Esercizi sulla sintesi per SSE

Tiziano Villa

Anno Accademico 2017-18

Nome e Cognome:
Matricola:
Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	X	
problema 2	X	
problema ...	X	
totale	X	

1. (a) Per una funzione a due valori, si risponda ai seguenti quesiti:
   i. Si definisca la nozione di copertura minima,
   ii. Si definisca la nozione di copertura minimale o irridondante,
   iii. Si definisca la nozione di copertura minimale rispetto al contenimento da parte di implicanti singoli.
   iv. Una copertura irridondante e' minimale rispetto al contenimento da parte di implicanti singoli ?
   v. Una copertura minimale rispetto al contenimento da parte di implicanti singoli e' irridondante ?

   Traccia di soluzione.

   Una copertura e' un insieme d'implicanti che contengono i punti dove la funzione vale 1 (eventualmente punti dove la funzione non e' specificata), e non contengono punti dove la funzione vale 0.

   Una copertura minima e' una copertura di cardinalita' minima.

   Una copertura e' minimale o irridondante se nessun suo sottoinsieme proprio e' una copertura.

   Una copertura e' minimale rispetto al contenimento da parte d'implicanti singoli se nessun implicante della copertura e' contenuto in alcun altro implicante della copertura.

   Una copertura irridondante e' minimale rispetto al contenimento da parte d'implicanti singoli.

   Una copertura minimale rispetto al contenimento da parte d'implicanti singoli non e' detto che sia irridondante, perche' un suo implicante puo' essere ridondante in quanto contenuto nell'unione di un insieme d'implicanti della copertura (ma non essere contenuto tutto in un implicante singolo).

(b) Si consideri la funzione a 3 ingressi e 2 uscite $f_1 = a'b'c' + a'b'c + ab'c + abc + abc'$, $f_2 = a'b'c + ab'c$.

Si mostrino le seguenti coperture, sia disegnando gli ipercubi e gl'implicanti, che scrivendo le coperture in forma matriciale:

i. copertura minima di cardinalita' 3,
ii. copertura irridondante di cardinalita' 4,
iii. copertura ridondante di cardinalita' 5 minimale rispetto a implicanti singoli.

Traccia di soluzione.

Esempio di copertura minima di cardinalita' 3.

```
00- 10
-01 11
11- 10
```

Esempio di copertura irridondante di cardinalita' 4.

```
00- 10
-01 01
1-1 10
11- 10
```

Esempio di copertura ridondante di cardinalita' 5 minimale rispetto a implicanti singoli.

```
00- 10
-01 01
-01 10
1-1 10
11- 10
```

Questa copertura e' ridondante perche' il terzo implicante e' contenuto nell'unione del primo e quarto implicante.

2. (a) Si descriva l'algoritmo per calcolare il complemento di una funzione Booleana con il paradigma di ricorsione monotona.

   (b) Si consideri la seguente funzione Booleana completamente specificata f rappresentata dalla copertura

$$F = xy + x'y'z + xy'z'.$$

   Si calcoli una copertura del complemento della funzione con il paradigma di ricorsione monotona.

3. Data la definizione di cofattore generalizzato come funzione incompletamente specificata
$$co(f, g) = (fg, \bar{g}, \bar{f}g),$$
si dimostri che che il cofattore di Shannon f_x e' una copertura di $co(f, x)$, cioe'
$$fx \subseteq f_x \subseteq fx + \bar{x};$$
inoltre si dimostri che e' l'unica copertura di $co(f, x)$ indipendente da x.

Si dimostrino le seguenti proprieta' del cofattore di Shannon:

(a) $xf_x + \overline{x}f_{\overline{x}} = f$

(b) $(f_x)_y = f_{xy}$

(c) $(fg)_x = f_x g_x$

(d) $(\overline{f})_x = \overline{(f_x)}$

Si dimostrino le seguenti proprieta' del cofattore generalizzato:

(a) $f = g\ co(f, g) + \overline{g}\ co(f, \overline{g})$

(b) $co(co(f, g), h) = co(f, gh)$

(c) $co(fg, h) = co(f, h)co(g, h)$

(d) $co(\overline{f}, g) = \overline{co(f, g)}$

4. (a) Si dimostri che $\overline{f} = x\overline{(f_x)} + \overline{x}\overline{(f_{\overline{x}})}$.

   (b) Si applichi il paradigma di ricorsione monotona per complementare la funzione:

$$f = xy'z' + wxy' + wy'z + wx'z + wyz + xyz + w'xy'z$$

   usando il processo di fusione indicato dal punto precedente.

   (c) Si complementi f usando la legge di De Morgan.

   (d) Si calcoli

$$U \sharp \mathcal{F},$$

   dove U e' il cubo universale e $\mathcal{F}$ e' la copertura data sopra

$$\mathcal{F} = \{xy'z', wxy', wy'z, wx'z, wyz, xyz, w'xy'z\}.$$

   L'operazione $\sharp$ tra due cubi α e β e' definita come segue:

$$\alpha \sharp \beta = \begin{cases} a_1.b_1' & a_2 & \dots & a_n \\ a_1 & a_2.b_2' & \dots & a_n \\ \dots & & & \\ a_1 & a_2 & \dots & a_n.b_n' \end{cases}$$

   Che cosa si ottiene ?

   (e) Si calcoli

$$U \sharp (U \sharp \mathcal{F}).$$

   Che cosa si ottiene ?

5. Si applichi il paradigma di ricorsione monotona per complementare la funzione:
$$f = ab'c' + ab'd + b'cd + a'cd + bcd + abc + ab'cd'$$

   (a) E' noto che l'operazione di fusione delle chiamate ricorsive si esegue con la seguente procedura MERGE_WITH_CONTAINMENT, dove data una funzione g s'indicano rispettivamente con $H_0 = \{\cup_{i=1}^{l} c_i\}$ e $H_1 = \{\cup_{j=1}^{k} d_j\}$ le coperture di $g_{x'}$ e g_x:

   MERGE_WITH_CONTAINMENT(H_0, H_1) {
   $H_2 = \{c_j \mid \exists i \ s.t. c_j \subseteq d_i\} \cup \{d_j \mid \exists \ is.t.d_j \subseteq c_i\}$
   $H_0 = H_0 \setminus H_2$
   $H_1 = H_1 \setminus H_2$
   return$(x'H_0 + xH_1 + H_2)$
   }

   Un'alternativa alla procedura MERGE_WITH_CONTAINMENT e' la seguente procedura MERGE_WITH_COVER:

   MERGE_WITH_COVER(H_0, H_1) {
   $H_2 = \{c_j \mid c_j \subseteq \{\cup d_i\}\} \cup \{d_j \mid d_j \subseteq \{\cup c_i\}\}$
   $H_0 = H_0 \setminus H_2$
   $H_1 = H_1 \setminus H_2$
   return$(x'H_0 + xH_1 + H_2)$
   }

   Si ripeta il calcolo del complemento di
$$f = ab'c' + ab'd + b'cd + a'cd + bcd + abc + ab'cd'$$
   usando MERGE_WITH_COVER invece che MERGE_WITH_CONTAINMENT e si confrontino i risultati.
   Che cosa si puo' dire delle proprieta' delle coperture del complemento ottenute usando l'una o l'altra procedura di fusione ?
   Traccia. Che cosa si puo' dire circa l'irridondanza o la primalita' delle coperture ottenute ?

6. Sia f una funzione con copertura $F = w'x'y'z' + wx'z' + wx + w'x'yz$.

   (a) Si complementi f con il metodo di ricorsione monotona. Si mostrino i risultati ad ogni passo.

   Traccia di soluzione.

   Cofattorizzando con l'ordine w, x, y, z, si ottiene la seguente copertura del complemento $F_1' = wx'z + w'x + w'yz' + w'y'z$.

(b) Dati due cubi α e β, si definisca la seguente operazione $\alpha \,\tilde{\sharp}\, \beta$:

$$\alpha \,\tilde{\sharp}\, \beta = \begin{cases} a_1.b_1' & a_2 & \dots & a_n \\ a_1.b_1 & a_2.b_2' & \dots & a_n \\ \dots \\ a_1.b_1 & a_2.b_2 & \dots & a_n.b_n' \end{cases}$$

Si analizzi l'effetto di tale operazione e la si usi per calcolare il complemento di f, mostrando i passi del calcolo.

Traccia di soluzione.

Calcolando $U \,\tilde{\sharp}\, F$ (dove U e' l'universo su x, y, z, w), si ottiene la seguente copertura disgiunta del complemento $F_2' = wx'z + w'x + w'x'yz' + w'x'y'z$.

(c) Si usi la legge di De Morgan per ottenere il complemento di f. Si semplifichi tale risultato rispetto al contenimento per cubo singolo. Ci sono primi di $\overline{f}$ che mancano dalla lista cosi' ottenuta ? Questo fatto a quale congettura induce ?

Traccia di soluzione.

Si ottiene la seguente copertura del complemento $F_3' = w'x + w'yz' + w'y'z + wx'z + x'y'z$.

Tale copertura contiene tutti i primi di $\overline{f}$.

E' una proprieta' generale di questo procedimento di produrre tutti i primi del complemento.

7. (a) Si dimostri che p e' un primo di f se e solo se e' un cubo massimale (un cubo massimale non e' contenuto in nessun altro cubo) dell'insieme che contiene i seguenti cubi:

   i. $p = qx$, q e' un primo di f_x, $q \not\subseteq f_{\bar{x}}$

   ii. $p = r\bar{x}$, r e' un primo di $f_{\bar{x}}$, $r \not\subseteq f_x$

   iii. $p = qr$, q e' un primo di f_x, r e' un primo di $f_{\bar{x}}$

   (b) Si dimostri il teorema precedente anche nel caso in cui la terza clausola iii. sia sostituita da

   iii. $p = \text{CONSENSO}(qx, r\bar{x})$, q e' un primo di f_x, r e' un primo di $f_{\bar{x}}$.

   L'operazione CONSENSO tra due cubi α e β e' definita come segue:

$$\text{CONSENSO}(\alpha, \beta) = \begin{cases} a_1 + b_1 & a_2.b_2 & \dots & a_n.b_n \\ a_1.b_1 & a_2 + b_2 & \dots & a_n.b_n \\ \dots \\ a_1.b_1 & a_2.b_2 & \dots & a_n + b_n \end{cases}$$

   (c) Si consideri il risultato del calcolo $U \sharp (U \sharp \mathcal{F})$, eseguito per l'esercizio 2. (e). Data una qualsiasi copertura $\mathcal{F}$ di una funzione f, si puo' dimostrare un'affermazione generale sul risultato del calcolo $U \sharp (U \sharp \mathcal{F})$?

8. Date le funzioni

$$
\begin{aligned}
f_1 &= \overline{a}\overline{b}c + \overline{a}b\overline{c} + \overline{a}\overline{b}\overline{c} \\
f_2 &= \overline{a}bc + \overline{a}\overline{b}c + abc + \overline{a}b\overline{c}
\end{aligned}
$$

   (a) Si generino tutti i primi di f_1 usando la procedura monotona ricorsiva di generazione dei primi;

   (b) Si generino tutti i primi di f_2 usando la procedura monotona ricorsiva di generazione dei primi;

   (c) Si generi con l'algoritmo di tautologia la tavola di Quine-McCluskey ridotta di f_1;

   (d) Si generi con l'algoritmo di tautologia la tavola di Quine-McCluskey ridotta di f_2;

   Nota: in alternativa alla procedura monotona si possono calcolare i primi con l'algoritmo di espansione rispetto al complemento della funzione.

9. (a) Si generino tutti i primi della funzione multi-uscita (f_1, f_2) (dove f_1 e f_2 sono date nel problema precedente), usando l'algoritmo di espansione rispetto al complemento della funzione.

   (b) Si generi la tavola di Quine-McCluskey ridotta della funzione multi-uscita (f_1, f_2) del problema precedente:

$$f_1 = \overline{a}\overline{b}c + \overline{a}b\overline{c} + \overline{a}\overline{b}\overline{c}$$
$$f_2 = \overline{a}bc + \overline{a}\overline{b}c + abc + \overline{a}b\overline{c}$$

10. (a) Data la seguente funzione Booleana incompletamente specificata:

$$\begin{aligned} f &= wx'y' + wxz' \\ d &= xy'z + x'yz' \end{aligned}$$

dove d sono i punti dove non e' specificata, si trovino tutti i primi con il metodo di ricorsione monotona.

Traccia di soluzione.

Una decomposizione ricorsiva e' la seguente

```
                  wxyz
                  1002
                  1120
                  2101
                  2010
                   (g)
             x               !x
           1220             1202
           2201             2210
            (e)              (f)
       z      !z        y      !y
     2202   1222      2220   1222
      (a)    (b)       (c)    (d)
```

I seguenti primi sono associati ai vari stadi del processo:

- (a): $\overline{y}$
- (b): w
- (c): $\overline{z}$
- (d): w
- (e): i massimali in $\{\overline{y}z, w\overline{z}, w\overline{y}\}$, cioe' $\overline{y}z, w\overline{z}, w\overline{y}$
- (f): i massimali in $\{y\overline{z}, w\overline{y}, w\overline{z}\}$, cioe' $y\overline{z}, w\overline{y}, w\overline{z}$
- (g): i massimali in $\{wx\overline{y}, x\overline{y}z, wx\overline{z}, \overline{x}y\overline{z}, w\overline{x}\overline{y}, w\overline{x}\overline{z}, w\overline{y}, w\overline{y}\overline{z}, w\overline{y}z, wy\overline{z}, w\overline{z}\}$, cioe' $w\overline{y}, w\overline{z}, \overline{x}y\overline{z}, x\overline{y}z$

Nel fondere i primi di due sottoalberi non si dimentichi di eseguire i prodotti incrociati.

