Problem set 1
Esercizio 1

Alessandro Straziota

Algorithm 1 `GreedyHorn`

INPUT: Una formula di Horn $\mathcal{F}$ su un insieme di variabili X
OUTPUT: Un'assegnazione di verità σ oppure `None`

Imposta $\sigma(x) =$ `False` per ogni variabile $x \in X$
while esiste una clausola $C \in$ IMP non soddisfatta da σ **do**
  Sia $C \in$ IMP una clausola non soddisfatta e sia x la variabile corrispondente all'unico letterale positivo di C
  Imposta $\sigma(x) =$ `True`
if Esiste una clausola $C \in$ NEG che non è soddisfatta da σ **then**
  Return `None`
else
  Return σ

Proof. Certamente se `GreedyHorn` restituisce un'assegnazione σ allora la formula $\mathcal{F}$ è soddisfacibile. Infatti per ritornare σ vuol dire che durante l'esecuzione dell'algoritmo si è prima usciti dal ciclo `WHILE` (quindi tutte le clausole $C \in IMP$ sono soddisfatte) e poi si è entrati nello statement `ELSE` (quindi anche tutte le clausole $C \in NEG$ sono soddisfatte) e dato che IMP e NEG bipartiscono le clausole di $\mathcal{F}$ vuol dire che σ soddisfa $\mathcal{F}$.

Quello che manca è invece mostrare che

> *se* `GreedyHorn` *ritorna* `None` *allora certamente $\mathcal{F}$ non è soddisfacibile.*

Il dubbio che l'affermazione non sia vera sorge dal pensiero che nel ciclo `WHILE` potrebbe accadere di porre a `TRUE` delle variabili di troppo, che in realtà non lo sarebbero dovute essere.

Supponiamo che $\mathcal{F}$ sia soddisfacibile e sia τ una qualunque assegnazione di verità che la soddisfa. Dato che durante tutto il corso dell'algoritmo non può che aumentare il numero di variabili per le quali $\sigma(x) =$ `TRUE`, si vuole dimostrare che non verranno assegnati più valori `TRUE` del dovuto. Più in generale si vuole mostrare che il numero di variabili $x \in X$ tali che $\sigma(x) = \texttt{TRUE}$ non è maggiore del numero di variabili tali che $\tau(x) = \texttt{TRUE}$.

σ stays ahead of τ.

Quindi per ogni $i = 1, 2, ...$ sia $y_i \in X$ la variabile posta a `TRUE` durante l'i-esima iterazione del ciclo `WHILE`: verrà mostrato per induzione su i che $\forall i, \tau(y_i) = \texttt{TRUE}$.

Base: Sicuramente alla prima iterazione le uniche clausole $C \in IMP$ che non sono soddisfatte da σ saranno della forma

$$(x)$$

Perciò per $i = 1$ avremo che $\tau(y_1) = \sigma(y_1) = \texttt{TRUE}$.

Ipotesi: Supponiamo che per $i > 1$ sia vero che $\tau(y_i) = \sigma(y_i)$.

Induzione: Supponendo che per $i > 1$ sia vero che $\tau(y_i) = \sigma(y_i) = \texttt{TRUE}$ si vuole dimostrare che l'ipotesi sia vera anche per $i + 1$. Perciò consideriamo una clausola $C \in IMP$ che durante l'iterazione $(i + 1)$-esima del ciclo `WHILE` non è soddisfatta da σ. Questa clausola potrà essere di due forme:

$$(x)$$

oppure

$$(x_1 \vee \overline{x_2} \vee ... \vee \overline{x_k})$$

Nel primo caso è banale osservare che se $\mathcal{F}$ è soddisfacibile allora necessariamente $\tau(y_{i+1}) = \texttt{TRUE}$, confermando quindi che $\tau(y_{i+1}) = \sigma(y_{i+1})$.
Il secondo caso invece non è così banale. Innanzitutto ricordando l'equivalenza $\overline{a} \vee b \equiv a \Rightarrow b$ potremmo riscrivere la clausola in questione come segue

$$(x_2 \wedge x_3 \wedge ... \wedge x_k) \Rightarrow x_1$$

A questo punto è più semplice verificare che se la clausola non è soddisfatta allora siamo nella situazione in cui

$$\forall j = 2, 3, ..., k[\sigma(x_j) = TRUE] \wedge \sigma(x_1) = FALSE$$

Sicuramente se siamo in questa situazione è giusto che $\forall j = 2, 3, ..., k[\sigma(x_j) = TRUE]$ dato che $\sigma(x_j) = \tau(x_j)$ (<u>per ipotesi induttiva</u>). Mentre partendo dall'ipotesi iniziale che $\mathcal{F}$ sia soddisfacibile e che τ sia una **qualunque** assegnazione di verità che la soddisfa, allora necessariamente sarà vero che $\tau(x_1) = \tau(y_{i+1}) = TRUE$. Dato che a quella iterazione sappiano che l'algoritmo si comporterà ponendo $\sigma(x_1) = TRUE$ possiamo quindi affermare che

$$\tau(y_{i+1}) = \sigma(y_{i+1})$$

Quindi è stato dimostrato che non è possibile che σ assegni alle variabili dei valori `TRUE` di troppo, smentendo il dubbio precedentemente sorto, e che quindi è possibile che l'algoritmo `GreedyHorn` restituisca il valore `TRUE` solamente qualora non esista una assegnazione che soddisfi $\mathcal{F}$

$\square$