Azzolini Riccardo 2020-04-17

Calcoli a tableaux

1 Classificazione delle regole

Le regole del calcolo a tableaux vengono classificate in base alla loro struttura – in particolare, in base al numero di conclusioni:

- si dicono **α-regole** quelle che hanno un'unica conclusione:

$$\frac{\Gamma, A \wedge B}{\Gamma, A, B}\,\wedge \qquad \frac{\Gamma, \neg(A \vee B)}{\Gamma, \neg A, \neg B}\,\neg\vee \qquad \frac{\Gamma, \neg(A \to B)}{\Gamma, A, \neg B}\,\neg\to$$

- le **β-regole** sono le regole con due conclusioni:

$$\frac{\Gamma, \neg(A \wedge B)}{\Gamma, \neg A \mid \Gamma, \neg B}\,\neg\wedge \qquad \frac{\Gamma, A \vee B}{\Gamma, A \mid \Gamma, B}\,\vee \qquad \frac{\Gamma, A \to B}{\Gamma, \neg A \mid \Gamma, B}\,\to$$

- la regola per la doppia negazione viene trattata separatamente:

$$\frac{\Gamma, \neg\neg A}{\Gamma, A}\,\neg\neg$$

Ciò permette di parlare delle regole in modo più compatto, considerando le strutture in comune tra regole diverse, invece di dover trattare ciascuna regola singolarmente.

2 Classificazione delle formule

A partire da questa classificazione delle regole, si può introdurre un'analoga classificazione sulle formule:

- le **α-formule** sono le formule a cui si applicano le α-regole, cioè quelle che occorrono come formule principali nell'applicazione delle α-regole:

$$A \wedge B \qquad \neg(A \vee B) \qquad \neg(A \to B)$$

- le **β-formule** sono quelle che vengono destrutturate dalle β-regole:

$$\neg(A \wedge B) \qquad A \vee B \qquad A \to B$$

2.1 Ridotti delle α-formule

La seguente tabella riporta i ridotti di ciascun tipo di α-formula:

α-formula	Ridotti	
$A \wedge B$	A	B
$\neg(A \vee B)$	$\neg A$	$\neg B$
$\neg(A \to B)$	A	$\neg B$

Si osserva che, dal punto di vista semantico, ogni α-formula è equivalente alla congiunzione dei suoi ridotti:

$$A \wedge B \equiv A \wedge B$$
$$\neg(A \vee B) \equiv \neg A \wedge \neg B \qquad \text{(De Morgan)}$$
$$\neg(A \to B) \equiv A \wedge \neg B \qquad \text{(semantica dell'implicazione)}$$

2.2 Ridotti delle β-formule

I ridotti di ciascun tipo di β-formula sono riassunti nella seguente tabella:

β-formula	Ridotti	
$\neg(A \wedge B)$	$\neg A$	$\neg B$
$A \vee B$	A	B
$A \to B$	$\neg A$	B

Ogni β-formula è equivalente alla disgiunzione dei suoi ridotti:

$$\neg(A \wedge B) \equiv \neg A \vee \neg B \qquad \text{(De Morgan)}$$
$$A \vee B \equiv A \vee B$$
$$A \to B \equiv \neg A \vee B \qquad \text{(semantica dell'implicazione)}$$

3 Rango

Più avanti, sarà necessario dare delle definizioni sulla base della struttura induttiva delle formule analizzate dalle regole. In questo caso, però, non è particolarmente comodo trattare la struttura induttiva delle formule che si è utilizzata finora, perché, quando una regola destruttura una formula, in generale essa non introduce delle sottoformule

immediate della formula, bensì delle altre formule costruite a partire da tali sottoformule. Ad esempio, osservando la regola

$$\frac{\Gamma, \neg(A \wedge B)}{\Gamma, \neg A \mid \Gamma, \neg B}\ \neg\wedge$$

nelle sue conclusioni si trovano $\neg A$ e $\neg B$, e non le sottoformule $A \wedge B$, A o B di $\neg(A \wedge B)$.

Per ovviare a questo problema, si introduce sulle formule misura di complessità definita appositamente per gestire questa situazione, che tiene in considerazione come sono generati i ridotti a partire dalle formule analizzate dalle regole.