(b) Data la funzione Booleana a 3 ingressi e 2 uscite

$$\begin{aligned} f_1 &= xz' + x'z \\ f_2 &= yz' + x'z + xy \end{aligned}$$

e il cubo

```
x y z  f1f2
- 1 -  0 1
```

si calcoli se il cubo e' contenuto nella funzione riducendo il problema a una verifica di tautologia.

Traccia.

Si tratti la funzione a due uscite come una funzione a piu' valori e una sola uscita (dove le uscite diventano un nuovo ingresso a due valori).

Traccia della soluzione.

L'idea e' di usare il teorema per cui dati un cubo c e una copertura F, $c \subseteq F \Leftrightarrow F_c = 1$.

La funzione a piu' uscite data si puo' rappresentare con la seguente copertura F:

```
xyzw
1200
0210
2101
0211
1121
```

dove w e' la variabile ausiliaria binaria per rappresentare le uscite; inoltre si ha c

```
xyzw
2121
```

Allora si tratta di verificare se $F_c = 1$, dove F_c si riduce a:

```
xyzw
2202
0212
1222
```

Applicando la ricorsione monotona si vede che si ha una tautologia in tutte le foglie.

```
               xyzw
               2202
               0212
               1222
         x            !x
       2202         2202
       2222         2212
   tautologia      z      !z
                 2222  2222
                 taut. taut.
```

Percio' il cubo c dato e' contenuto nella funzione (f_1, f_2).

Per maggiore semplicita' si poteva osservare che essendo c contenuto banalmente in f_1 (come cubo vuoto nella prima uscita), il problema si poteva ridurre a verificare subito il contenimento solo per f_2 per le tre variabili d'ingresso (senza introdurre la variabile w per le uscite).

11. (a) Si consideri la seguente funzione non completamente specificata $\mathcal{F}$ il cui insieme di punti dove vale 1 e' rappresentato dalla copertura $F = wx'y' + wxz'$, e il cui insieme dei punti dove la funzione non e' specificata e' rappresentato dalla copertura $D = xy' + x'yz'$.

Si calcolino i primi della funzione con il paradigma di ricorsione monotona. Inoltre si ottenga una rappresentazione minima con il metodo di Quine-McCluskey.

Traccia di soluzione.

Si richiede di applicare il calcolo dei primi con il paradigma di ricorsione monotona decomponendo secondo Shannon la copertura e continuando fino a che alle foglie sono associate coperture monotone. Poi si risale per ottenere i primi delle coperture cofattorizzate.

I primi sono xy', $x'yz'$, wz', wy'.

(b) Si determini se la funzione completamente specificata f rappresentata dalla copertura $F = wxy + wyz' + w'x'y' + w'z + xy' + w'y$ e' una tautologia usando il metodo di ricorsione monotona, mostrando il procedimento di calcolo.

Traccia di soluzione.

Si richiede di verificare la tautologia con il paradigma di ricorsione monotona, applicando tutte le condizioni strutturali note per stabilire quando una copertura rappresenta la tautologia.

Si decomponga rispetto a w. Si ottiene una copertura monotona positiva in x e monotona negativa in z. Se si cofattorizza rispetto a $x'z$ si trova la funzione nulla. Percio' il cubo 1021 non e' coperto da F e la funzione non e' una tautologia.

12. (a) Si consideri la seguente funzione Booleana f rappresentata dalla copertura

$$F = v'xyz + v'w'x + v'x'z' + v'wxz + w'yz' + vw'z + vwx'z.$$

Si calcolino i primi della funzione con il paradigma di ricorsione monotona.

Avvertenza Nelle risposte a tutte le domande di questo esercizio si richiede di scrivere i cubi mantenendo l'ordine delle variabili v, w, x, y, z.

(b) Si ottenga una copertura di cardinalita' minima con il metodo di Quine-McCluskey, eseguendo iterativamente i seguenti passi: essenziali, dominanza di righe, dominanza di colonne, e infine risoluzione del nucleo ciclico finale.

(c) Si ottenga una copertura di cardinalita' minima con il metodo di Quine-McCluskey, eseguendo iterativamente i seguenti passi: essenziali, dominanza di colonne, dominanza di righe, e infine risoluzione del nucleo ciclico finale.

Commentare i risultati delle due procedure. E' garantito che produrranno la medesima copertura ? E' garantito che produrranno coperture della medesima cardinalita' ?

13. (a) Si consideri la seguente funzione Booleana f rappresentata dalla copertura

$$F = v'xyz + v'w'x + v'x'z' + v'wxz + w'yz' + vw'z + vwz'.$$

Si calcolino i primi della funzione con il paradigma di ricorsione monotona.

Avvertenza Nelle risposte a tutte le domande di questo esercizio si richiede di scrivere i cubi mantenendo l'ordine delle variabili v, w, x, y, z.

(b) Si ottenga una copertura di cardinalita' minima con il metodo di Quine-McCluskey, eseguendo iterativamente i seguenti passi: essenziali, dominanza di righe, dominanza di colonne, e infine risoluzione del nucleo ciclico finale.

(c) Si ottenga una copertura di cardinalita' minima con il metodo di Quine-McCluskey, eseguendo iterativamente i seguenti passi: essenziali, dominanza di colonne, dominanza di righe, e infine risoluzione del nucleo ciclico finale.

Commentare i risultati delle due procedure. E' garantito che produrranno la medesima copertura ? E' garantito che produrranno coperture della medesima cardinalita' ?

14. Sia dato un circuito che puo' avere uno dei seguenti otto guasti:

$$\{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8\}.$$

Si sono calcolati sei vettori di collaudo:

$$\{v_2, v_2, v_3, v_4, v_5, v_6\},$$

ciascuno dei quali puo' rilevare alcuni dei guasti, secondo la seguente lista:

(a) v_1 rileva la presenza dei guasti $\{g_1, g_3, g_5\}$;

(b) v_2 rileva la presenza dei guasti $\{g_2, g_4, g_6\}$;

(c) v_3 rileva la presenza dei guasti $\{g_1, g_2, g_4, g_7\}$;

(d) v_4 rileva la presenza dei guasti $\{g_3, g_5, g_8\}$;

(e) v_5 rileva la presenza dei guasti $\{g_2, g_6, g_8\}$;

(f) v_6 rileva la presenza dei guasti $\{g_3, g_4, g_7\}$.

Si trovi un sottoinsieme minimo dei vettori di guasto che rilevi tutti gli otto possibili guasti, se presenti. Si supponga che il circuito abbia al piu' un guasto e che tutti i vettori di collaudo abbiano lo stesso costo.

Traccia. Lo si modelli come un problema di copertura monotona.

15. Si considerino le seguenti due affermazioni:
Teorema Data una funzione $ff = (f, d, r)$, sia $\mathcal{F} = \mathcal{G} \cup \{\alpha\}$, dove $\mathcal{F}$ e' una copertura di ff ed α e' un implicante primo disgiunto da $\mathcal{G}$.

Allora α e' un primo essenziale se e solo se

$$\text{CONSENSO}(\mathcal{G}, \{\alpha\})$$

non copre α.

L'operazione CONSENSO tra due cubi $\alpha = a_1 a_2 \dots a_n$ e $\beta = b_1 b_2 \dots b_n$ (a_i e $b_i, i = 1, \dots, n$ sono i letterali in notazione posizionale) e' definita come segue:

$$\text{CONSENSO}(\alpha, \beta) = \begin{cases} a_1 + b_1 & a_2.b_2 & \dots & a_n.b_n \\ a_1.b_1 & a_2 + b_2 & \dots & a_n.b_n \\ \dots \\ a_1.b_1 & a_2.b_2 & \dots & a_n + b_n \end{cases}$$

Nota:

- e' vuoto quando distanza ≥ 2;
- contiene un solo cubo quando distanza = 1;
- contiene n cubi quando distanza = 0.

Corollario Data una funzione $ff = (f, d, r)$, sia $\mathcal{F}$ una copertura di f, $\mathcal{D}$ una copertura di d e α un implicante primo di ff.

Allora α e' un primo essenziale se e solo se $\mathcal{H} \cup \mathcal{D}$ non copre α, dove

$$\mathcal{H} = \text{CONSENSO}((\mathcal{F} \cup \mathcal{D}) \sharp \alpha, \alpha),$$

e l'operazione $\sharp$ tra due cubi α e β e' definita come segue:

$$\alpha \sharp \beta = \begin{cases} a_1.b_1' & a_2 & \dots & a_n \\ a_1 & a_2.b_2' & \dots & a_n \\ \dots \\ a_1 & a_2 & \dots & a_n.b_n' \end{cases}$$

Si noti che invece di $\sharp$ si puo' usare l'operazione $\tilde{\sharp}$ definita come:

$$\alpha \tilde{\sharp} \beta = \begin{cases} a_1.b_1' & a_2 & \dots & a_n \\ a_1.b_1 & a_2.b_2' & \dots & a_n \\ \dots \\ a_1.b_1 & a_2.b_2 & \dots & a_n.b_n' \end{cases}$$

(a) Si dimostrino informalmente le affermazioni precedenti, spiegando che cosa si ottiene applicando le operazioni indicate.

(b) Sfruttando il corollario precedente e' possibile ridurre la prova di essenzialita' a una verifica di contenimento. Indicare di quale verifica si tratta. A quale operazione puo' a sua volta ridursi la verifica di contenimento ? Per quale motivo tale riduzione e' conveniente ?

(c) Si calcolino tutti gl'implicanti primi della funzione $f = a'b'c' + a'b'c + ab'c + abc + abc'$. Quali primi sono essenziali ?

Traccia di soluzione.

La copertura $\mathcal{F} = a'b' + b'c + ac + ab$ contiene tutti e soli i primi di f.

I primi essenziali sono $a'b'$ e ab.

(d) Applicando il corollario precedente, si determini quali implicanti primi della copertura $\mathcal{F} = a'b' + b'c + ac + ab$ sono essenziali. Si mostrino i calcoli in dettaglio.

Applicando il corollario precedente, si determini se l'implicante primo $a'b'$ e' essenziale. Si mostrino i calcoli in dettaglio.

Traccia di soluzione.

Il cubo $a'b'$ in notazione posizionale si scrive come $a_1a_2a_3 = 10\ 10\ 11$, etc.

In base al corollario dobbiamo calcolare se $\mathcal{H} \cup \mathcal{D}$ copre α.

Si consideri la copertura iniziale $\mathcal{F}$ ($\mathcal{D} = \emptyset$) e $\alpha = 10\ 10\ 11$:

```
a'b'c'   10 10 10
a'b'c    10 10 01
ab'c     01 10 01
abc      01 01 01
abc'     01 01 10
```

$(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$ si ottiene come unione dei cubi generati applicando l'operazione $\sharp$ tra i singoli cubi di $(\mathcal{F} \cup \mathcal{D})$ e il cubo α.

10 10 10 $\sharp$ 10 10 11:

```
00 10 10
10 00 10
10 10 00
```

(si tratta di cubi vuoti come ci aspetta dalla definizione dell'operatore $\sharp$ che sottrae i punti del secondo cubo da quelli del primo cubo).

10 10 01 $\sharp$ 10 10 11:

```
00 10 01
10 00 01
10 10 00
```

(si tratta di cubi vuoti come ci aspetta dalla definizione dell'operatore $\sharp$ che sottrae i punti del secondo cubo da quelli del primo cubo).

01 10 01 $\sharp$ 10 10 11:

```
01 10 01
01 00 01
01 10 00
```

(genera il cubo ab'c).

01 01 01 $\sharp$ 10 10 11:

```
01 01 01
01 01 01
01 01 00
```

(genera il cubo minuendo abc).

01 01 10 $\sharp$ 10 10 11:

```
01 01 10
01 01 10
01 01 00
```

(genera il cubo minuendo abc').

In conclusione, $(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$:

```
01 10 01
01 01 01
01 01 10
```

$\mathcal{H}$ = CONSENSO$((\mathcal{F} \cup \mathcal{D}) \sharp \alpha, \alpha)$ si ottiene come unione dei cubi generati applicando l'operazione di consenso tra i singoli cubi di $(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$ e il cubo α.

CONSENSO$(01\ 10\ 01,\ 10\ 10\ 11)$:

```
11 10 01
00 10 01
00 10 11
```

(genera il cubo ponte b'c).

CONSENSO$(01\ 01\ 01,\ 10\ 10\ 11)$:

```
11 00 01
00 11 01
00 00 11
```

(non genera alcun cubo ponte).

CONSENSO$(01\ 01\ 10,\ 10\ 10\ 11)$:

```
11 00 10
00 11 10
00 00 11
```

(non genera alcun cubo ponte).

In conclusione, $\mathcal{H} =$ CONSENSO$((\mathcal{F} \cup \mathcal{D}) \sharp \alpha, \alpha)$:

```
11 10 01
```

che e' il cubo $b'c$.

