Implementazione del Simplesso Duale

Corso di Ricerca Operativa 2012-2013

Roberto Roberti

roberto.roberti6@unibo.it

23 Aprile 2013

Introduzione

- Implementare il Simplesso Duale
  - Definizione dell'input
  - Definizione dell'output
  - Definizione delle strutture dati
  - Analisi dell'algoritmo
- Implementazione in C in ambiente Visual Studio 2008/2010/2012 (facoltativo l'uso di C++ e/o di un altro IDE)
- Test su 9 problemi test disponibili su `http://campus.unibo.it/`

Definizione dell'Input

Un esempio di file di input

```
   4      3
2.00   1.00   1.00
  -1      0      1
20.00     3      1   1.0   2   2.0   3   2.0
-6.00     2      2   1.0   3   1.0
 8.00     2      2   2.0   3   1.0
 3.00     2      1   4.0   2   5.0
```

... che corrisponde al problema

$$
\begin{aligned}
z = \min \quad & 20x_1 - 6x_2 + 8x_3 + 3x_4 && (1)\\
\text{s.t.} \quad & x_1 + 4x_4 \geq 2 && (2)\\
& 2x_1 + x_2 + 2x_3 + 5x_4 = 1 && (3)\\
& 2x_1 + x_2 + x_3 \leq 1 && (4)\\
& x_1, x_2, x_3, x_4 \geq 0 && (5)
\end{aligned}
$$

Definizione dell'Input

... quindi

- la prima riga riporta il numero di variabili n e il numero di vincoli m
- la seconda riga riporta i termini noti b_i, $i = 1, \ldots, m$
- seguono i segni degli m vincoli, dove 0 significa $=$, 1 significa $\leq$, -1 significa $\geq$
- infine sono riportati i dati per ognuna delle n variabili:
  - c_j (il coefficiente in funzione obiettivo)
  - r_j (il numero di coefficienti diversi da zero nella colonna j dei vincoli)
  - per ciascuno degli r_j coefficienti, l'indice di riga i e il coefficiente a_{ij}

Definizione dell'Output

L'applicazione deve restituire in un file di testo le seguenti informazioni

- stato della soluzione
  - 1: soluzione ottima trovata
  - -1: il problema non ha soluzione ammissibile
  - -2: soluzione illimitata
- valore della funzione obiettivo
- valore di tutte le variabili
- tempo di calcolo (in secondi)

Cosa È Richiesto alla Discussione dell'Elaborato

- Il codice sorgente sviluppato e l'eseguibile usato per i test finali, accertandosi che tutto funzioni sui computer del laboratorio
- Una relazione di una facciata con
  - nome e cognome
  - matricola
  - data
  - linguaggio di programmazione
  - processore sui cui sono stati svolti i test
  - una tabella che riporti, per ognuna delle nove istanze
    - nome dell'istanza
    - costo della soluzione ottima trovata
    - tempo di calcolo

Strutture Dati

```
#define   Prec     0.000001
#define   Inf      10000000000.
#define   BigM     10000.
#define   MaxVar   5000          // #colonne massime tableau
#define   MaxCon   500           // #righe massime tableau

// Dati input
long    m;                      // #vincoli
long    n;                      // #variabili
double  Mat[MaxCon][MaxVar];    // tableau
int     Segno[MaxCon];          // segno vincoli

// Variabili simplesso duale
long m1;                // #righe + variabile artificiale
long n1;                // #colonne + variabili slack
long n2;                // #colonne + variabili slack + variabile articifiale
long Base[MaxCon];      // colonna in base per ogni riga
long Stato[MaxVar];     // riga per cui ogni colonna e in base, 0=fuori base
int  Artif;             // vincolo artificiale? 0=no >0=si
```

Strutture Dati

Posizione dei vari dati nel tableau

	0	1	...	n	n+1 ... n1	n2
0	z	c_1	$\ldots$	c_n		
1	b_1	a_{11}	$\ldots$	a_{1n}	variabili	
$\vdots$	$\ldots$	$\vdots$	$\ddots$	$\vdots$	di	
m	b_m	a_{m1}	$\ldots$	a_{mn}	slack	
m1	vincolo artificiale					slack

