Intelligenza Artificiale

Logica proposizionale: calcolo automatico

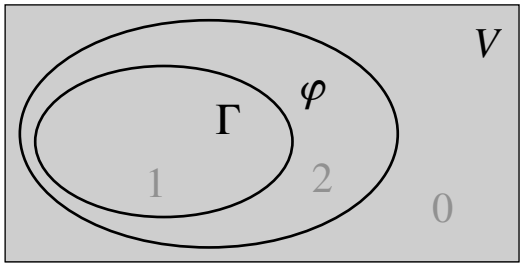
Marco Piastra

Parte 3

Calcolo automatico
Forme normali ed a clausole
Risoluzione e refutazione
Forward chaining
Metodo a tableau

Calcolo automatico

- L_P è **decidibile**
  - Esiste una **procedura effettiva** per stabilire se: $\Gamma \models \varphi$
  - Perché:
    - Assumiamo esista una procedura effettiva per determinare se una fbf è soddisfacibile
    - Si può dimostrare che $\Gamma \models \varphi$ sse $\Gamma \cup \{\neg\varphi\}$ è insoddisfacibile



(Nell'algebra dei sottoinsiemi di interpretazioni)

$\Gamma \models \varphi \Rightarrow v(\Gamma) \subseteq v(\{\varphi\})$ $\quad 1 \subseteq 1 + 2$

$v(\{\neg\varphi\}) = 0$

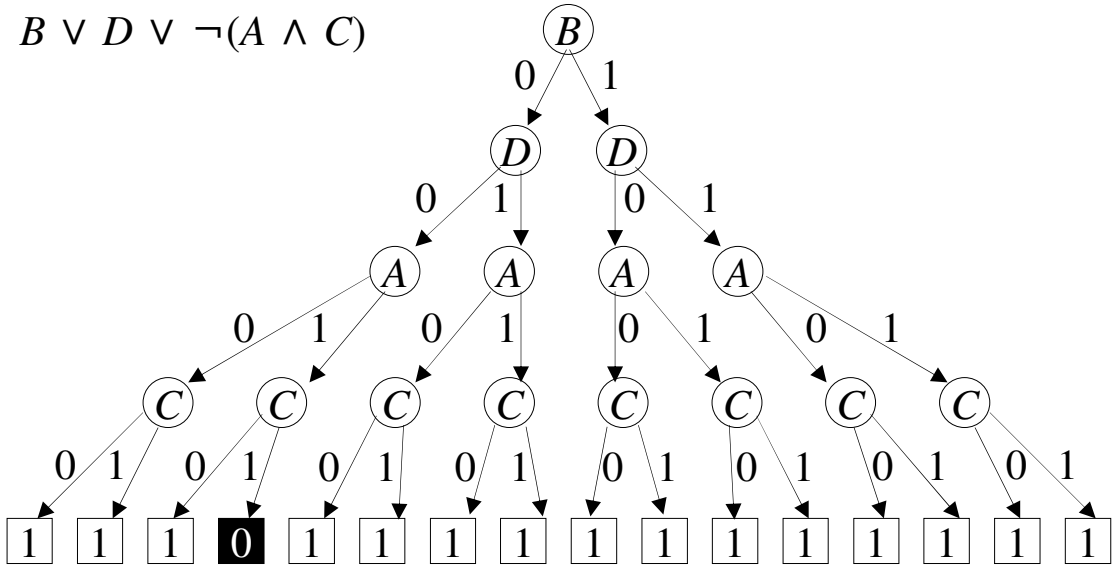
$v(\Gamma \cup \{\neg\varphi\}) = v(\Gamma) \cap v(\{\neg\varphi\})$

$v(\Gamma \cup \{\neg\varphi\}) = \varnothing$ $\quad 1 \cap 0 = \varnothing$

- Dato $\Gamma \cup \{\neg\varphi\}$, si consideri la fbf $\bigwedge(\Gamma \cup \{\neg\varphi\})$ (chiusura congiuntiva di tutte le fbf dell'insieme)
- L'insieme $\Gamma \cup \{\neg\varphi\}$ è soddisfacibile sse $\bigwedge(\Gamma \cup \{\neg\varphi\})$ lo è

Soddisfacibilità (in L_P)

- Esiste una **procedura effettiva** per stabilire se una fbf è soddisfacibile
  - Basta provare tutte le possibili assegnazioni di valore di verità
  - Esempio:



  - La procedura esaustiva ha complessità $O(2^n)$
- Il problema SAT è NP-completo
  - Qualsiasi problema NP-completo è traducibile in SAT in tempo polinomiale