Definizione: Il **rango** $\mathrm{rg}(H)$ di una formula H è così definito:

- se H è un letterale, allora $\mathrm{rg}(H) = 1$;
- se $H = \neg\neg K$, allora
$$\mathrm{rg}(H) = \mathrm{rg}(K) + 1$$
- se H è una α-formula o una β-formula con ridotti H_1 e H_2, allora
$$\mathrm{rg}(H) = \mathrm{rg}(H_1) + \mathrm{rg}(H_2) + 1$$

Ad esempio, la formula $\neg(\neg p \wedge q)$ è una β-formula, i cui ridotti sono $\neg\neg p$ e $\neg q$, quindi:

$$\begin{aligned}\mathrm{rg}(\neg(\neg p \vee q)) &= \mathrm{rg}(\neg\neg p) + \mathrm{rg}(\neg q) + 1\\ &= \mathrm{rg}(p) + 1 + 1 + 1\\ &= 1 + 1 + 1 + 1\\ &= 4\end{aligned}$$

Definizione: Il rango $\mathrm{rg}(\Gamma)$ di un insieme finito di formule Γ è la somma dei ranghi delle formule in Γ:

$$\mathrm{rg}(\Gamma) = \sum_{H \in \Gamma} \mathrm{rg}(H)$$

Ad esempio, considerando $\Gamma = \{q, \neg(\neg p \wedge q), \neg\neg q\}$, si ha:

$$\begin{aligned}\mathrm{rg}(\Gamma) &= \mathrm{rg}(q) + \overbrace{\mathrm{rg}(\neg(\neg p \wedge q))}^{=4,\ \text{da prima}} + \mathrm{rg}(\neg\neg q)\\ &= 1 + 4 + \mathrm{rg}(q) + 1\\ &= 1 + 4 + 1 + 1\\ &= 7\end{aligned}$$

4 Costruzione dei tableaux

Un **tableau** per un insieme finito Γ di formule è un albero $\mathcal{T}$ che ha come radice un nodo etichettato da Γ, e i cui nodi sono etichettati con ("contengono") insiemi di formule costruite a partire dalle sottoformule di Γ.

Un tableau si dice **completo** se gli insiemi associati alle foglie contengono esclusivamente letterali.

Il tableau completo per Γ è costruito utilizzando un algoritmo che, a partire dalla radice Γ, effettua una serie di espansioni (corrispondenti all'applicazione di regole del calcolo), e termina quando le foglie dell'albero contengono solo letterali (ovvero quando non è più possibile applicare regole).

Nota: Nel seguito, si indicherà con N un generico nodo dell'albero, e con Γ_N l'insieme di formule che etichetta il nodo N.

4.1 Algoritmo di costruzione

Input: un insieme finito di formule Γ.
Output: un tableau completo per Γ.

`sia` $\mathcal{T}_0$ l'albero formato da un solo nodo N con etichetta $\Gamma_N = \Gamma$
$i = 0$ `//` i `conta i passi di costruzione`
`WHILE` (almeno una foglia di $\mathcal{T}_i$ contiene una formula composta) {
    `sia` N una foglia di $\mathcal{T}_i$ per cui Γ_N contiene almeno una formula composta
    `sia` H una formula composta in Γ_N
    $\mathcal{T}_{i+1} = \text{Espandi}(\mathcal{T}_i, N, H)$
    $i = i + 1$
}

Quest'algoritmo lavora per passi:

1. Nel passo 0, costruisce l'albero costituito solo dalla radice.
2. Successivamente, a ogni passo, seleziona una foglia che contiene almeno una formula composta, seleziona una formula composta all'interno di tale nodo, ed esegue un passo di espansione su tale formula, aggiungendo al nodo dei nuovi figli.
3. Quando non ci sono più foglie contenenti formule composte, il tableau è completo, e l'algoritmo termina.

Il passo di espansione è calcolato dalla funzione Espandi, che prende come argomenti

- l'albero $\mathcal{T}_i$ da espandere,

- il nodo N su cui agire,
- la formula H da decomporre,

e dà come risultato l'albero $\mathcal{T}_{i+1}$ ottenuto effettuando l'espansione. Essa è definita per casi, a seconda del tipo di formula H (doppia negazione, α-formula, o β-formula):

- Se $H = \neg\neg A$, allora l'albero $\mathcal{T}_{i+1}$ si costruisce aggiungendo al nodo N un figlio N', al quale è associato l'insieme ottenuto togliendo $\neg\neg A$ da Γ_N e aggiungendo A:

$$\Gamma_{N'} = (\Gamma_N \setminus \{\neg\neg A\}) \cup \{A\}$$

  Osservazioni:

  – Ricordando che l'insieme Γ_N contiene la formula $H = \neg\neg A$, si osserva che questo passo di espansione corrisponde a un'applicazione della regola $\neg\neg$:

$$\frac{\Gamma_N}{\Gamma_{N'} = (\Gamma_N \setminus \{\neg\neg A\}) \cup \{A\}}\,\neg\neg \quad \text{ovvero} \quad \frac{(\Gamma_N \setminus \{\neg\neg A\}),\ \neg\neg A}{(\Gamma_N \setminus \{\neg\neg A\}),\ A}\,\neg\neg$$