$\mathcal{H} \cup \mathcal{D}$ copre α ?
Sappiamo che $\alpha \subseteq \mathcal{H} \cup \mathcal{D}$ se e solo se $(\mathcal{H} \cup \mathcal{D})_\alpha = 1$.
$(\mathcal{H} \cup \mathcal{D})_\alpha = (11\ 10\ 01)_{(10\ 10\ 11)} = 11\ 11\ 01$, che e' il cubo c e percio' non e' la tautologia (cioe' non e' l'universo che sarebbe il cubo $11\ 11\ 11$). Percio' α non e' contenuto in $\mathcal{H} \cup \mathcal{D}$ e quindi α e' essenziale.

(e) Applicando il corollario precedente, si determini se l'implicante primo $b'c$ e' essenziale. Si mostrino i calcoli in dettaglio.

Traccia di soluzione.
Il cubo $b'c$ in notazione posizionale si scrive come $a_1a_2a_3 = 11\ 10\ 01$, etc.
In base al corollario dobbiamo calcolare se $\mathcal{H} \cup \mathcal{D}$ copre α.
Si consideri la copertura iniziale $\mathcal{F}$ ($\mathcal{D} = \emptyset$) e $\alpha = 11\ 10\ 01$:

```
a'b'c'   10 10 10
a'b'c    10 10 01
ab'c     01 10 01
abc      01 01 01
abc'     01 01 10
```

$\mathcal{D} = \emptyset$.
$(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$ si ottiene come unione dei cubi generati applicando l'operazione $\sharp$ tra i singoli cubi di $(\mathcal{F} \cup \mathcal{D})$ e il cubo α.

10 10 10 $\sharp$ 11 10 01:

```
00 10 10
10 00 10
10 10 10
```

(genera il cubo minuendo a'b'c').

10 10 01 $\sharp$ 11 10 01:

```
00 10 01
10 00 01
10 10 00
```

(si tratta di cubi vuoti come ci aspetta dalla definizione dell'operatore $\sharp$ che sottrae i punti del secondo cubo da quelli del primo cubo).

01 10 01 $\sharp$ 11 10 01:

```
00 10 01
01 00 01
01 10 00
```

(si tratta di cubi vuoti come ci aspetta dalla definizione dell'operatore $\sharp$ che sottrae i punti del secondo cubo da quelli del primo cubo).

01 01 01 $\sharp$ 11 10 01:

```
00 01 01
01 01 01
01 01 00
```

(genera il cubo minuendo abc).

01 01 10 $\sharp$ 11 10 01:

```
00 01 10
01 01 10
01 01 10
```

(genera il cubo minuendo abc’).

In conclusione, $(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$:

```
10 10 10
01 01 01
01 01 10
```

$\mathcal{H} = \text{CONSENSO}((\mathcal{F} \cup \mathcal{D}) \sharp \alpha, \alpha)$ si ottiene come unione dei cubi generati applicando l'operazione di consenso tra i singoli cubi di $(\mathcal{F} \cup \mathcal{D}) \sharp \alpha$ e il cubo α.

CONSENSO$(10\,10\,10,\ 11\,10\,01)$:

```
11 10 00
10 10 00
10 10 11
```

(genera il cubo ponte a'b').

CONSENSO$(01\,01\,01,\ 11\,10\,01)$:

```
11 00 01
01 11 01
01 00 01
```

(genera alcun cubo ponte ac).

CONSENSO$(01\,01\,10,\ 11\,10\,01)$:

```
11 00 01
01 11 00
01 00 11
```

(non genera alcun cubo ponte).

In conclusione, $\mathcal{H} = \text{CONSENSO}((\mathcal{F} \cup \mathcal{D}) \sharp \alpha, \alpha)$:

```
a'b' 10 10 11
ac   01 11 01
```

$\mathcal{H} \cup \mathcal{D}$ copre α ?
Sappiamo che $\alpha \subseteq \mathcal{H} \cup \mathcal{D}$ se e solo se $(\mathcal{H} \cup \mathcal{D})_\alpha = 1$.
$(\mathcal{H} \cup \mathcal{D})_\alpha = (10\,10\,11 \cup 01\,11\,01)_{(11\ 10\ 01)} = (10\,11\,11 \cup 01\,11\,11)$, che e' la somma di cubi $a' + a$ e quindi e' la tautologia (cioe' e' l'universo 11 11 11). Percio' α e' contenuto in $\mathcal{H} \cup \mathcal{D}$ e quindi α non e' essenziale.

(f) Applicando il corollario precedente, si stabilisca quali implicanti primi della seguente copertura di una funzione multi-valore sono essenziali:

```
.mv 3 0 5 5 5
11111 00001 11110
01100 00011 01010
01010 00100 11111
00110 01001 11010
00001 11111 10110
.e
```

Si ricordi che nel caso multi-valore la distanza tra i termini prodotto S and T e' il numero di letterali vuoti dell'intersezione $S \cap T$ e che il consenso si definisce in generale come:

$$\text{CONSENSO}(S,T) = \emptyset \text{ se distanza}(S,T) \geq 2,$$

$$\text{CONSENSO}(S,T) = \bigcup_{i=1}^{n} X_1^{S_1 \cap T_1} \dots X_i^{S_i \cup T_i} \dots X_n^{S_n \cap T_n} \text{ altrimenti.}$$

16. Si descriva la procedura basata sull'espansione di Shannon per il calcolo degl'implicanti primi di una funzione a piu' valori. Si spieghino le operazioni usate nel calcolo e se argomenti informalmente la correttezza.

17. (a) Per una funzione a due valori, si definiscano le seguenti nozioni:

    i. monotonia crescente rispetto a una variabile,
    ii. monotonia decrescente rispetto a una variabile,
    iii. monotonia rispetto a una variabile,
    iv. monotonia.

    Traccia di soluzione.

    Intuitivamente, una funzione Booleana f e' monotona crescente in x_i se cambiando x_i da 0 a 1 la funzione f rimane invariata o cambia da 0 a 1.

    Una funzione $f(x_1, x_2, \ldots, x_n)$ si dice monotona crescente in x_i se e solo se $f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \subseteq f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$, $\forall x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$.

    Una funzione $f(x_1, x_2, \ldots, x_n)$ si dice monotona decrescente in x_i se e solo se $f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \supseteq f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$, $\forall x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$.

    Se non e' vera nessuna delle due proprieta', f non e' monotona in x_i.

    Una funzione $f(x_1, x_2, \ldots, x_n)$ si dice monotona in x_i se e solo se e' o monotona crescente o monotona decrescente in x_i.

    Una funzione $f(x_1, x_2, \ldots, x_n)$ si dice monotona se e solo se e' monotona in tutte le sue variabili.

    (b) Si definisca la nozione di monotonia crescente e di monotonia decrescente rispetto a una variabile di una copertura F di una funzione f a due valori.

    Traccia di soluzione.

    Una copertura F si dice monotona crescente nella variabile x_i se e solo x_i non appare mai in fase negativa nei termini prodotto di F. Una copertura F si dice monotona decrescente nella variabile x_i se e solo x_i non appare mai in fase positiva nei termini prodotto di F. Se non e' vera nessuna delle due proprieta', F non e' monotona in x_i.

    Se una funzione f e' monotona in x_1 allora esiste una copertura di f monotona in x_i. Se una copertura F di f e' monotona in x_i, allora f e' monotona in x_i.

(c) Si consideri la seguente copertura F della funzione $f(w, x, y, z)$:

```
w x y z
1 - 0 -
- 1 1 -
0 - - 0
1 0 0 -
- 1 - 0
```

Si determini in quali variabili tale copertura F e' monotona.

Traccia di soluzione.

La copertura F e' monotona decrescente in z.

(d) Si consideri la funzione espressa dalla precedente copertura $f = wy' + xy + w'z' + wx'y' + xz'$. Si determini se ci sono delle variabili rispetto a cui f sia monotona crescente o decrescente, mostrando il procedimento per ottenere la risposta.

Traccia di soluzione.

Variabile z. La funzione f e' monotona decrescente in z, poiche' lo e' la sua copertura F.

Variabile x. Il quarto cubo e' contenuto nel primo cubo, percio' e' ridondante e si puo' eliminare. Si ottiene una copertura di f monotona crescente anche in x, per cui f e' monotona crescente anche in x.

Variabile w. Si verifichi se $f_{\overline{w}} \subseteq f_w$, cioe' se vale $xy + z' + xz' \subseteq y' + xy + x'y' + xz'$. Poiche' $w'x'yz' \in xy + z' + xz$, ma $w'x'yz' \notin y' + xy + x'y' + xz'$, la disuguaglianza e' falsa. Anche la disuguaglianza $f_w \subseteq f_{\overline{w}}$, cioe' $y' + xy + x'y' + xz' \subseteq xy + z' + xz'$, e' falsa, poiche' $w'x'y'z \in y' + xy + x'y' + xz'$, ma $w'x'y'z \notin xy + z' + xz'$. Quindi f non e' monotona in w.

Variabile y. Si verifichi se $f_{\overline{y}} \subseteq f_y$, cioe' se vale $w + w'z' + wx' + xz' \subseteq x + w'z' + xz'$. Poiche' $wx'y'z \in w + w'z' + wx' + xz'$, ma $wx'y'z \notin x + w'z' + xz'$, la disuguaglianza e' falsa. Anche la disuguaglianza $f_y \subseteq f_{\overline{y}}$, cioe' $x + w'z' + xz' \subseteq w + w'z' + wx' + xz'$, e' falsa, poiche' $w'xy'z \in x + w'z' + xz'$, ma $w'xy'z \notin w + w'z' + wx' + xz'$. Quindi f non e' monotona in y.

Un metodo sistematico di trovare i punti di controesempio e' quello di ridurre il problema del contenimento (es. $f_{\overline{w}} \subseteq f_w$) alla verifica di tautologia (es. $(f_w)_{(f_{\overline{w}})} = 1$).

18. (a) Si dimostri o si trovi un controesempio alla seguente affermazione: una funzione Booleana a due valori e' monotona se e solo se ogni sua copertura e' monotona.

Traccia di soluzione.

Se una copertura F e' monotona in x_j, la funzione corrispondente e' monotona in x_j. Infatti, sia F monotona crescente in x_j, allora essa si puo' rappresentare come somma di prodotti dove in ogni prodotto o x_j non compare (valore 2) o compare in fase positiva (valore 1). Percio', dato un mintermine, cambiando x_j da 0 a 1, il valore della funzione o rimane invariato (c'e' un prodotto contenente il mintermine con x_j cambiato da 0 a 1 in cui la variabile x_j ha il valore 2) o cambia da 0 a 1 (in ogni prodotto contenente il mintermine con x_j cambiato da 0 a 1 la variabile x_j ha il valore 1).

E' falso che ogni copertura di una funzione monotona sia monotona. Si consideri il seguente semplice contro-esempio: la copertura $F_1 = x'y'z' + x'yz' + x'z$ non e' monotona, ma esiste una copertura monotona equivalente $F_2 = x'$ per cui la funzione f di cui F_1 e F_2 sono coperture e' monotona. Similmente la copertura $G_1 = x'z' + y'z + z$ non e' monotona, ma esiste una copertura monotona equivalente $G_2 = x' + z$ per cui la funzione g di cui G_1 e G_2 sono coperture e' monotona.

Pero' se una funzione e' monotona, c'e' almeno una copertura monotona. Supponiamo che f sia monotona crescente in x_j. Dato un mintermine con $x_j = 0$ per cui la funzione vale 1, dato che cambiando il valore della coordinata x_j da 0 a 1 il valore di f non cambia oppure cambia da 0 a 1, vuol dire che si puo' definire un cubo dove la variabile x_j vale 2 (e il resto e' uguale al mintermine considerato); dato un mintermine con $x_j = 0$ per cui la funzione vale 0, mentre cambiando x_j da 0 a 1 per lo stesso mintermine la funzione vale 1, si puo' definire un cubo dove la variabile x_j vale 1 e il resto e' uguale al mintermine considerato. In questo modo per ogni mintermine dove la funzione vale 1 si puo' definire un cubo dove la variabile x_j vale 2 o 1 e in tal modo si costruisce una copertura monotona crescente in x_j.

(b) Si elenchino le proprieta’ principali delle funzioni binarie monotone e se ne commentino brevemente le loro applicazioni.

Traccia di soluzione.

i. Il complemento di una funzione monotona e’ monotona.
ii. I cofattori di una funzione monotona sono monotoni.
iii. Ogni implicante primo di una funzione monotona e’ essenziale.
iv. Una copertura di primi di una funzione monotona e’ monotona.
v. Una copertura monotona e’ una tautologia se e solo se contiene il cubo tautologo.
vi. Una copertura minimale rispetto al contenimento cubo a cubo e’ una copertura minima di primi.
vii. Una copertura monotona minimale rispetto al contenimento cubo a cubo e’ la copertura minima unica.

19. (a) Si consideri la funzione $f(a, b, c)$ rappresentata dalla seguente copertura F:

```
a b c
0 0 0
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1
```

Si determini in quali variabili tale copertura F e' monotona.

Traccia di soluzione.

La copertura F non e' monotona in alcuna variabile.

(b) Si consideri la funzione f espressa dalla precedente copertura. Si determini se ci sono delle variabili rispetto a cui f sia monotona crescente o decrescente, mostrando il procedimento per ottenere la risposta.

Traccia di soluzione.