Derivazione automatica

- La definizione di derivabilità non è traducibile in una procedura effettiva
  - Perché:
  - Volendo stabilire se $\Gamma \vdash \varphi$
    - Possiamo inserire una ad una nella dimostrazione le fbf $\psi \in \Gamma$ (OK)
    - Possiamo applicare la regola MP non appena possibile (OK)
    - NON possiamo inserire tutte le istanze degli assiomi Ax_n (KO)
      - A rigor di teoria, tali istanze sono infinite
  - Morale: *il problema sono gli assiomi*

- Esiste una procedura effettiva per la derivazione automatica?
    - (Intesa come calcolo simbolico)
    - In caso contrario possiamo solo automatizzare il metodo delle tavole di verità

Regola di risoluzione

- Una nuova regola di inferenza (calcolo simbolico)

  $\varphi \vee \chi, \neg\chi \vee \psi \vdash \varphi \vee \psi$

  - La fbf $\varphi \vee \psi$ viene anche detta *risolvente* di $\varphi \vee \chi$ e $\neg\chi \vee \psi$

  – La regola di risoluzione è corretta

φ	ψ	χ	$\varphi \vee \chi$	$\neg\chi \vee \psi$	$\varphi \vee \psi$
0	0	0	0	1	0
0	0	1	1	0	0
0	1	0	0	1	1
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	1

  – La regola del *modus ponens* è un caso particolare di *risoluzione*

  $\chi \rightarrow \psi, \chi \vdash \psi$ può essere riscritta come $\chi, \neg\chi \vee \psi \vdash \psi$

Forme normali

- In generale
  - Traduzione di una fbf qualsiasi in una forma logicamente equivalente

- **Forma normale congiuntiva** (FNC)
  - Una fbf in cui $\wedge$ appare solo al livello più esterno

    $\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n$ $\quad$ α_i sono fbf in cui compare solo $\vee$ e $\neg$ (solo davanti alle lettere)

    - Esempio:

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

- **Forma normale disgiuntiva** (FND)
  - Una fbf in cui $\vee$ appare solo al livello più esterno

    $\beta_1 \vee \beta_2 \vee \ldots \vee \beta_n$ $\quad$ β_i sono fbf in cui compare solo $\wedge$ e $\neg$ (solo davanti alle lettere)

    - Esempio:

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

Forma normale congiuntiva

- Traduzione automatica in FNC
  - Algoritmo:
    1) Si eliminano $\rightarrow$ e $\leftrightarrow$ in base alle regole di riscrittura
    2) Si muove $\neg$ all'interno
       con le "leggi De Morgan": $\neg(\varphi \wedge \psi) \Leftrightarrow (\neg\varphi \vee \neg\psi)$
       $\neg(\varphi \vee \psi) \Leftrightarrow (\neg\varphi \wedge \neg\psi)$
    3) Si distribuisce $\vee$
       proprietà del connettivo: $((\varphi \wedge \psi) \vee \chi) \Leftrightarrow ((\varphi \vee \chi) \wedge (\psi \vee \chi))$
  - Esempi:
    - $(\neg B \rightarrow D) \vee \neg(A \wedge C)$
      $B \vee D \vee \neg(A \wedge C)$ (eliminazione di $\rightarrow$)
      $B \vee D \vee \neg A \vee \neg C$ (De Morgan)
    - $\neg(B \rightarrow D) \vee \neg(A \wedge C)$
      $\neg(\neg B \vee D) \vee \neg(A \wedge C)$ (eliminazione di $\rightarrow$)
      $(B \wedge \neg D) \vee (\neg A \vee \neg C)$ (De Morgan)
      $(B \vee \neg A \vee \neg C) \wedge (\neg D \vee \neg A \vee \neg C)$ (distribuzione di $\vee$)

Forma a clausole

- In generale:
  - Traduzione di un insieme di fbf in un insieme equivalente in cui occorrono solo $\vee$ e $\neg$
- Traduzione in FC
  - Dato un insieme di fbf tradotte in FNC
  - Si sostituisce ciascuna fbf della forma $\alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n$ con l'insieme di fbf $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ (le congiunzioni diventano implicite)
  - Esempi:
    - $B \vee D \vee \neg A \vee \neg C$ (FNC e FC)
    - $(\neg B \vee \neg A \vee \neg C) \wedge (\neg D \vee \neg A \vee \neg C)$ (FNC)
      $\{(\neg B \vee \neg A \vee \neg C), (\neg D \vee \neg A \vee \neg C)\}$ (FC)
  - Notazione:
    - Le clausole si scrivono come insiemi di letterali (negativi e positivi)
    - $\{B, D, \neg A, \neg C\}$
    - $\{\{\neg B, \neg A, \neg C\}, \{\neg D, \neg A, \neg C\}\}$

Derivazioni: risoluzione

- Il problema visto in precedenza ("Giorgio è contento")

