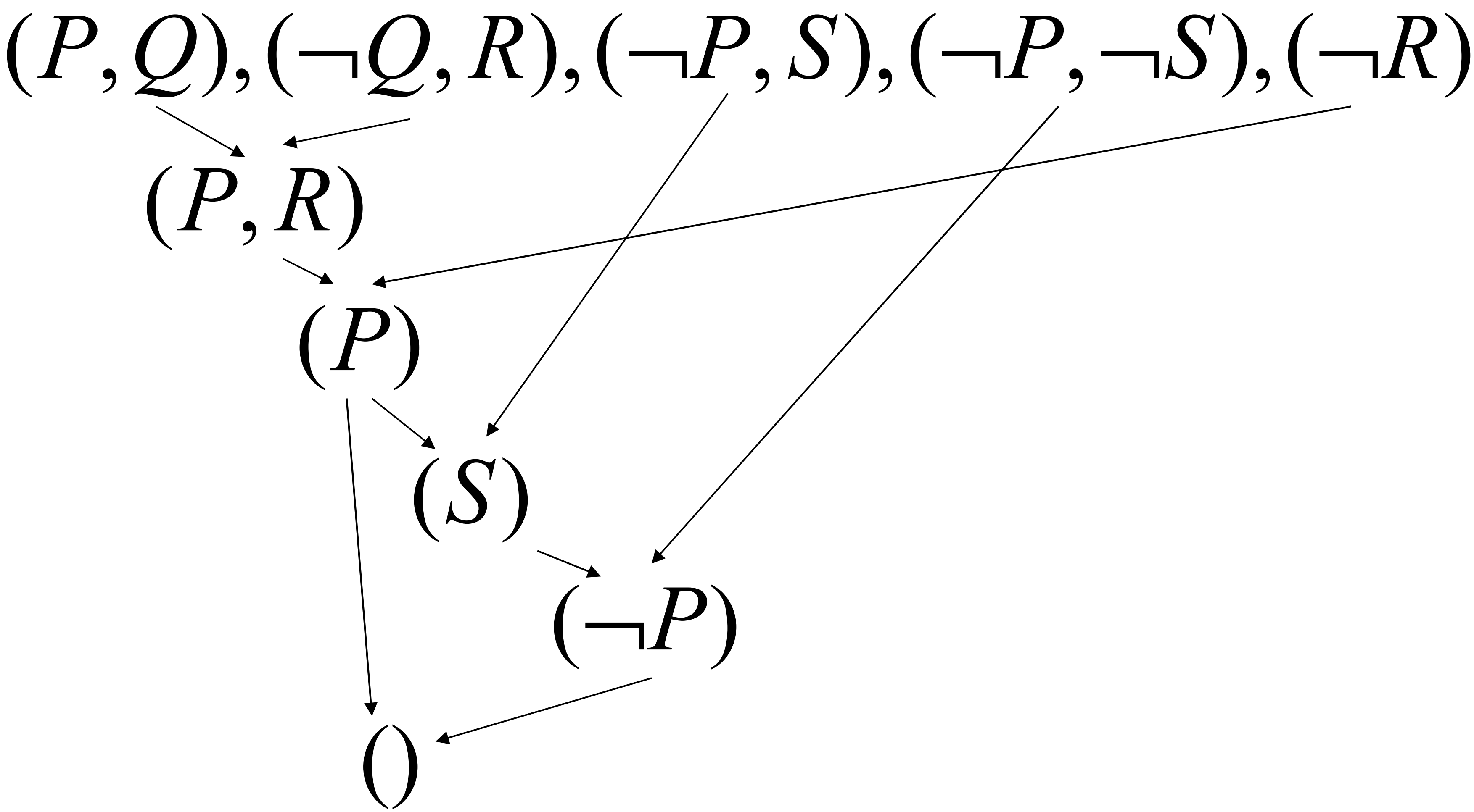
Complejidad de Resolución Primero en Anchura

- Dado un conjunto de m cláusulas con n variables proposicionales:
- Almacenamos cada cláusula como un arreglo de n bits
  - Aplicar regla de resolución sobre dos cláusulas: $O(n)$ pasos
- Para cada literal, guardamos la lista de cláusulas a las que pertenece.
  - Encontrar dos cláusulas a resolver: $O(n)$ pasos
- Tiempo total del algoritmo: $O(3^n n)$ pasos

Estrategia Primero en Profundidad

- Almacenamos la demostración como una lista de cláusulas .
- En cada paso resolvemos usando el resolvente generado en el paso anterior y una cláusula de la lista.
- Es necesario incluir mecanismo para detectar loops.
- Usualmente se pone una cota de profundidad.

Ejemplo: Primero en Profundidad



Estrategia Primero en Profundidad (sin detección de loops)

- Dado un conjunto de cláusulas S = (KB $\wedge \neg\alpha$)
- Resp:=Falso
- Para toda c en S
  - RPP(<c>);
- return(Resp)

RPP(L: lista de resolventes)

- Sea c cláusula inicial de L
- Para toda cláusula c' en S o en L que resuelva con c
  - c'':=Res(c',c)
  - Si (c''==())
    - Resp:=Verdadero
    - return() // aborta todos los llamados a RPP
  - Else
    - agregar c'' al inicio de L
    - RPP(L)

Loops

$$(\neg P, Q), (\neg Q, P), (\neg P, R), (R), (\neg P)$$

$$(\neg Q)$$

$$(\neg P)$$

$$(\neg Q)$$

$$(\neg P)$$

...

Primero en Profundidad vs. Primero en Anchura

- Ambas estrategias son completas (y correctas)
- Ventajas Primero en Profundidad:
  - Requiere menos memoria: sólo almacenamos resolventes en la lista L
- Desventaja Primero en Profundidad:
  - Podemos generar listas (o demostraciones parciales) muy largas.
  - Podemos llegar varias veces a la misma cláusula (redundancia).

Ejercicio

Encuentre una refutación para las siguientes cláusulas usando las dos estrategias de ordenación vistas.

$$A \vee C$$

$$A \vee \neg C \vee F$$

$$\neg A \vee E \vee F$$

$$\neg C \vee \neg E \vee F$$

$$\neg A \vee \neg F$$

$$C \vee F$$