La funzione f e' monotona crescente nelle variabli a e b, e monotona decrescente nella variabile c. Cio' e' testimoniato dalla copertura $f = a + b + c'$.

20. (a) Si consideri la funzione $f(a, b, c, d)$ rappresentata dalla seguente copertura F:

a	b	c	d
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	1	0
1	0	1	1
1	1	1	0
1	1	1	1

Si determini in quali variabili tale copertura F e' monotona.

Traccia di soluzione.

La copertura F non e' monotona in alcuna variabile.

(b) Si consideri la funzione f espressa dalla precedente copertura. Si determini se ci sono delle variabili rispetto a cui f sia monotona crescente o decrescente, mostrando il procedimento per ottenere la risposta.

Traccia di soluzione.

La funzione f e' monotona decrescente nella variable a, e monotona crescente nella variabile c, ed e' monotona sia cresente che decrescente nelle variabili b e d. Cio' e' testimoniato dalla copertura $f = a' + c$.

(c) Stabilire per ognuna delle seguenti affermazioni se e' vera o falsa, motivando la risposta.

   i. Una funzione e' monotona se e solo se esiste una copertura monotona.
   Traccia di soluzione.
   VERO.

   ii. Una funzione e' monotona se e solo se ogni copertura e' monotona.
   Traccia di soluzione.
   FALSO. Una funzione monotona puo' avere coperture non-monotone (ma ovviamente deve avere almeno una copertura monotona).

   iii. Se una copertura con degli implicanti non-primi e' monotona, la funzione e' monotona.
   Traccia di soluzione.
   VERO. Basta che ci sia una copertura monotona per dedurre che la funzione e' monotona.
   Domanda: Puo' esistere una copertura monotona con degli implicanti non-primi ? Risposta: Si, ma deve essere ridondante. Es. $f(a,b) = b + ab$.

   iv. Se una copertura con degli implicanti non-primi non e' monotona, la funzione non e' monotona.
   Traccia di soluzione.
   FALSO. Ci potrebbe essere un'altra copertura monotona a testimoniare che la funzione e' monotona. Tale copertura potrebbe contenere anche non-primi o solo primi. (es. $f(a,b) = b + ab + a'b$, $f(a,b) = b + ab$, $f(a,b) = b$).

   v. Se una copertura di primi e' monotona, la funzione e' monotona.
   Traccia di soluzione.
   VERO. Basta che ci sia una copertura monotona (una copertura di primi e' una copertura), affinche' la funzione sia monotona.

   vi. Se una copertura di primi non e' monotona, la funzione non e' monotona.
   Traccia di soluzione.
   VERO. E' la contropositiva del teorema: *una copertura di primi di una funzione monotona e' monotona*, che si puo' riscrivere come: *se una funzione e' monotona, allora ogni sua copertura di primi e' monotona*.

   vii. Solo una tavola di verita' puo' indicare se una funzione e' monotona.

Traccia di soluzione.

FALSO. Non serve avere la tavola della verita', basta una qualsiasi copertura (la cui espansione a primi produce una copertura monotona se e solo la funzione e' monotona).

viii. Per stabilire se una funzione e' monotona, e' necessario generare tutti i primi.

Traccia di soluzione.

FALSO. Non serve generare tutti i primi, basta avere una copertura di primi (che si ottiene espandendo a primi i cubi di una copertura data).

ix. Una funzione completamente specificata e' monotona positiva in x_i se e solo se ogni sua copertura minima (rappresentazione minima come somma di prodotti primi) non contiene il letterale x_i'.

Traccia di soluzione.

VERO. Si noti che una funzione monotona ha un'unica copertura minima che contiene tutti i primi (in quanto sono tutti essenziali).

21. (a) Si definisca il cofattore di Shannon per le funzioni a due valori (a piu' uscite) e poi lo si definisca per le funzioni a piu' valori.
    (b) Si mostri che la definizione per le funzioni a due valori e' un caso speciale di quella a piu' valori.

22. (a) Si definisca il cofattore generalizzato per funzioni completamente specificate e incompletamente specificate.
    (b) Si stabilisca la relazione tra il cofattore generalizzato e quello di Shannon.

23. Si consideri la funzione a due valori con 4 ingressi e 1 uscita rappresentata dalla copertura $F = a'b + a' + bd + bc' + bc'd$.

Applicando l'algoritmo per la complementazione delle funzioni monotone si ottenga una copertura F' del complemento f'. Si verifichi anche che il risultato e' corretto, mostrando le coperture di f e f' su una mappa di Karnaugh.

Traccia di soluzione.

Si ottiene la seguente matrice di personalita' B:

```
a b c d
1 1 0 0
1 0 0 0
0 1 0 1
0 1 1 0
0 1 1 1
```

le cui coperture minimali sono: $(a, b), (a, c, d)$ da cui si ottiene la seguente copertura del complemento:

$F' = ab' + acd'$.

24. Usando il paradigma di ricorsione monotona si calcoli il complemento $\overline{f}$ della seguente funzione f:

```
.mv 3 0 5 5 5
11111 00001 11110
01100 00011 01010
01010 00100 11111
00110 01001 11010
00001 11111 10110
.e
```

La copertura precedente di f e' debolmente monotona ?

La copertura ottenuta di $\overline{f}$ e' debolmente monotona ?

25. (a) Si dimostri che se una funzione f e' fortemente monotona, allora il suo complemento $\overline{f}$ e' una funzione fortemente monotona.
    (b) La copertura del complemento ottenuta con il procedimento di complementazione monotona e' minima ? Dimostrarlo o proporre un controesempio.

26. (a) Si definiscano la nozione di monotonia per funzioni binarie e quella di monotonia debole per le funzioni a piu' valori.

Traccia di soluzione.

Si vedano le dispense per i dettagli.

Una funzione Booleana f e' monotona crescente (decrescente) in x_j se cambiando x_j da 0 a 1 la funzione f rimane invariata o cambia da 0 a 1 (da 1 a 0).

Due definizioni equivalenti sono:

Una funzione Booleana f e' monotona crescente (decrescente) in x_j se $f(m_-) \leq f(m_+)$ ($f(m_-) \geq f(m_+)$), per ogni coppia di mintermini m_-, m_+ che differiscono nella variabile x_j (a 0 in m_- e a 1 in m_+).

Una funzione Booleana f e' monotona crescente (decrescente) in x_j se $f_{\overline{x}_j} \subseteq f_{x_j}$ ($f_{\overline{x}_j} \supseteq f_{x_j}$).

Accettata una come definizione le altre ne conseguono come teoremi.

Una funzione Booleana f e' debolmente monotona in X_j se esiste un valore i tale che cambiando X_j dal valore i ad un qualsiasi altro valore k, la funzione f rimane invariata o cambia da 0 a 1. Il caso binario e' un caso speciale per $i = 0$ (monotonia crescente) o $i = 1$ (monotonia decrescente).

(b) Quali proprieta' fondamentali della monotonia per funzioni binarie si estendono alla monotonia debole per funzioni a piu' valori ?

Traccia di soluzione.

Si vedano le dispense per i dettagli.

Ci sono tre proprieta' fondamentali della monotonia nel caso binario:

(a) Una copertura monotona e' la tautologia se e solo se contiene il cubo universale.

(b) Tutti gli implicanti primi di una funzione monotona sono essenziali.

(c) Il complemento di una funzione monotona e' monotona.

La prima vale anche nel caso di coperture debolmente monotone. Le altre due non valgono per coperture debolmente monotone.

La prima proprieta' nel caso della monotonia debole si dimostra come caso particolare della seguente proposizione. Sia F una copertura debolmente monotona nella variabile X_i, e sia $G = \{c \in F \mid c$ non dipende da $X_i\}$. Allora $G = 1 \Leftrightarrow F = 1$. Infatti come suo corollario si deduce che una copertura debolmente monotona e' una tautologia se e solo se uno dei cubi della copertura e' il cubo universale.

Un'altra proprieta' della monotonia debole usata per calcolare il supercubo del complemento (cioe' il cubo piu' piccolo che contiene il complemento) e' che se F e' una copertura debolmente monotona e F^i rappresenta l'i-esimo cubo della copertura allora

$$supercubo(\overline{F}) = \bigcap_{i=1}^{|F|} supercubo(\overline{F}^i).$$

27. (a) Si consideri la seguente copertura della funzione f:

w	x	y	z	f
1	–	0	–	1
–	1	1	–	1
0	–	–	0	1
1	0	0	–	1
–	1	–	0	1

Si determini in quali variabili tale copertura e' monotona.

(b) Si consideri la funzione $f = wy' + xy + w'z' + wx'y' + xz'$. Si determini se ci siano delle variabili rispetto a cui f e' monotona, mostrando il procedimento per ottenere la risposta.

28. Si determini se la funzione $f = wxy + wyz' + w'x'y' + w'z + xy' + w'y$ e' una tautologia usando il metodo di ricorsione monotona, mostrando il procedimento di calcolo.

29. Si consideri una funzione incompletamente specificata rappresentata con la seguente copertura F:

	a	b	c	d
c1	2	0	0	2
c2	2	0	2	1
c3	1	0	2	2
c4	1	2	2	0
c5	2	1	2	0
c6	2	2	0	0
c7	1	2	1	2

dove D rappresenta l'insieme dei punti dove la funzione non e' specificata:

a	b	c	d
1	1	1	1
2	0	0	1

Si vogliono calcolare le sottocoperture irridondanti di quella data. Si faccia riferimento all'algoritmo presentato in classe, dove s'introduce una variabile y_i per ogni cubo c_i.

(a) Si descriva brevemente l'algoritmo presentato in classe.

Si vedano le note del corso.

L'algoritmo si basa due fatti:

i. $c \subseteq F$ se e solo se $F_c = 1$.

ii. Una coperturna monotona e' tautologa se c'e' almeno una riga che rappresenta l'universo (riga di tutti 2).

L'obiettivo e' calcolare $g(y)$, cui si arriva calcolando $\overline{g}$ e poi complementando. Per calcolare $\overline{g}$, si calcola F_c per ogni cubo $c \in F$; siccome $c \in F$ deve risultare $F_c = 1$ (se $c \in F$, a fortiori $c \subseteq F$). Sviluppando l'albero del calcolo della tautologia di F_c sino a foglie tutte monotone, ogni tale foglia indica attraverso le sue righe universali (fatte di 2) quali cubi dovrebbero essere rimossi contemporaneamente per scoprire almeno un punto di c, definendo un implicante di $\overline{g}$. Complementando $\overline{g}$, si ottiene la $g(y)$, cioe' le combinazioni di cubi c da tenere nella copertura finale per coprire tutti i punti coperti dalla F originale.

Si deve tenere conto anche di D (copertura dei punti dove la funzione non e' definita come 1 o 0), perche' se un cubo c risultasse scoperto in punti che appartengono a D (indicato da una riga di 2 in D_c), allora tale foglia non porterebbe informazioni su cubi da non rimuovere contemporaneamente (perche' la rimozione dei cubi corrispondenti a righe di 2 in F_c scoprirebbe punti in D, come dimostrato da righe di 2 in D_c).

Un implicante primo massimo di g (cioe' con il minimo numero di letterali) definisce una sottocopertura irridondante di F di cardinalita' minima.

(b) Si scriva la funzione $g(y)$ per ispezione della copertura data.

Traccia di soluzione.

Per ogni punto dove la funzione vale 1 (sono 9) si determinano i primi c_i della copertura F che lo contengono (il primo ci e' denotato dalla variabile y_i).

$0000 \Rightarrow y_1 + y_6$
$0011 \Rightarrow y_2$
$0100 \Rightarrow y_5 + y_6$
$0110 \Rightarrow y_5$
$1000 \Rightarrow y_1 + y_3 + y_4 + y_6$
$1011 \Rightarrow y_2 + y_3 + y_7$
$1010 \Rightarrow y_3 + y_4 + y_7$
$1100 \Rightarrow y_4 + y_5 + y_6$
$1110 \Rightarrow y_4 + y_5 + y_7$
$g(y) = (y_1 + y_6)y_2(y_5 + y_6)y_5(y_1 + y_3 + y_4 + y_6)(y_2 + y_3 + y_7)\ (y_3 + y_4 + y_7)(y_4 + y_5 + y_6)(y_4 + y_5 + y_7)$.

(c) Si calcoli la funzione $\overline{g}_1(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia. Si commentino i passi dell'algoritmo.

Traccia di soluzione.

In conclusione risulta $\overline{g}_1(y) = \overline{y}_1\overline{y}_6$.

(d) Si calcoli la funzione $\overline{g}_2(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia. Si commentino i passi dell'algoritmo.

Traccia di soluzione.

Calcolo di $\overline{g}_2(y)$.

Si cofattorizza la copertura $[FD]$ rispetto al cubo c_2 ottenendosi $[F_{c_2} D_{c_2}]$.

F_{c_2}:

```
a b c d
2 2 0 2 c1
2 2 2 2 c2
1 2 2 2 c3
1 2 1 2 c7
```

D_{c_2}:

```
2 2 0 2
```

Poi si cofattoriza la precedente copertura $[F_{c_2} D_{c_2}]$ fino ad arrivare a foglie monotone.

Poiche' la colonna di c non e' monotona, si cofattorizza rispetto a tale variabile, rispetto sia a $\overline{c}$ che a c.