  $B \vee D \vee \neg(A \wedge C),\ B \vee C,\ A \vee D,\ \neg B \vdash D$

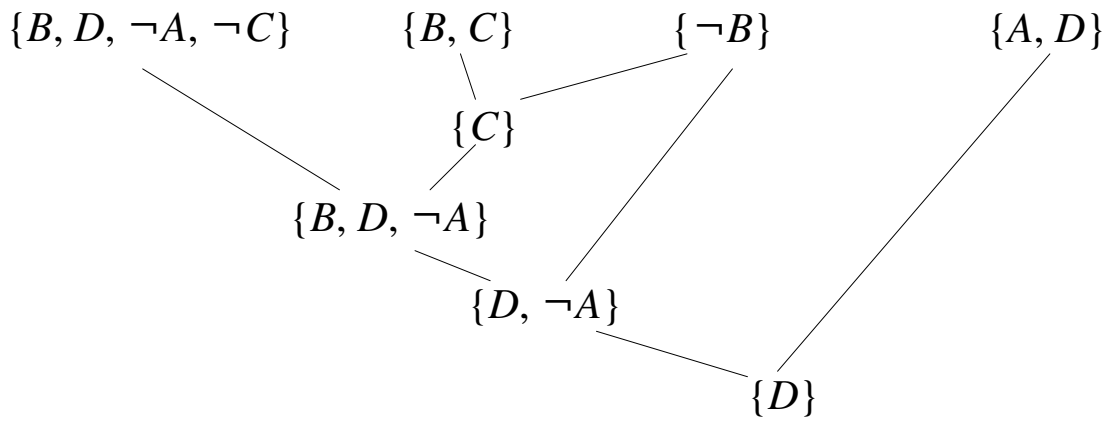
  - Riscrivendo le formule dell'ipotesi in FNC:

    $B \vee D \vee \neg A \vee \neg C, B \vee C,\ A \vee D,\ \neg B$

  - In FC:

    $\{B, D, \neg A, \neg C\}, \{B, C\}, \{A, D\},\ \{\neg B\}$

  - Applicando la regola di risoluzione (in forma binaria):



Risoluzione: ipotesi di metodo

- Descrizione:
  - Volendo stabilire se $\Gamma \vdash \varphi$
  - Si traducono Γ e φ in FNC e quindi in FC
  - Si applica la regola di risoluzione, in forma *binaria* (una coppia di letterali alla volta) ed in modo esaustivo
    - Si ottiene un insieme Σ di fbf in FC, $\Sigma = \{\psi : \Gamma \vdash \psi, per\ risoluzione\}$
    - L'algoritmo termina con successo se $\varphi \in \Sigma$
  - Vantaggi:
    - Niente assiomi
    - Una sola regola (risoluzione)
    - Procedura effettiva (algoritmo)
  - Svantaggi:
    - Come metodo di derivazione, **non è completo**
    - Contro-esempio:
    - $B \vee D \vee \neg A \vee \neg C, B \vee C,\ A \vee D,\ \neg B \vdash D \vee E$

      Identico al caso precedente, con $D \vee E$ al posto di D, ma la derivazione fallisce

Risoluzione per refutazione

- Descrizione:
  - Volendo stabilire se $\Gamma \vdash \varphi$
  - Si trasforma il problema in $\Gamma \cup \{\neg\varphi\}$ (refutazione della tesi φ)
    - Se $\Gamma \vdash \varphi$, $\Gamma \cup \{\neg\varphi\}$ è insoddisfacibile
  - Si traduce $\Gamma \cup \{\neg\varphi\}$ in FNC e quindi in FC
  - Si applica la regola di risoluzione, in forma ristretta
    - (Forma ristretta: si risolve un solo simbolo proposizionale alla volta)
    - L'algoritmo termina con successo quando viene derivata una **clausola vuota** $\{\ \}$
    - Diversamente, l'algoritmo termina quando la risoluzione non è più applicabile (fallimento)
- Vantaggi:
    - Niente assiomi
    - Una sola operazione (applicazione della regola di risoluzione)
    - Procedura effettiva (algoritmo)
    - Come metodo di derivazione, è completo

