Laboratorio di Programmazione
Descrizione del progetto

28 marzo 2008

1 Il problema della soddisfacibilità di espressioni booleane

Una espressione booleana è composta da operatori logici, variabili e parentesi. Una *interpretazione* di una espressione booleana consiste in una assegnazione dei valori di verità (TRUE o FALSE o in breve T e F) alle variabili che compaiono nell'espressione.

Una espressione è *soddisfacibile* se esiste una interpretazione data la quale l'espressione assume il valore TRUE. Un'espressione è *insoddisfacibile* se, per ogni interpretazione, l'espressione assume il valore FALSE.

Due espressioni sono *equivalenti* se assumono lo stesso valore di verità per ogni interpretazione. Ogni espressione logica può essere trasformata in una espressione equivalente che utilizza solamente gli operatori logici AND ($\wedge$), OR ($\vee$) e NOT ($\neg$), dove l'operatore di complemento NOT può essere applicato solamente a variabili e non a formule complesse quali $\neg\,(a \vee b)$. A titolo di esempio, l'espressione $\neg(a \Rightarrow b)$ (dove $\Rightarrow$ rappresenta l'operatore logico IMPLIES) è equivalente all'espressione $(a \wedge \neg b)$.

a	b	$\neg\,(a \Rightarrow b)$	$(a \wedge \neg b)$	$(a \Rightarrow b) \wedge (a \wedge \neg b)$	$(a \Rightarrow b) \Rightarrow (\neg b \Rightarrow \neg a)$
F	F	F	F	F	T
F	T	F	F	F	T
T	F	T	T	F	T
T	T	F	F	F	T

Figura 1: esempio di tabella di verità

Chiameremo *letterale* una variabile o una variabile negata (e.g., a e $\neg b$ sono entrambi letterali) e *clausola* una disgiunzione di letterali (e.g., $a \vee \neg b \vee \ \neg c$ è una clausola). Una espressione è in *forma normale congiuntiva* (CNF) se è composta unicamente da congiunzioni di clausole. Ad esempio l'espressione

$$(\neg a \vee \ b \vee \ c) \wedge \ (\neg b \vee \ c) \wedge \ (\neg c)$$

è in CNF. Ogni espressione booleana può essere trasformata in una espressione equivalente in CNF.

Data un'espressione booleana, il problema della soddisfacibilità booleana (SAT) richiede di determinare se tale espressione è soddisfacibile o insoddisfacibile.

2 SAT Solver

Un SAT-solver è un programma data in input una formula booleana F proporzionale (normalmente come CNF), decide se F é soddisfacibile e in caso positivo rilascia l'assegnamento che la soddisfa. Attualmente non si conosce alcun algoritmo che nel caso peggiore risolve questo probelma efficientemente. Tuttavia esistono classi di SAT-solver, basati su un algoritmo chiamato DPLL, che consentono di risolvere il probelma dell soddisfcibilitá su moltissimi esempi a carattere pratico. Si ricorda che probelmi come la verifica dll'hardware, la verifica del software, l'inferenza di decisioni, e molti altri problemi sono facilemtne riconducibil al probelma della soddifisfaciblitá.

Il corso di laboratorio prevede lo sviluppo di un SAT-Solver guidato da conflitti, basato sull'algoritmo DPLL. Il Solver deovrá essere in grado di leggere un file in formato DIMACS (vedi specifiche primo modulo) contenente un espressione booleana in CNF.

La struttura fondamentale del Solver può essere descritta ad alto livello dal ciclo seguente:

```
1:  loop
2:    propagate() {Propagazione delle assegnazioni di valori di verità alle
      variabili determinate da clausole unitarie}
3:    if (non ci sono conflitti) then
4:      if (tutte le variabili sono state assegnate) then
5:        return SODDISFACIBILE
6:      else
7:        decide() {Assegna un valore di verità ad una variabile libera}
8:    else
9:      analyze() {Verifica il conflitto ed aggiungi un clausola di conflitto}
10:     if (è presente un conflitto al livello radice) then
11:       return INSODDISFACIBILE
12:     else
13:       backtrack() {Annulla le assegnazioni fatte fino a rendere la
          clausola di conflitto unitaria}.
```

L'algoritmo terminerá quando trova un assegnamento che soddisfa la formula, oppure non appena tutti gli assegnamenti sono stati provati e la formula non é mai risultata soddisfacibile.

La procedura *propagate* consetirá di propagare informazioni immediate come per esempio quella di forzare al valore di veritá vero il valore di una clausola formata da un unico letterale (si veda il Modulo 2).

Un *conflitto* si verifica quando una clausola assume il valore FALSE in seguito ad una assegnazione. Se un assegemento ha portato ad un conflitto, la procedura *analyze* si occupera di salvare alcune informazione su questo conflitto sotto forma di clsuole in una nuovo insieme di clausole chiamate clausole di conflitto, che informalmente codificheranno assegnamenti"vietati" da provare, perche giá precedentemente hanno prodotto la falsificazione di una clausola della formula in input.

Il progetto di laboratorio é pensato suddiviso in 3 moduli dei quali il primo si occuperá di leggere la formula dall'input e salvarla nelle opportune strutture dati, il secondo della procedura *propagate* e il terzo della procedura *analyze*.