$(F_{c_2})_{\overline{c}}$:

```
2 2 2 2 c1
2 2 2 2 c2
1 2 2 2 c3
```

$(D_{c_2})_{\overline{c}}$:

```
2 2 2 2
```

Non c'e' modo di non avere la tautologia in questa foglia eliminando dei cubi c_i (cioe' in questo caso eliminando c_1 e c_2) per la presenza di una riga di 2 proveniente da D. In altri termini, il sottospazio di questo ramo di cofattorizzazione (prima rispetto a c_2, poi rispetto a $\overline{c}$), contiene solo punti di $[FD]$ che sono in D (sono i punti 0001 e 1001) e che percio non importerebbe se rimanessero scoperti eliminando dei cubi c_i. Percio' questa foglia non contribuisce a $\overline{g_2}$.

$(F_{c_2})_c$:

```
2 2 2 2 c2
1 2 2 2 c3
1 2 2 2 c7
```

$(D_{c_2})_c$:
$\emptyset$.
Se si rimovesse il cubo c_2 non si avrebbe piu' la tautologia, perche' c_2 non sarebbe ricopribile dall'unione degli altri cubi, rimanendo scoperto il punto 0011. Si noti anche che il cubo c_2 e' l'unico a coprire tale punto 0011, il che lo rende primo essenziale.
In conclusione risulta $\overline{g}_2(y) = \overline{y}_2$.

(e) Si calcoli la funzione $\overline{g}_3(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia. Si commentino i passi dell'algoritmo.

Traccia di soluzione.

Calcolo di $\overline{g}_3(y)$.

Si cofattorizza la copertura $[FD]$ rispetto al cubo c_3 ottenendosi $[F_{c_3}D_{c_3}]$.

F_{c_3}:

```
a b c d
2 2 0 2 c1
2 2 2 1 c2
2 2 2 2 c3
2 2 2 0 c4
2 2 0 0 c6
2 2 1 2 c7
```

D_{c_3}:

```
2 2 0 1
```

Poi si cofattoriza la precedente copertura $[F_{c_3}D_{c_3}]$ fino ad arrivare a foglie monotone.

Dato che la colonna di c non e' monotona, si cofattorizza rispetto a tale variabile, rispetto sia a $\overline{c}$ che a c.

Ramo $([FD]_{c_3})_{\overline{c}}$
$(F_{c_3})_{\overline{c}}$:

```
2 2 2 2 c1
2 2 2 1 c2
2 2 2 2 c3
2 2 2 0 c4
2 2 2 0 c6
```

$(D_{c_3})_{\overline{c}}$:

```
2 2 2 1
```

Poiche' la colonna di d non e' monotona, si cofattorizza rispetto a tale variabile, rispetto sia a $\overline{d}$ che a d.
$((F_{c_3})_{\overline{c}})_{\overline{d}}$:

```
2 2 2 2 c1
2 2 2 2 c3
2 2 2 2 c4
2 2 2 2 c6
```

$((D_{c_3})_{\overline{c}})_{\overline{d}}$:
$\emptyset$.
Se si rimovessero contemporaneamente i cubi c_1, c_3, c_4, c_6 non si avrebbe piu' la tautologia, perche' c_3 non sarebbe ricopribile dall'unione degli altri cubi rimasti, rimanendo scoperto il punto 1000.
$((F_{c_3})_{\overline{c}})_d$:

```
2 2 2 2 c1
2 2 2 2 c2
2 2 2 2 c3
```

$((D_{c_3})_{\overline{c}})_d$:

```
2 2 2 2
```

Questa foglia non contribuisce termini prodotto.

Ramo $([FD]_{c_3})_c$

$(F_{c_3})_c$:

```
2 2 2 1 c2
2 2 2 2 c3
2 2 2 0 c4
2 2 2 2 c7
```

$(D_{c_3})_c$:
$\emptyset$.

Poiche' la colonna di d non e' monotona, si cofattorizza rispetto a tale variabile, rispetto sia a $\overline{d}$ che a d.

$((F_{c_3})_c)_{\overline{d}}$:

```
2 2 2 2 c3
2 2 2 2 c4
2 2 2 2 c7
```

$((D_{c_3})_c)_{\overline{d}}$:
$\emptyset$.

Se si rimovessero contemporaneamente i cubi c_3, c_4, c_7 non si avrebbe piu' la tautologia, perche' c_3 non sarebbe ricopribile dall'unione degli altri cubi rimasti, rimanendo scoperto il punto 1010.

$((F_{c_3})_c)_d$:

```
2 2 2 2 c2
2 2 2 2 c3
2 2 2 2 c7
```

$((D_{c_3})_c)_d$:
$\emptyset$.

Se si rimovessero contemporaneamente i cubi c_2, c_3, c_7 non si avrebbe piu' la tautologia, perche' c_3 non sarebbe ricopribile dall'unione degli altri cubi rimasti, rimanendo scoperto il punto 1011.

In conclusione risulta $\overline{g}_3(y) = \overline{y}_1\overline{y}_3\overline{y}_4\overline{y}_6 + \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_2\overline{y}_3\overline{y}_7$.

(f) Si calcoli la funzione $\overline{g}_4(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia. Si commentino i passi dell'algoritmo.

Traccia di soluzione.

In conclusione risulta $\overline{g}_4(y) = \overline{y}_1\overline{y}_3\overline{y}_4\overline{y}_6 + \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_4\overline{y}_5\overline{y}_6 + \overline{y}_4\overline{y}_5\overline{y}_7$.

(g) Si calcoli la funzione $\overline{g}_5(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia.

Traccia di soluzione.

In conclusione risulta $\overline{g}_5(y) = \overline{y}_5\overline{y}_6 + \overline{y}_5 = \overline{y}_5$.

(h) Si calcoli la funzione $\overline{g}_6(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia.

Traccia di soluzione.

In conclusione risulta $\overline{g}_6(y) = \overline{y}_1\overline{y}_6 + \overline{y}_5\overline{y}_6$.

(i) Si calcoli la funzione $\overline{g}_7(y)$, usando l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia.

Traccia di soluzione.

In conclusione risulta $\overline{g}_7(y) = \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_2\overline{y}_3\overline{y}_7 + \overline{y}_4\overline{y}_5\overline{y}_7$.

(j) Date le funzioni $\overline{g}_i(y), i = 1, \ldots n$, si ottenga $\overline{g}_(y)$ e poi si calcoli $g(y)$. si effettui la complementazione con l'algoritmo specializzato per funzioni monotone (passando dalla matrice di personalita' B).

Infine, si ricavi dalla $g(y)$ una sottocopertura irridondante minima della copertura F originale.

Traccia di soluzione.

Dai calcoli precedenti con l'algoritmo di tautologia modificato per determinare le condizioni che falsificano la tautologia si sono determinate le seguenti funzioni $\overline{g}_i(y), i = 1, \ldots 7$:

i. $\overline{g}_1(y) = \overline{y}_1\overline{y}_6$,
ii. $\overline{g}_2(y) = \overline{y}_2$,
iii. $\overline{g}_3(y) = \overline{y}_1\overline{y}_3\overline{y}_4\overline{y}_6 + \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_2\overline{y}_3\overline{y}_7$,
iv. $\overline{g}_4(y) = \overline{y}_1\overline{y}_3\overline{y}_4\overline{y}_6 + \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_4\overline{y}_5\overline{y}_6 + \overline{y}_4\overline{y}_5\overline{y}_7$,
v. $\overline{g}_5(y) = \overline{y}_5\overline{y}_6 + \overline{y}_5 = \overline{y}_5$,
vi. $\overline{g}_6(y) = \overline{y}_1\overline{y}_6 + \overline{y}_5\overline{y}_6$,
vii. $\overline{g}_7(y) = \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_2\overline{y}_3\overline{y}_7 + \overline{y}_4\overline{y}_5\overline{y}_7$.

da cui $\overline{g}(y) = \sum_{i=1}^{7} \overline{g}_i(y)$.

Semplificando con il contenimento cubo a cubo ($\overline{g}(y)$ e' monotona) si ottiene la forma minimizzata $\overline{g}(y) = \overline{y}_1\overline{y}_6 + \overline{y}_2 + \overline{y}_3\overline{y}_4\overline{y}_7 + \overline{y}_5$.

La matrice di personalita' B e' data da:

```
1 0 0 0 0 1 0
0 1 0 0 0 0 0
0 0 1 1 0 0 1
0 0 0 0 1 0 0
```

Una copertura minima di B corrisponde a una sottocopertura irridondante minima della F data

Le coperture minimali (e minime) di colonne sono: $(1,6)(2)(5)(3,4,7)$, cioe' $(2,5,1,3), (2,5,1,4), (2,5,1,7), (2,5,6,3), (2,5,6,4), (2,5,6,7)$, il che corrisponde alla copertura di $g(y)$ espressa come: $g(y) = y_2y_5y_1y_3 + y_2y_5y_1y_4 + y_2y_5y_1y_7 + y_2y_5y_6y_3 + y_2y_5y_6y_4 + y_2y_5y_6y_7$.

Qualsiasi termine prodotto della precedenta copertura definisce una copertura irridondante della funzione originale, ad es. $y_2y_5y_1y_3$ corrisponde alla copertura irridondante $\{c_1, c_2, c_3, c_5\}$. In tutto ci sono 7 coperture irridondanti della funzione originale tutte con la medesima cardinalita' minima di 4 cubi.

30. (a) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $x_1 < x_2 < x_3 < x_4$ (senza lati complementati) della funzione

$$f = x_1 x_2 + \overline{x}_2 x_3 + \overline{x}_1 \overline{x}_3.$$

Si applichi la procedura *apply_ite*. Si mostrino i passi del procedimento seguito.

Traccia di soluzione.

Si considerino come disponibili i diagrammi delle costanti 0 e 1 e delle variabili x_1, x_2 e x_3. Si costruiscano prima i diagrammi dei singoli prodotti $x_1 x_2$, $x_2 x_3$ e $\overline{x}_1 \overline{x}_3$ usando *apply_ite* per calcolare i prodotti. Poi si costruisca il diagramma della somma dei prodotti usando *apply_ite* per calcolare le somme.

(b) Si ottenga il diagramma di decisione binaria ridotto e ordinato della precedente funzione usando anche i lati complementati. Per farlo si ripeta la costruzione precedente usando anche il caso terminale

$$ITE(F, 0, 1) = F'$$

per scoprire quando usare lati complementati.

Si mostrino i passi del procedimento seguito.

Si adotti la convenzione che le negazioni compaiono solo sui lati *else*, per cui i lati *else* si etichettano con un cerchietto vuoto se sono regolari o con un cerchietto pieno se sono complementati (regola per verificare il diagramma della funzione: poiche' i lati *then* non sono mai complementati, il valore di una funzione per il mintermine $1, 1, \ldots, 1$ sara' 0 se e solo se il lato uscente dal nodo associato alla funzione e' complementato).

31. (a) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $a < b < c$ (senza lati complementati) della funzione

$$f = (a + b)c.$$

Si mostrino i passi del procedimento seguito.

(b) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $a < c < b$ (senza lati complementati) della funzione

$$f = (a + b)c.$$

Si mostrino i passi del procedimento seguito.

32. (a) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $a < b < c < d$ (senza lati complementati) della funzione

$$f = abd' + ab'd + a'c + a'c'd.$$

Si mostrino i passi del procedimento seguito.

Traccia di soluzione.

Si applichi la procedura *robdd_build*.

(b) Si ottenga il diagramma di decisione binaria ridotto e ordinato della precedente funzione usando anche i lati complementati, trasformando il diagramma precedente. Si mostrino i passi del procedimento seguito.

Si adotti la convenzione che le negazioni compaiono solo sui lati *else*, per cui i lati *else* si etichettano con un cerchietto vuoto se sono regolari o con un cerchietto pieno se sono complementati (regola per verificare il diagramma della funzione: poiche' i lati *then* non sono mai complementati, il valore di una funzione per il mintermine $1, 1, \ldots, 1$ sara' 0 se e solo se il lato uscente dal nodo associato alla funzione e' complementato).

Traccia di soluzione.
Si converta il grafo ottenuto in precedenza propagando una negazione dalla radice alle foglie. Si ottiene:

```
radice f: (un solo) lato complementato punta a nodo n1 con variabile a
nodo n1: lato T punta a nodo n2 con variabile b
         lato E complementato punta a nodo n3 con variabile c
nodo n2: lato T punta a nodo n4 con variabile d
         lato E complementato punta a nodo n4 con variabile d
nodo n3: lato T punta a nodo '1'
         lato E punta a nodo n4 con variabile d
nodo n4: lato T punta a nodo '1'
         lato E complementato punta a nodo '1'
```

Si noti che il grafo con i lati complementati ha 3 nodi in meno del precedente.

33. (a) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $a < b$ (senza lati complementati) della funzione

$$f = a'b' + a'b + ab' + ab.$$

Si applichi la procedura *robdd_build*. Si mostrino i passi del procedimento seguito.