Derivazioni: risoluzione per refutazione

- Il contro-esempio visto in precedenza

  $B \vee D \vee \neg A \vee \neg C, B \vee C, A \vee D, \neg B \vdash D \vee E$

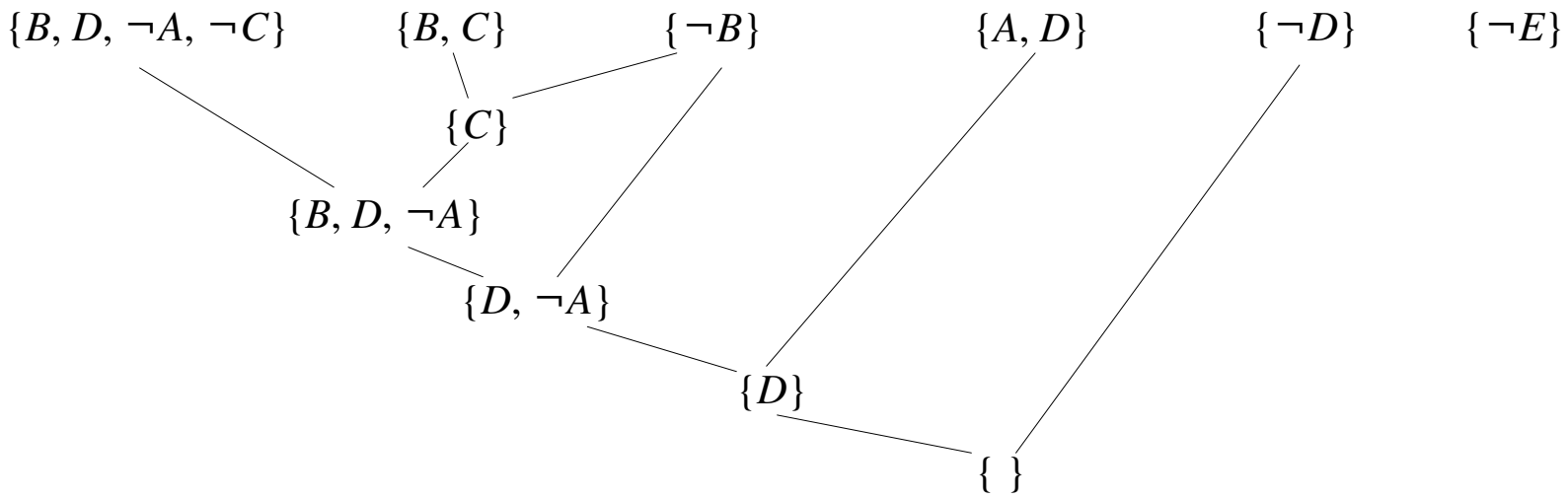
  - Refutando la tesi e riscrivendo il tutto in FNC:

    $B \vee D \vee \neg A \vee \neg C, B \vee C, A \vee D, \neg B, \neg D, \neg E$

  - In FC:

    $\{B, D, \neg A, \neg C\}, \{B, C\}, \{A, D\}, \{\neg B\}, \{\neg D\}, \{\neg E\}$

  - Applicando la regola di risoluzione:



Metodo della risoluzione per refutazione

- Il metodo della risoluzione per refutazione in L_P
  - Termina sempre
  - E` corretto: $\Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi$
  - E` completo: $\Gamma \vdash \varphi \Leftrightarrow \Gamma \models \varphi$

- Complessità computazionale:
  - Ha complessità $O(2^n)$ dove n è il numero delle *lettere*

Clausole di Horn

- Definizione
  - Una **clausola di Horn** è una fbf in FC
    in cui si ha al massimo un letterale in forma positiva
- Tre tipi particolari:
  - **Regole**: più letterali, uno solo positivo
    - Esempi: $\{B, \neg D, \neg A, \neg C\}, \{A, \neg B\}$ (traduzione: $(D \wedge A \wedge C) \rightarrow B, B \rightarrow A$)
  - **Fatti**: un solo letterale, positivo
    - Esempi: $\{B\}, \{A\}$
  - **Goal**: uno o più letterali, tutti negativi
    - Esempi: $\{\neg B\}, \{\neg A, \neg B\}$ (refutazione di: $B,\ A \wedge B$)

  - Ulteriore terminologia
    - Regole e fatti si dicono anche clausole definite (*definite clauses*)
    - I goal si dicono anche clausole negative (*negative clauses*)