  – $\mathrm{rg}(\Gamma_{N'}) < \mathrm{rg}(\Gamma_N)$, in quanto $\mathrm{rg}(\neg\neg A) = \mathrm{rg}(A) + 1$. Infatti, togliendo $\neg\neg A$, il rango di Γ_N diminuisce di $\mathrm{rg}(\neg\neg A)$, e poi, aggiungendo A, aumenta di $\mathrm{rg}(A) < \mathrm{rg}(\neg\neg A)$.

- Se H è una α-formula con ridotti H_1 e H_2, allora l'albero $\mathcal{T}_{i+1}$ si costruisce aggiungendo a N un figlio N', la cui etichetta è la conclusione dell'α-regola che si applica all'α-formula:

$$\Gamma_{N'} = (\Gamma_N \setminus \{H\}) \cup \{H_1, H_2\}$$

  Anche in questo caso, $\mathrm{rg}(\Gamma_{N'}) < \mathrm{rg}(\Gamma_N)$, poiché $\mathrm{rg}(H) = \mathrm{rg}(H_1) + \mathrm{rg}(H_2) + 1$.

- Infine, se H è una β-formula con ridotti H_1 e H_2, l'albero $\mathcal{T}_{i+1}$ si costruisce aggiungendo al nodo N due figli, N' e N'', etichettati con le due conclusioni della β-regola che si applica ad H:

$$\Gamma_{N'} = (\Gamma_N \setminus \{H\}) \cup \{H_1\} \qquad \Gamma_{N''} = (\Gamma_N \setminus \{H\}) \cup \{H_2\}$$

  Si verifica anche in questo caso che $\mathrm{rg}(\Gamma_{N'}) < \mathrm{rg}(\Gamma_N)$ e $\mathrm{rg}(\Gamma_{N''}) < \mathrm{rg}(\Gamma_N)$, dato che $\mathrm{rg}(H) = \mathrm{rg}(H_1) + \mathrm{rg}(H_2) + 1$.

4.2 Caratteristiche dell'algoritmo

- L'algoritmo di costruzione *termina sempre*.

  Infatti, se così non fosse, verrebbe costruita una sequenza infinita di alberi:

  $$\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k, \ldots$$

  Siccome ognuno di questi alberi è un'espansione del precedente (lo si ottiene aggiungendo uno o due figli a una foglia dell'albero precedente), e siccome il grado degli alberi è finito[1] (quindi essi non possono crescere all'infinito "in orizzontale") la sequenza deve generare almeno un ramo infinito.

  Siano

  $$N_0, N_1, N_2, \ldots, N_k, \ldots$$

  i nodi sul ramo infinito. Come già osservato, Espandi$(\mathcal{T}_i, N, H)$ ha la proprietà di introdurre sempre figli con rango minore del nodo di partenza (il padre), quindi dovrebbe essere

  $$\mathrm{rg}(\Gamma_{N_0}) > \mathrm{rg}(\Gamma_{N_1}) > \mathrm{rg}(\Gamma_{N_2}) > \cdots > \mathrm{rg}(\Gamma_{N_k}) > \cdots$$

  ma ciò è impossibile, perché:

  - l'insieme Γ di partenza è finito, quindi $\mathrm{rg}(\Gamma)$ ha un valore finito;
  - per definizione, il rango è un valore positivo ($\mathrm{rg}(\Delta) > 0$ per ogni insieme di formule Δ).

  Allora, l'espansione può essere effettuata al massimo un numero finito di volte, ovvero l'algoritmo termina sicuramente.

- L'algoritmo di costruzione è *non deterministico*, cioè non specifica in modo deterministico come vengano effettuate certe scelte. In particolare, non indica come selezionare, a ogni passo:

  - la foglia N su cui operare, tra le diverse che contengono formule composte;
  - la formula composta H su cui operare, tra le diverse che possono comparire in Γ_N.

  Scelte diverse producono, in generale, alberi diversi.

  Osservazione: Conviene applicare per prime, finché possibile, le α-regole, che non biforcano l'albero, e poi le β-regole. Infatti, facendo la scelta opposta, bisognerebbe applicare le stesse α-regole in tutti i rami generati dalle β-regole, invece che una volta sola.

[1] Per la precisione, gli alberi generati dall'algoritmo hanno grado 2, cioè ogni nodo può avere al massimo 2 figli.