Traccia di soluzione.

robdd_build($a'b' + a'b + ab' + ab$,1) {
  $\phi \leftarrow \pi(1)$ /* $= a$ */
  $\eta \leftarrow$ robdd_build($b' + b$,2) {
    $\phi \leftarrow \pi(2)$ /* $= b$ */
    $\eta \leftarrow$ robdd_build(1,3)
      return v_1
    $\lambda \leftarrow$ robdd_build(1,3)
      return v_1
    return v_1 /* $\eta = \lambda$ */
  }
  $\lambda \leftarrow$ robdd_build($b' + b$,2) {
    $\phi \leftarrow \pi(2)$ /* $= b$ */
    $\eta \leftarrow$ robdd_build(1,3)
      return v_1
    $\lambda \leftarrow$ robdd_build(1,3)
      return v_1
    return v_1 /* $\eta = \lambda$ */
  }
  return v_1 /* $\eta = \lambda$ */
}

Il diagramma risultante e' costituito dal solo nodo v_1 a cui punta la funzione f.

(b) Si ottenga il diagramma di decisione binaria ridotto e ordinato secondo l'ordine $a < b$ (senza lati complementati) della funzione

$$g = (a' + b')(a' + b)(a + b')(a + b).$$

Si applichi la procedura *robdd_build*. Si mostrino i passi del procedimento seguito.

Traccia di soluzione.

robdd_build($(a' + b')(a' + b)(a + b')(a + b)$,1) {
    $\phi \leftarrow \pi(1)$ /* $= a$ */
    $\eta \leftarrow$ robdd_build($b'b(1 + b')(1 + b)$,2) {
        $\phi \leftarrow \pi(2)$ /* $= b$ */
        $\eta \leftarrow$ robdd_build(0,3)
            return v_0
        $\lambda \leftarrow$ robdd_build(0,3)
            return v_0
        return v_0 /* $\eta = \lambda$ */
    }
    $\lambda \leftarrow$ robdd_build($(1 + b')(1 + b)b'b$,2) {
        $\phi \leftarrow \pi(2)$ /* $= b$ */
        $\eta \leftarrow$ robdd_build(0,3)
            return v_0
        $\lambda \leftarrow$ robdd_build(0,3)
            return v_0
        return v_0 /* $\eta = \lambda$ */
    }
    return v_0 /* $\eta = \lambda$ */
}

Il diagramma risultante e' costituito dal solo nodo v_0 a cui punta la funzione g.

Si noti che il punto di entrambi gli esercizi non e' semplificare come $f = 1$ e $g = 0$, bensi' applicare meccanicamente la procedura *robdd_build*, per ottenere come risultato finale rispettivamente il nodo che rappresenta la funzione identita' e il nodo che rappresenta la funzione nulla.

34. Si rappresentino con diagrammi di decisione binaria ridotti e ordinati secondo l'ordine $a < b < c < d$ (senza lati complementati) le funzioni f e g rappresentate rispettivamente dalle seguenti coperture

$$F = ab + bcd',$$

e

$$G = ab + ac'd.$$

Si applichi la procedura *robdd_build*. Si mostrino i passi del procedimento seguito e il grafo risultante.

Traccia di soluzione.

```
robdd_build(F = ab + bcd',1) {
    φ ← π(1)  /* = a */
    η ← robdd_build(b + bcd',2) {
        φ ← π(2)  /* = b */
        η ← robdd_build(1,3)
            return v_1
        λ ← robdd_build(0,3)
            return v_0
        return v_2 = (b, v_1, v_0)
    }
    λ ← robdd_build(bcd',2) {
        φ ← π(2)  /* = b */
        η ← robdd_build(cd',3) {
            φ ← π(3)  /* = c */
            η ← robdd_build(d',4) {
                φ ← π(4)  /* = d */
                η ← robdd_build(0,5)
                    return v_0
                λ ← robdd_build(1,5)
                    return v_1
                return v_3 = (d, v_0, v_1)
            }
            λ ← robdd_build(0,4)
                return v_0
            return v_4 = (c, v_3, v_0)
        }
        λ ← robdd_build(0,3)
            return v_0
        return v_5 = (b, v_4, v_0)
    }
    return v_6 = (a, v_2, v_5)
}
```

```
robdd_build(G = ab + ac'd,1) {
    φ ← π(1)  /* = a */
    η ← robdd_build(b + c'd,2) {
        φ ← π(2)  /* = b */
        η ← robdd_build(1,3)
            return v_1
        λ ← robdd_build(c'd,3) {
            φ ← π(3)  /* = c */
            η ← robdd_build(0,4)
                return v_0
            λ ← robdd_build(d,4) {
                φ ← π(4)  /* = d */
                η ← robdd_build(1,5)
                    return v_1
                λ ← robdd_build(0,5)
                    return v_0
                return v_7 = (d, v_1, v_0)
            }
            return v_8 = (c, v_0, v_7)
        }
        return v_9 = (b, v_1, v_8)
    }
    λ ← robdd_build(0,2)
        return v_0
    return v_10 = (a, v_9, v_0)
}
```

Si noti che in questo esempio non si riescono a condividere sottografi del diagramma di decisione. Se s'introducessero i lati complementati, al posto dei due nodi v_3 e v_7 si potrebbe usare un nodo solo, a cui si punterebbe da v_4 con un lato positivo e da v_8 con un lato complementato (per indicare che da v_8 si punta al complemento della funzione a cui punta v_4). In realta' con l'introduzione dei lati complementati, tutta la procedura sarebbe modificata per introdurre in modo consistente e canonico le complementazioni.

Addendum sull'operazione ITE

I casi terminali sono

$$
\begin{aligned}
ITE(F,1,0) &= F \\
ITE(1,F,G) &= F \\
ITE(0,F,G) &= G \\
ITE(F,G,G) &= G \\
ITE(F,0,1) &= \overline{G}
\end{aligned}
$$

Ci sono molte triple che danno il medesimo risultato, ad es., $ITE(\overline{x}_1 + x_2x_4, (x_1+x_2)x_4, 0)$ e $ITE(x_2x_4(x_1+\overline{x}_3), 1, \overline{x}_1x_2x_3x_4)$ producono il medesimo risultato x_2x_4.

Sarebbe auspicabile, ma praticamente difficile, identificare tutte le triple che producono il medesimo risultato, perche' potremmo associarle allo stesso elemento della "computed table". Ci sono dei casi speciali in cui si possono identificare facilmente triple equivalenti, ad es.,

$$ITE(F,G,F) = ITE(F,G,0) = ITE(G,F,G) = ITE(G,F,0) = F.G$$

Per questi casi si trasforma una tripla in una forma convenzionale prima di cercarla nella "computed table". In tal modo tutte le triple che sono associate alla stessa forma hanno il medesimo elemento nella tavola.

I seguenti casi permettono di stabilire equivalenze tra triple:

(a) Due argomenti dell'ITE sono uguali, es. $ITE(F,F,G)$

(b) Due argomenti dell'ITE sono l'uno il complemento dell'altro, es. $ITE(F,G,\overline{F})$

(c) Almeno un argomento e' costante, es. $ITE(F,G,0)$

Per arrivare a delle triple convenzionali, si applica una prima serie di trasformazioni che rimpiazza il massimo numero di argomenti con una costante:

$$
\begin{aligned}
ITE(F,F,G) &\Rightarrow ITE(F,1,G) \\
ITE(F,G,F) &\Rightarrow ITE(F,G,0) \\
ITE(F,G,\overline{F}) &\Rightarrow ITE(F,G,1) \\
ITE(F,\overline{F},G) &\Rightarrow ITE(F,0,G)
\end{aligned}
$$

Una seconda serie di trasformazioni usa le seguente uguaglianze

$$\begin{aligned}
ITE(F,1,G) &= ITE(G,1,F) \\
ITE(F,G,0) &= ITE(G,F,0) \\
ITE(F,G,1) &= ITE(\overline{G},\overline{F},1) \\
ITE(F,0,G) &= ITE(\overline{G},0,\overline{F}) \\
ITE(F,G,\overline{G}) &= ITE(G,F,\overline{F})
\end{aligned}$$

per scegliere la tripla il cui primo argomento ha la variabile piu' in alto nel suo grafo con indice minimo nell'ordinamento.

Un'altra serie di trasformazioni usa le seguenti uguaglianze

$$ITE(F,G,H) = ITE(\overline{F},H,G) = \overline{ITE(F,\overline{G},\overline{H})} = \overline{ITE(\overline{F},\overline{H},\overline{G})}$$

In questo caso si sceglie la tripla ai cui due primi argomenti si punta con lati non complementati. Per es., se F non e' complementato e G e' complementato, si sostituisce $ITE(F,G,H)$ con $ITE(F,\overline{G},\overline{H})$ e se ne prende il complemento (da $ITE(F,G,H) = \overline{ITE(F,\overline{G},\overline{H})}$. Similmente, se F e' complementato e G non e' complementato, si si sostituisce $ITE(F,G,H)$ con $ITE(\overline{F},H,G)$ (da $ITE(F,G,H) = ITE(\overline{F},H,G)$).

Tra le conseguenze di queste trasformazioni c'e' la possibilita' di applicare la legge di De Morgan: $F.G = \overline{(\overline{F}+\overline{G})}$; infatti

$$F.G = ITE(F,G,0) = \overline{ITE(\overline{F},1,\overline{G})} = \overline{(\overline{F}+\overline{G})}$$

(si ha $ITE(F,G,0) = \overline{ITE(\overline{F},1,\overline{G})}$ da $ITE(F,G,H) = \overline{ITE(\overline{F},\overline{H},\overline{G})}$).

35. Per ognuna delle seguenti espressioni, si stabilisca se sono algebriche, prime, irridondanti:

   (a) $abc + a'b + bcd$,

   (b) $abc + abd + a'b'c$,

   (c) $abc + bcd + bd$.

36. Si divida $F = ab + ac + ad' + bc + bd'$ per $G = a + b$.

37. Si divida l'espressione algebrica

$$F = abrs + abrt + abd + abe + abu + ghrs + ghrw + ghd + ghe + ghu + dp + eq + rstuw$$

per il divisore $D = ab + gh$. Si ottengano $Q = F/D$ e R.

38. Data l'espressione algebrica $F = abc + abd' + a'bd + abef$ si trovi il divisore primario corrispondente al cubo ab. Tale divisore primario e' un nucleo ?

39. Sia data l'espressione algebrica $F = uwxy + uvxy + tx + tz + rsw + vwxy$.

    (a) Si calcolino i nuclei di livello 0.

    (b) Per ogni nucleo k di livello 0, si esprima F nella forma $ck + r$, dove c e' il co-nucleo e r e' il resto.

La sintesi architetturale di cui alle pagine seguenti non e' inclusa nel programma dell'anno in corso.

40. Il grafo delle precedenze (V, E) ha come nodi le operazioni $v_i,\ i = 0, 1, \ldots, n$ (v_0 e v_n operazioni nulle) e come lati (v_j, v_i) le relazioni di precedenza (cioe' $(v_j, v_i) \in E$ se v_j deve precedere v_i).

Inoltre siano $d_i,\ i = 0, 1, \ldots, n$ i tempi di esecuzione delle operazioni $v_i,\ i = 0, 1, \ldots, n$ ($d_0 = d_n = 0$), e $\mathcal{T} : V \to \{1, 2, \ldots, n_{res}\}$ l'unico tipo di risorsa che realizza una data operazione.

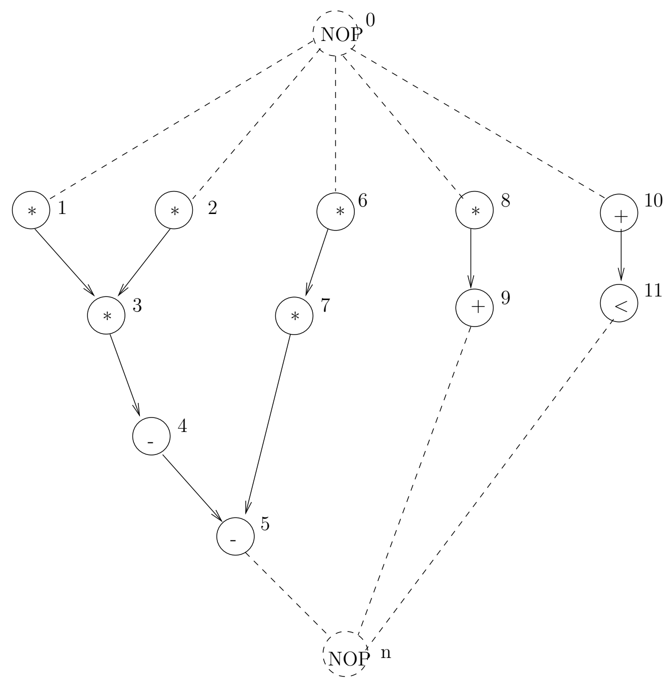

Figure 1: Grafo delle precedenze (da un algoritmo per integrare numericamente con Eulero in avanti l'equazione differenziale $y'' + 3xy' + 3y = 0$).

Si consideri il grafo delle precedenze in Fig. 1. Negli esercizi seguenti, si assumano le seguenti ipotesi, quando non specificato diversamente. Siano $d_i = 1,\ i = 1, 2, \ldots, 11,\ d_0 = d_n = 0$. L'assegnamento delle risorse sia $\mathcal{T}(1) = 1$, $\mathcal{T}(2) = 1$, $\mathcal{T}(3) = 1$, $\mathcal{T}(6) = 1$, $\mathcal{T}(7) = 1$, $\mathcal{T}(8) = 1$; $\mathcal{T}(4) = 2$, $\mathcal{T}(5) = 2$, $\mathcal{T}(9) = 2$, $\mathcal{T}(10) = 2$, $\mathcal{T}(11) = 2$.