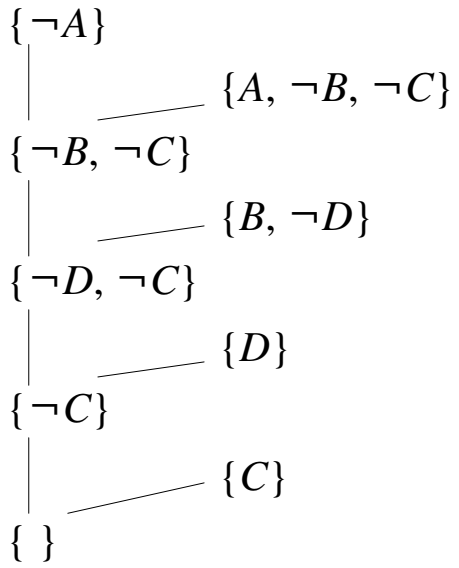
Traduzione in clausole di Horn

- Non tutte le fbf sono traducibili in clausole di Horn
  - Esempi (negativi):
    - $(A \wedge \neg B) \rightarrow C$
      $\neg(A \wedge \neg B) \vee C$ (eliminazione di $\rightarrow$)
      $\neg A \vee B \vee C$ (De Morgan)
    - $A \rightarrow (B \vee C)$
      $\neg A \vee B \vee C$ (eliminazione di $\rightarrow$)
  - Esempi (positivi):
    - $A \rightarrow (B \wedge C)$
      $\neg A \vee (B \wedge C)$ (eliminazione di $\rightarrow$)
      $(\neg A \vee B) \wedge (\neg A \vee C)$ (distribuzione di $\vee$)
      $(\neg A \vee B), (\neg A \vee C)$ (FC – si generano due regole)
    - $(A \vee B) \rightarrow C$
      $\neg(A \vee B) \vee C$ (eliminazione di $\rightarrow$)
      $(\neg A \wedge \neg B) \vee C$ (De Morgan)
      $(\neg A \vee C) \wedge (\neg B \vee C)$ (distribuzione di $\vee$)
      $(\neg A \vee C), (\neg B \vee C)$ (FC – si generano due regole)

Risoluzione SLD

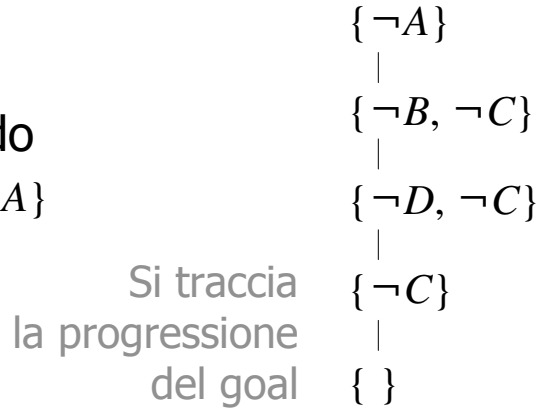
- *Linear resolution with Selection function for Definite clauses*

- Metodo particolare di risoluzione per refutazione
  - Si applica ad un set di clausole definite + un goal
- Descrizione
  - Applicazione sequenziale della regola di risoluzione (in forma ristretta) a partire dal goal usando una specifica *selection function*
  - Esempio:
    - Clausole definite: $\{C\}$, $\{D\}$, $\{B, \neg D\}$, $\{A, \neg B, \neg C\}$
    - Goal: $\{\neg A\}$
    - *Selection function*: ***leftmost subgoal first*** si risolve il letterale più a sinistra nel goal corrente
  - *Selection function*
    - Deve essere adeguata (*fair*): le condizioni teoriche sono abbastanza complesse
    - In pratica si usa (quasi) sempre la *leftmost subgoal first* (*fair* in L_P)



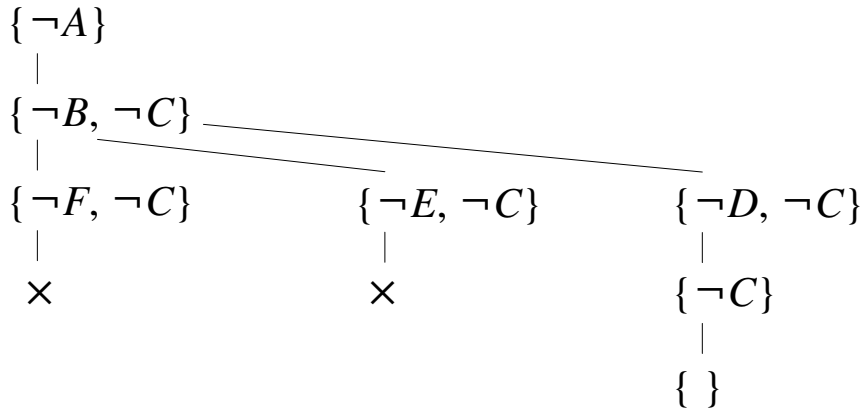
Alberi SLD

- Derivazioni SLD
  - Ciascun letterale nel goal è risolto in un solo modo
    - Esempio: $\{C\}, \{D\}, \{B, \neg D\}, \{A, \neg B, \neg C\}$ goal $\{\neg A\}$



Si traccia la progressione del goal

- Alberi SLD (=tutte le possibili derivazioni SLD)
  - Una diramazione per ciascun modo di risolvere un letterale nel goal
    - Esempio: $\{C\}, \{D\}, \{B, \neg F\}, \{B, \neg E\}, \{B, \neg D\}, \{A, \neg B, \neg C\}$ goal $\{\neg A\}$