Main

```
1 void main(void) {
2     // lettura dati di input
3     Risolvi_Duale();
4     // stampa output
5 }
```

Simplesso Duale

```
1  void Risolvi_Duale(void ) {
2     Opt = 1;
3
4     Aggiungi_Variabili_Slack();
5
6     Calcola_Base_Iniziale();
7     if ( Opt != 1 )    return;
8
9     Aggiungi_Variabile_Artificiale();
10
11    while ( ( i = Trova_Uscente() ) > 0 ) {
12       j = Trova_Entrante(i);
13       if ( j == 0 ) {
14          Opt = -1;
15          return;
16       }
17       Stato[Base[i]] = 0;
18       Base[i]        = j;
19       Stato[j]       = i;
20       Pivot(i,j);
21    }
22
23    // verifiche soluzione illimitata
24 }
```

Aggiunta Variabili di Slack

```
1  void Aggiungi_Variabili_Slack(void ) {
2     n1 = n;
3
4     for ( i = 1 ; i <= m ; ++i ) {
5        if ( Segno[i] != 0 ) {
6           ++n1;
7           for ( k = 0 ; k <= m ; ++k )  Mat[k][n1] = 0.0;
8           Mat[i][n1] = Segno[i];
9        }
10    }
11
12    n2 = n1;
13    m1 = m;
14 }
```

Calcolo Base Iniziale

```
1  void Calcola_Base_Iniziale(void ) {
2     for ( j = 0 ; j <= n1 ; ++j ) Stato[j] = 0;
3
4     for ( i = 1 ; i <= m1 ; ++i ) {
5        for ( j = n1 ; j >= 1 ; --j )
6           if ( ( Mat[i][j] < -Prec ) || ( Mat[i][j] > Prec ) )
7              break;
8
9        if ( j > 0 ) {
10          Base[i]  = j;
11          Stato[j] = i;
12          Pivot(i,j);
13       } else {
14          Base[i] = 0;
15          if ( ( Mat[i][0] < -Prec ) || ( Mat[i][0] > Prec ) ) {
16             Opt = -1;
17             return;
18          }
19       }
20    }
21 }
```

Aggiunta Variabile Artificiale

```
1  void Aggiunta_Variabile_Artificiale(void ) {
2      Artif = 0;
3      max   = Prec;
4      for ( j = 1 ; j <= n1 ; ++j )
5          if ( Mat[0][j] > max ) {
6              max   = Mat[0][j];
7              Artif = j;
8          }
9
10     if ( Artif != 0 ) {
11         ++m1;
12         ++n2;
13
14         Mat[m1][0]  = BigM;
15         Mat[m1][n2] = 1.0;
16         for ( j = 1 ; j < n2 ; ++j ) {
17             if ( Stato[j] == 0 ) Mat[m1][j] = 1.0;
18             else                 Mat[m1][j] = 0.0;
19         }
20         for ( i = 0 ; i < m1 ; ++i ) Mat[i][n2] = 0.0;
21
22         Base[m1]     = Artif;
23         Stato[Artif] = m1;
24         Stato[n2]    = 0;
25         Pivot(m1, Artif);
26     }
27 }
```

Operazione di Pivoting

```
1 void Pivot(long i, long j) {
2     // Dato l'indice di riga i e l'indice di colonna j
3     // questa funzione deve fare il pivoting, nella matrice Mat[][],
4     // sull'elemento Mat[i][j]
5 }
```

Ricerca Variabile Uscente

```
1 int Trova_Uscente(void ) {
2     // deve restituire l'indice di riga della
3     // variabile da far uscire dalla base
4 }
```

Ricerca Variabile Entrante

```
1 int Trova_Entrante(int i) {
2     // data la riga i deve restituire
3     // l'indice della colonna da far entrare in base
4 }
```