(a) (Schedulazione senza vincoli) Si applichi al grafo delle precedenze in Fig. 1 l'algoritmo di schedulazione ASAP e se ne mostrino i passi.

(b) (Schedulazione con vincolo sulla latenza) Si applichi al grafo delle precedenze in Fig. 1 l'algoritmo di schedulazione ALAP con latenza $\overline{\lambda} = 4$ e se ne mostrino i passi.

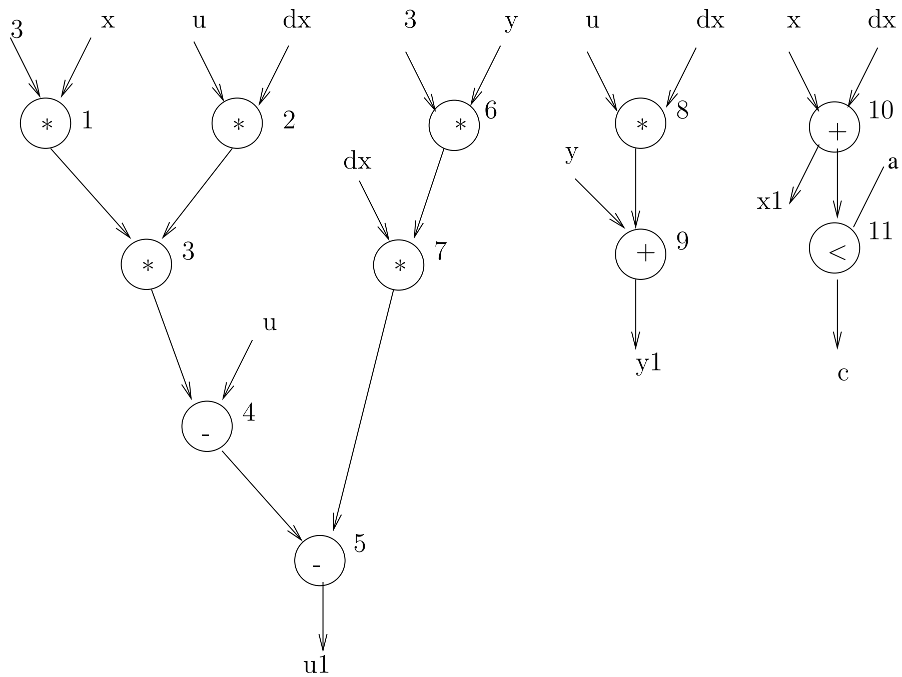

Figure 2: Grafo del flusso dei dati (da un algoritmo per integrare numericamente con Eulero in avanti l'equazione differenziale $y'' + 3xy' + 3y = 0$).

(c) (Schedulazione euristica per latenza minima con l'algoritmo a lista)

i. Si applichi l'algoritmo a lista al grafo delle precedenze in Fig. 1 con $a_1 = 2$ moltiplicatori e $a_2 = 2$ unita' aritmetico-logiche. La funzione di priorita' e' data dalla seguente etichettatura: $v_1 \to 4$, $v_2 \to 4$, $v_3 \to 3$, $v_4 \to 2$, $v_5 \to 1$, $v_6 \to 3$, $v_7 \to 2$, $v_8 \to 2$, $v_9 \to 1$, $v_{10} \to 2$, $v_{11} \to 1$, che indica la lunghezza del percorso piu' lungo dal nodo al pozzo.
Inoltre i tempi di esecuzione delle operazioni siano $d_1 = 2$ per il moltiplicatore e $d_2 = 1$ per l'unita' aritmetico-logica.

ii. Si ripeta l'esercizio precedente supponendo $d_1 = 1$ per il moltiplicatore e $d_2 = 1$ per l'unita' aritmetico-logica.

iii. Si applichi l'algoritmo a lista al grafo delle precedenze in Fig. 1 con $a_1 = 3$ moltiplicatori e $a_2 = 1$ unita' aritmetico-logica. Inoltre i tempi di esecuzione delle operazioni siano $d_1 = 2$ per il moltiplicatore e $d_2 = 1$ per l'unita' aritmetico-logica.
Si usi come funzione di priorita' ancora la lunghezza del percorso piu'

lungo dal nodo al pozzo.

iv. Si ripeta l'esercizio precedente assumendo una funzione di priorita' diversa che assegna l'inizio delle operazioni v_2, v_6, v_8 al primo ciclo di calcolo.

(d) (Schedulazione euristica per risorse minime con l'algoritmo a lista)
Si applichi l'algoritmo a lista al grafo delle precedenze in Fig. 1 nell'ipotesi che tutte le operazioni richiedano un tempo unitario e che si richieda una latenza di 4 cicli.

(e) Qual e' la relazione tra l'algoritmo euristico a lista e l'algoritmo di Hu (per la schedulazione di multiprocessori) ? Nel rispondere si descriva l'algoritmo di Hu.

(f) (Schedulazione per latenza minima con vincoli sulle risorse)
Si consideri la formulazione classica con ILP della schedulazione con vincoli sulle risorse, rispetto alle variabili $X = \{x_{il}, i = 0, 1, \ldots, n; l = 1, 2, \ldots, \overline{\lambda} + 1\}$, con $x_{il} = 1$ solo quando l'operazione v_i inizia al passo l della schedulazione, e $\overline{\lambda}$ maggiorante sulla latenza:
Si noti che $x_{il} = 0$ per $l < t_i^S$ o $l > t_i^L$ per ogni operazione $v_i, i = 0, 1, \ldots, n$, dove t_i^S sono i minoranti calcolati dall'algoritmo ASAP ($x_{il} = 1 \rightarrow l \geq t_i^S$) e t_i^L sono i maggioranti calcolati dall'algoritmo ALAP ($x_{il} = 1 \rightarrow l \leq t_i^L$), da cui

$$\Sigma_l \, x_{il} = \Sigma_{l=t_i^S}^{t_i^L} \, x_{il}, \; i = 0, 1, \ldots, n$$

(il tempo d'inizio d'ogni operazione e' unico, percio' l'inizio t_i di ogni operazione v_i puo' essere scritto come $t_i = \Sigma_l \, l.x_{il}$, $t_i^S \leq t_i \leq t_i^L$).
I vincoli sono:

$$(*) \; \Sigma_l \, x_{il} = 1, i = 0, 1, \ldots, n$$

(il tempo d'inizio d'ogni operazione e' unico)

$$(**) \; \Sigma_l \, l.x_{il} \geq \Sigma_l \, l.x_{jl} + d_j, \; i, j = 0, 1, \ldots, n, \; (v_j, v_i) \in E$$

(le relazioni di precedenza devono essere soddisfatte)

$$(***) \; \Sigma_{i:\mathcal{T}(v_i)=k} \, \Sigma_{m=l-d_i+1}^{l} x_{im} \leq a_k, \; k = 1, 2, \ldots, n_{res}, \; l = 1, 2, \ldots, \overline{\lambda}+1$$

(i vincoli sulle risorse devono essere soddisfatti ad ogni passo della schedulazione)

Ogni insieme di tempi iniziali che soddisfa i vincoli precedenti ci da' una soluzione ammissibile.

Si consideri la schedulazione con vincoli sulle risorse del grafo in Fig. 1. Si assuma che ci siano due tipi di risorse: moltiplicatore e unita' aritmetico-logica (che esegue somme/sottrazioni e confronti). Entrambe le risorse eseguano in un ciclo. Si assuma anche che si possano usare al piu' due moltiplicatori e due unita' aritmetico-logiche: $a_1 = 2$, $a_2 = 2$.

i. Usando l'algoritmo euristico di schedulazione a lista ("list scheduling") si trovi un maggiorante sulla latenza.

ii. Con gli algoritmi ASAP and ALAP si trovino dei vincoli sui tempi d'inizio, t_i^S and t_i^L.

iii. Si scrivano in dettaglio tutte le equazioni precedenti per l'unicita' dei tempi di partenza, le relazioni di precedenza e le quantita' di risorse disponibili.

iv. Si risolvano i vincoli precedenti (*), (**), (***) per ottenere una soluzione ammissibile.

v. Si risolvano i vincoli precedenti (*), (**), (***) con la funzione di minimo
$$min\, \mathbf{c}^T \mathbf{t}$$
e $\mathbf{c} = [0, 0, \dots, 1]^T$ equivalente a
$$min\, t_n = \Sigma_l\, l.x_{nl}$$
che corrisponde a minimizzare la latenza della schedulazione (da $\overline{\lambda} = t_n - t_0$ si ha che una soluzione ottima implica $t_0 = 1$).

vi. Si risolvano i vincoli precedenti (*), (**), (***) con la funzione obiettivo
$$min\, \mathbf{c}^T \mathbf{t}$$
e $\mathbf{c} = [1, 1, \dots, 1]^T$ equivalente a
$$min\, \Sigma_i t_i = \Sigma_i \Sigma_l\, l.x_{il}$$
che corrisponde a minimizzare il tempo iniziale per tutte le operazioni.

(g) (Schedulazione per risorse minime con vincoli sulla latenza)

L'obiettivo dell'ottimizzazione e' una somma pesata delle risorse usate rappresentata dal vettore $\mathbf{a}$, esprimibile come

$$min\, \mathbf{c}^T \mathbf{a}$$

dove $\mathbf{c}$ e' il vettore dei costi delle singole risorse. Nelle equazioni (***) gli $a_k,\ k = 1, 2, \ldots, n_{res}$ sono adesso incognite ausiliarie.
C'e' inoltre un vincolo sulla latenza

$$t_n = \Sigma_l\, l.x_{nl} \leq \overline{\lambda} + 1$$

Per i dati del problema precedente, assumendo inoltre $\mathbf{c} = [5, 1]^T$ (il moltiplicatore costa 5 unita' di area e l'unita' aritmetico-logica ne costa una) e un maggiorante sulla latenza di $\overline{\lambda} = 4$, si riscrivano i vincoli del tipo (***) (quelli dei tipi (*) e (**) sono uguali al caso precedente) e si minimizzi la funzione di costo

$$\mathbf{c}^T \mathbf{a} = 5.a_1 + 1.a_2$$

41. Il problema dell'associazione ("binding") e' quello di associare risorse a operazioni. Tale problema si puo' formulare con un modello di programmazione lineare per interi, che estende quello visto per la schedulazione per latenza minima con vincoli sulle risorse.

Per semplicita' si assuma che tutte le operazioni e risorse siano dello stesso tipo. In questo modo si gestiscono i problemi con piu' tipi di operazioni come problemi indipendenti uno per ogni tipo di risorsa (grazie al fatto che i tipi sono disgiunti).

S'introduce un insieme di variabili di decisione binarie con due indici, $B = \{b_{ir}; i = 1, 2, \ldots, n_{op}; r = 1, 2, \ldots, a\}$, e un insieme di costanti di decisione binarie con due indici, $X = \{x_{il}; i = 1, 2, \ldots, n_{op}; l = 1, 2, \ldots, \lambda + 1\}$, dove $a \leq n_{op}$ e' un maggiorante sul numero di risorse da usarsi. La variabile binaria b_{ir} vale 1 solo quando l'operazione v_i e' associata alla risorsa r, cioe' $\beta(v_i) = (1, r)$, dove il primo argomento di β e' il tipo di risorsa. La costante binaria x_{ir} vale 1 solo quando l'operazione v_i inizia al passo l della schedulazione, cioe' $l = t_i$, come definito nel modello ILP per la schedulazione con vincoli sulle risorse. Questi valori sono costanti conosciute, poiche' si assume di partire da grafi schedulati.

La ricerca di un'associazione compatibile con una data schedulazione (quest'ultima rappresentata da X) e un maggiorante sulla risorsa a e' equivalente a ricercare dei valori B che soddisfano i seguenti vincoli:

$$
\begin{aligned}
\sum_{r=1}^{a} b_{ir} &= 1, \; i = 1, 2, \ldots, n_{op} \\
\sum_{i=1}^{n_{op}} \sum_{m=l-d_i+1}^{l} x_{im} &\leq 1, \; l = 1, 2, \ldots, \lambda + 1, \; r = 1, 2, \ldots, a \\
b_{ir} &\in \{0, 1\}, \; i = 1, 2, \ldots, n_{op}, \; r = 1, 2, \ldots, a
\end{aligned}
$$

Il primo gruppo di vincoli assicura che ogni operazione v_i sia assegnata ad una e una sola risorsa. Il secondo gruppo di vincoli impone che al piu' un'operazione sia in esecuzione, tra quelle assegnate alla risorsa r, in ogni istante di tempo. Poiche' e' sufficiente imporre che le variabili in B siano interi non-negativi per soddisfare l'ultimo gruppo di vincoli (che le variabili in B siano 0 o 1, il problema puo' risolversi come programmazione lineare per interi (invece che programmazione lineare per $0 - 1$). Il modello puo' essere esteso facilmente a tipi multipli di operazioni.