Notare la strategia di ricerca *depth first*

Metodo della risoluzione SLD

- Il metodo della risoluzione SLD in L_P
  - Si applica solo alle clausole di Horn (clausole definite)
  - Termina sempre
  - E` corretto (per le clausole di Horn): $\Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi$
  - E` completo (per le clausole di Horn): $\Gamma \vdash \varphi \Leftrightarrow \Gamma \models \varphi$

- Complessità computazionale:
  - Ha complessità polinomiale (rispetto al numero di lettere ed alla dimensione delle fbf)
    - Rispetto ai metodi generali, si scambia la limitazione alle clausole di Horn con un'efficienza molto più elevata
- Limitazioni
  - Non tutti i problemi sono traducibili in clausole di Horn
    - 'Giorgio è contento' non lo è
    - Solo combinazioni di regole e fatti (con fatti – negati – come goal)

Forward chaining

- Per le regole e fatti come clausole di Horn, anche il *Modus Ponens* può bastare

- *Modus Ponens* Generalizzato (GMP)
  - Una regola di inferenza

    $\alpha_1,\ \alpha_2,\ \ldots\ \alpha_n, \alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n \rightarrow \beta \vdash \beta$ (MP originale: $\alpha, \alpha \rightarrow \beta \vdash \beta$)

- Descrizione:
  - Per stabilire se $\Gamma \vdash \varphi$
    - (Γ e φ in forma di clausole di Horn, Γ regole e fatti, φ fatto)
  - Si applica la regola GMP a Γ in modo esaustivo
    - Si ottiene un insieme Σ di fatti, $\Sigma = \{\psi : \Gamma \vdash \psi, per\ GMP\}$
    - L'algoritmo termina con successo se $\varphi \in \Sigma$

- Caratteristiche:
  - Niente assiomi
  - Una sola regola (GMP)
  - Niente refutazione o goal

Derivazioni: forward chaining

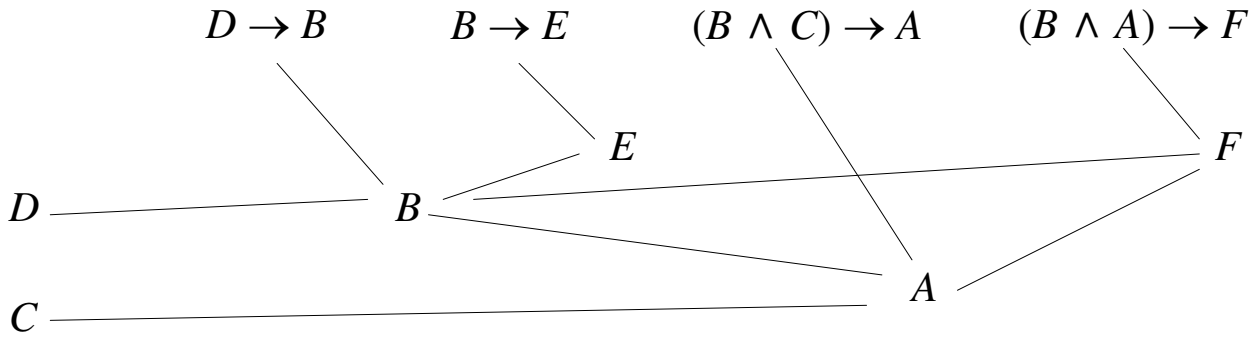
- Esempio

  $\{C\}, \{D\}, \{E, \neg B\}, \{B, \neg D\}, \{F, \neg B, \neg A\}, \{A, \neg B, \neg C\} \vdash \{A\}$

  - Riscrivendo il tutto in termini di $\wedge$ e $\rightarrow$:

    $C,\ D,\ B \rightarrow E,\ D \rightarrow B, (B \wedge A) \rightarrow F, (B \wedge C) \rightarrow A \vdash A$

  - Applicando il metodo del forward chaining:



Metodo del forward chaining

- Il metodo del forward chaining in L_P
  - Si applica alle clausole di Horn (clausole definite)
  - Termina sempre
  - E` corretto (per le clausole di Horn): $\Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi$
  - E` completo (per le clausole di Horn): $\Gamma \vdash \varphi \Leftrightarrow \Gamma \models \varphi$

- Complessità computazionale:
  - Ha complessità polinomiale (rispetto al numero di lettere ed alla dimensione delle fbf)

- Limitazioni
  - Non tutti i problemi sono traducibili in clausole di Horn

Backward e forward chaining

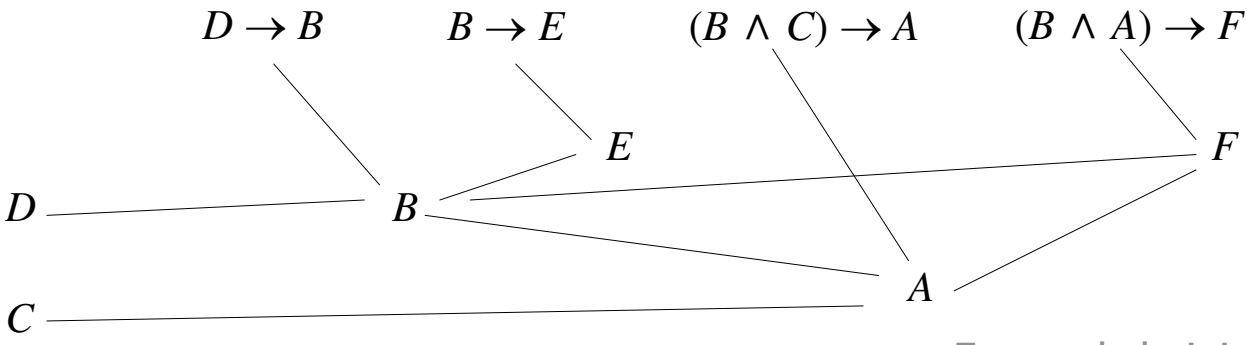
- I metodi della risoluzione SLD e del forward chaining a confronto

– Esempio:

$\{C\}, \{D\}, \{E, \neg B\}, \{B, \neg D\}, \{F, \neg B, \neg A\}, \{A, \neg B, \neg C\} \vdash \{A\}$

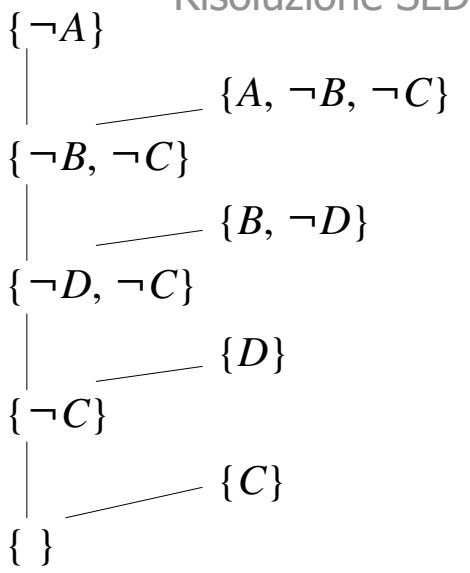
equivale a $C,\ D,\ B \rightarrow E,\ D \rightarrow B, (B \wedge A) \rightarrow F, (B \wedge C) \rightarrow A \vdash A$

- Caratteristiche
  - Risoluzione SLD (*backward chaining*)
    - Si parte dal goal, si applicano regole e fatti
  - Forward chaining
    - Si parte dai fatti e si applicano le regole



Forward chaining

Risoluzione SLD



Metodo a tableau

  - Anche 'tableau semantici' o a 'metodo a *tableaux*'
- E` un metodo per refutazione:
  - Per stabilire che $\Gamma \vdash \varphi$ si tenta di mostrare che $\Gamma \cup \{\neg\varphi\}$ è inconsistente
- Descrizione:
  - L'insieme $\Gamma \cup \{\neg\varphi\}$ (fbf in forma qualsiasi) diventa il *tableau* radice di un albero
    - Ciascun nodo (tableau) dell'albero rappresenta una **congiunzione** di fbf
  - Si espande il nodo iniziale applicando diverse regole di inferenza (9 per L_P)
    - Alcune regole espandono, altre provocano una diramazione
    - Ciascuna diramazione rappresenta una **disgiunzione** di tableau
  - Un ramo si chiude se il tableau contiene una contraddizione $\{\varphi, \neg\varphi\}$
    - $\Gamma \cup \{\neg\varphi\}$ è inconsistente (o insoddisfacibile) se tutti i rami sono **chiusi**
    - $\Gamma \cup \{\neg\varphi\}$ non è inconsistente se almeno un ramo che non si chiude

Regole di inferenza per i tableau

- Regole alfa (o di espansione)

(a1) $\neg(\neg\varphi)$ | φ

(a2) $\varphi \wedge \psi$ | φ, ψ

(a3) $\neg(\varphi \vee \psi)$ | $\neg\varphi, \neg\psi$

(a4) $\neg(\varphi \rightarrow \psi)$ | $\varphi, \neg\psi$

- Regole beta (o di biforcazione)