Sia data la seguente schedulazione del grafo delle precedenze in Fig. 1:

$x_{1,1} = x_{2,1} = x_{3,2} = x_{4,3} = x_{5,4} = x_{6,2} = x_{7,3} = x_{8,3} = x_{9,4} = x_{10,1} = x_{11,2} = 1.$

Ci sono due tipi di risorse (moltiplicatore, etichettato con 1, e unita' aritmetico-logica, etichettata con 2). Si supponga che ciascuna risorsa richieda un tempo di esecuzione di un'unita'. Un'associazione ammissibile delle operazioni alle risorse deve soddisfare i seguenti vincoli.
Per i moltiplicatori:

$$\begin{aligned} \sum_{r=1}^{a_1} b_{ir} &= 1,\ \forall\, i : \mathcal{T}(v_i) = 1 \\ \sum_{i:\mathcal{T}(v_i)=1} b_{ir} x_{il} &\leq 1,\ l = 1, 2, \ldots, \lambda + 1,\ r = 1, 2, \ldots, a_1 \end{aligned}$$

Per gli addizionatori:

$$\begin{aligned} \sum_{r=1}^{a_2} b_{ir} &= 1,\ \forall\, i : \mathcal{T}(v_i) = 2 \\ \sum_{i:\mathcal{T}(v_i)=2} b_{ir} x_{il} &\leq 1,\ l = 1, 2, \ldots, \lambda + 1,\ r = 1, 2, \ldots, a_2 \end{aligned}$$

(a) Si trovi, se esiste. una realizzazione con $a_1 = 1$ moltiplicatore, indicando i valori delle incognite in B.

(b) Si trovi, se esiste. una realizzazione con $a_1 = 2$ moltiplicatori, indicando i valori delle incognite in B.

(c) Si trovi, se esiste. una realizzazione con $a_2 = 1$ unita' aritmetico-logiche, indicando i valori delle incognite in B.

(d) Si trovi, se esiste. una realizzazione con $a_2 = 2$ unita' aritmetico-logiche, indicando i valori delle incognite in B.

(e) Si mostri il grafo delle precedenze con le associazioni di operazioni a risorse per il caso $a_1 = 2$, $a_2 = 2$.

42. Il grafo delle precedenze (V, E) ha come nodi le operazioni $v_i,\ i = 0, 1, \ldots, n$ (v_0 e v_n operazioni nulle) e come lati (v_j, v_i) le relazioni di precedenza (cioe' $(v_j, v_i) \in E$ se v_j deve precedere v_i).

    Inoltre siano $d_i,\ i = 0, 1, \ldots, n$ i tempi di esecuzione delle operazioni $v_i,\ i = 0, 1, \ldots, n$ ($d_0 = d_n = 0$), e $\mathcal{T} : V \to \{1, 2, \ldots, n_{res}\}$ l'unico tipo di risorsa che realizza una data operazione.

    Si consideri il grafo delle precedenze in Fig. 3. Siano $d_i = 1,\ i = 1, 2, \ldots, 11$, $d_0 = d_n = 0$. L'assegnamento delle risorse sia $\mathcal{T}(1) = 1$, $\mathcal{T}(2) = 1$, $\mathcal{T}(3) = 1$, $\mathcal{T}(6) = 1$, $\mathcal{T}(7) = 1$, $\mathcal{T}(8) = 1$; $\mathcal{T}(4) = 2$, $\mathcal{T}(5) = 2$, $\mathcal{T}(9) = 2$, $\mathcal{T}(10) = 2$, $\mathcal{T}(11) = 2$.

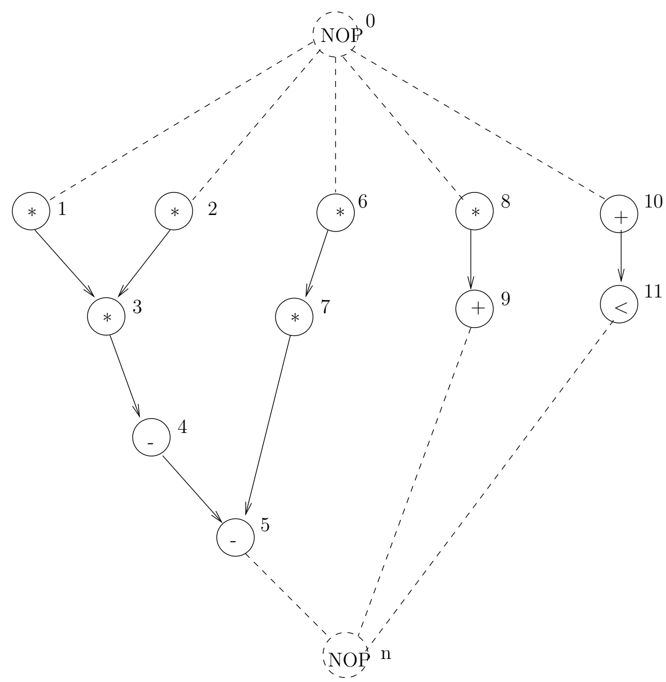

Figure 3: Grafo delle precedenze (da un algoritmo per integrare numericamente con Eulero in avanti l'equazione differenziale $y'' + 3xy' + 3y = 0$).

(a) (Schedulazione senza vincoli) Si applichi al grafo delle precedenze in Fig. 1 l'algoritmo di schedulazione ASAP e se ne mostrino i passi per ottenere il vettore dei risultati $\mathbf{t}^S$.

    Traccia di soluzione.

    Definiamo $\mathbf{t}^S$ il vettore dei tempi iniziali calcolati dall'algoritmo ASAP, cioe' $\mathtt{t}^S = \{t_i^S : i = 0, 1, \ldots, n\}$. L'algoritmo ASAP produce i tempi iniziali minimi.

L'algoritmo ASAP inizia con $t_0^S = 1$. Allora i vertici i cui predessori sono stati assegnati sono: $\{v_1, v_2, v_6, v_8, v_{10}\}$. Percio' il loro tempo iniziale e' posto a $t_0^S + d_0 = 1 + 0 = 1$ e cosi' via. Cosi' si attiene:
$t_0^S = 1$, $t_1^S = 1$, $t_2^S = 1$, $t_3^S = 2$, $t_4^S = 3$, $t_5^S = 4$, $t_6^S = 1$, $t_7^S = 2$, $t_8^S = 1$, $t_9^S = 2$, $t_{10}^S = 1$, $t_{11}^S = 2$, $t_n^S = 5$.

Poiche' $t_n^S = 5$, la latenza e' $\lambda = 5 - 1 = 4$.

(b) (Schedulazione con vincolo sulla latenza) Si applichi al grafo delle precedenze in Fig. 3 l'algoritmo di schedulazione ALAP con latenza $\overline{\lambda} = 4$ e se ne mostrino i passi per ottenere il vettore dei risultati $\mathtt{t}^L$.

Traccia di soluzione.

Definiamo $\mathtt{t}^L$ il vettore dei tempi iniziali calcolati dall'algoritmo ASAP, cioe' $\mathtt{t}^L = \{t_i^L : i = 0, 1, \ldots, n\}$. L'algoritmo ALAP produce i tempi iniziali massimi.

L'algoritmo ALAP con $\overline{\lambda} = 4$ inizia con $t_n^L = 5$. Allora i vertici i cui successori sono stati assegnati sono $\{v_5, v_9, v_{11}\}$. Percio' il loro tempo iniziale e' posto a $t_n^L - 1 = 5 - 1 = 4$ e cosi' via. Cosi' si attiene:
$t_0^L = 1$, $t_1^L = 1$, $t_2^L = 1$, $t_3^L = 2$, $t_4^L = 3$, $t_5^L = 4$, $t_6^L = 2$, $t_7^L = 3$, $t_8^L = 3$, $t_9^L = 4$, $t_{10}^L = 3$, $t_{11}^L = 4$, $t_n^L = 5$.

(c) Quante risorse di tipo 1 e quante di tipo 2 sono necessarie per ASAP ? Quante per ALAP ?

Traccia di soluzione.

ASAP usa 4 risorse di tipo 1 (moltiplicatori) e 2 risorse di tipo 2 (unita' aritmetico-logiche).

ALAP usa 2 risorse di tipo 1 (moltiplicatori) e 3 risorse di tipo 2 (unita' aritmetico-logiche).

Qual e' la mobilita' dei nodi ?

Traccia di soluzione.

La mobilita' e' definita come $\mu = t_i^L - t_i^S, i = 0, 1, \ldots, n$. Si ottiene:
$t_0^L - t_0^S = 0$, $t_1^L - t_1^S = 0$, $t_2^L - t_2^S = 0$, $t_3^L - t_3^S = 0$, $t_4^L - t_4^S = 0$, $t_5^L - t_5^S = 0$, $t_6^L - t_6^S = 1$, $t_7^L - t_7^S = 1$, $t_8^L - t_8^S = 2$, $t_9^L - t_9^S = 2$, $t_{10}^L - t_{10}^S = 2$, $t_{11}^L - t_{11}^S = 2$, $t_n^L - t_n^S = 0$.

Esistono schedulazioni che usano meno risorse di quante sono richieste da ASAP e ALAP a parita' di latenza ?

Traccia di soluzione.

Si, a parita' della latenza $\lambda = 4$ si puo' scendere a 2 risorse di tipo 1 (moltiplicatori) e 2 risorse di tipo 2 (unita' aritmetico-logiche):
$t_0 = 1$, $t_1 = 1$, $t_2 = 1$, $t_3 = 2$, $t_4 = 3$, $t_5 = 4$, $t_6 = 2$, $t_7 = 3$, $t_8 = 3$, $t_9 = 4$, $t_{10} = 1$, $t_{11} = 2$, $t_n = 5$.

43. Il grafo delle precedenze (V, E) ha come nodi le operazioni $v_i,\ i = 0, 1, \ldots, n$ (v_0 e v_n operazioni nulle) e come lati (v_j, v_i) le relazioni di precedenza (cioe' $(v_j, v_i) \in E$ se v_j deve precedere v_i).

    Inoltre siano $d_i,\ i = 0, 1, \ldots, n$ i tempi di esecuzione delle operazioni $v_i,\ i = 0, 1, \ldots, n$ ($d_0 = d_n = 0$), e $\mathcal{T} : V \to \{1, 2, \ldots, n_{res}\}$ l'unico tipo di risorsa che realizza una data operazione.

    Si consideri il grafo delle precedenze in Fig. 4. Siano $d_i = 1,\ i = 1, 2, \ldots, 11$, $d_0 = d_n = 0$. L'assegnamento delle risorse sia $\mathcal{T}(1) = 1$, $\mathcal{T}(2) = 1$, $\mathcal{T}(3) = 1$, $\mathcal{T}(6) = 1$, $\mathcal{T}(7) = 1$, $\mathcal{T}(8) = 1$; $\mathcal{T}(4) = 2$, $\mathcal{T}(5) = 2$, $\mathcal{T}(9) = 2$, $\mathcal{T}(10) = 2$, $\mathcal{T}(11) = 2$.

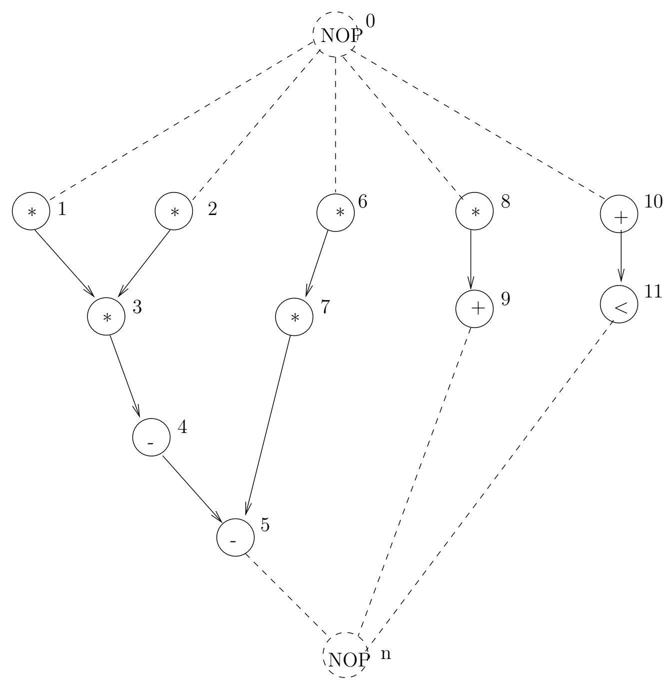

Figure 4: Grafo delle precedenze (da un algoritmo per integrare numericamente con Eulero in avanti l'equazione differenziale $y'' + 3xy' + 3y = 0$).

    (a) (Schedulazione euristica per latenza minima con l'algoritmo a lista)
    Si descriva l'algoritmo euristico a lista ("list scheduling") per schedulazione a latenza minima.

(b) Si applichi l'algoritmo a lista al grafo delle precedenze in Fig. 4 con $a_1 = 2$ moltiplicatori e $a_2 = 2$ unita' aritmetico-logiche. La funzione di priorita' sia data dalla seguente etichettatura: $v_1 \to 4$, $v_2 \to 4$, $v_3 \to 3$, $v_4 \to 2$, $v_5 \to 1$, $v_6 \to 3$, $v_7 \to 2$, $v_8 \to 2$, $v_9 \to 1$, $v_{10} \to 2$, $v_{11} \to 1$, che indica la lunghezza del percorso piu' lungo dal nodo al pozzo. Inoltre i tempi di esecuzione delle operazioni siano $d_1 = 1$ per il moltiplicatore e $d_2 = 1$ per l'unita' aritmetico-logica.

Si descrivano con precisione i passi dell'algoritmo.

(c) Qual e’ la relazione tra l’algoritmo euristico a lista e l’algoritmo di Hu (per la schedulazione di multiprocessori) ? Nel rispondere si descriva l’algoritmo di Hu.