(b1) $\varphi \vee \psi$ → φ | ψ

(b2) $\neg(\varphi \wedge \psi)$ → $\neg\varphi$ | $\neg\psi$

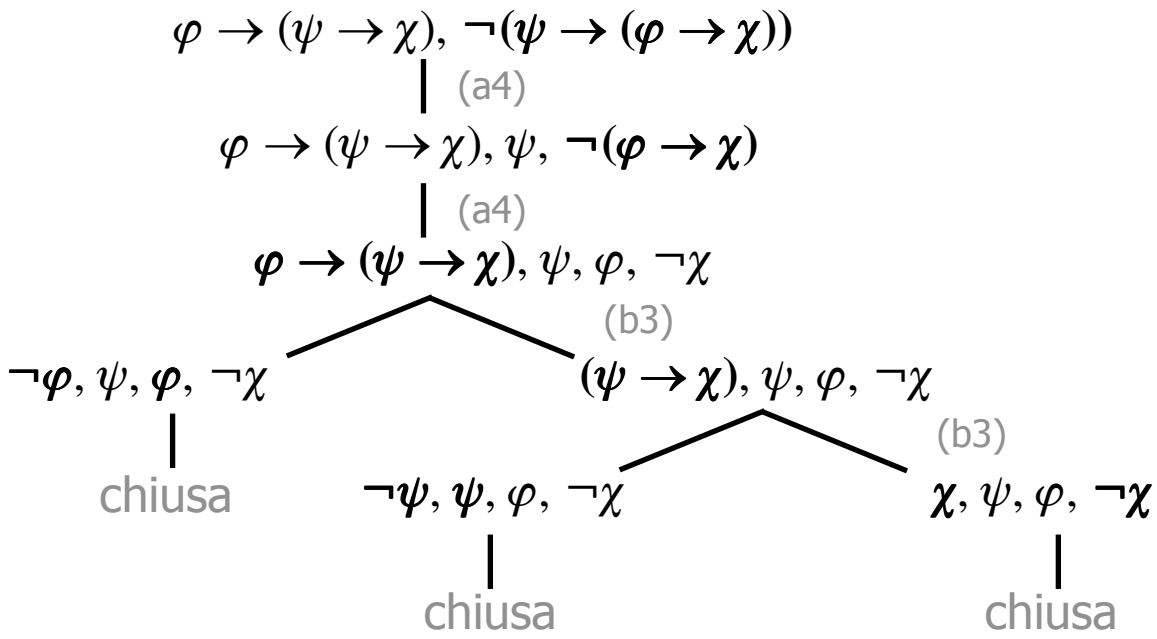
(b3) $\varphi \rightarrow \psi$ → $\neg\varphi$ | ψ

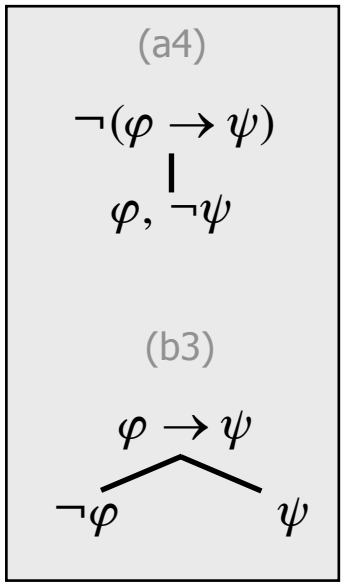
(b4) $\varphi \leftrightarrow \psi$ → $\neg\varphi, \neg\psi$ | φ, ψ

(b5) $\neg(\varphi \leftrightarrow \psi)$ → $\neg\varphi, \psi$ | $\varphi, \neg\psi$

Derivazione: metodo a tableau

- Obiettivo
  - $\varphi \to (\psi \to \chi) \vdash \psi \to (\varphi \to \chi)$
- Espansione





Algoritmo del metodo a tableau

- Procedura:
  - Dato il problema $\Gamma \vdash \varphi$
    1) Si assume $\Gamma \cup \{\neg\varphi\}$ come nodo *(tableau)* iniziale
    2) Su ciascun ramo, in modalità *depth-first*
    3) Se il nodo contiene solo letterali,
       se il nodo contiene una contraddizione, chiudere il ramo
       altrimenti terminare la procedura [fallimento]
    4) Se il nodo contiene formule composite:
       a) applicare le regole alfa
       b) applicare le regole beta
  - Notare: le regole alfa vengono applicate prima delle regole beta

Caratteristiche del metodo a tableau

- Il metodo a tableau in L_P
  - Termina sempre
  - E` corretto: $\Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi$
  - E` completo: $\Gamma \vdash \varphi \Leftrightarrow \Gamma \models \varphi$
- Complessità computazionale:
  - Ha complessità $O(2^n)$ dove n è il numero delle *lettere*
    - L'efficienza del metodo dipende in generale dalla struttura delle fbf, più che dal numero di letterali
    - In molti casi pratici, il metodo a tableau è estremamente efficace
- Leggibilità
  - Si applica a fbf in forma qualsiasi
- Generalità
  - Modificando opportunamente le regole di inferenza si possono costruire versioni del metodo anche per altre logiche (anche non classiche, come le logiche modali e multivalenti)