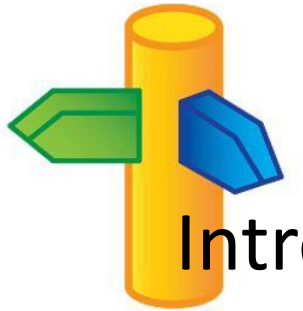
UNIVERSITÀ DELLA CALABRIA



Introduzione alle funzionalità del pacchetto

IBM ILOG CPLEX Optimization Studio

A cura di:

Cristian Belfiore, Dottore Magistrale in Ingegneria Gestionale

de-Health Lab - Laboratory of Decision Engineering for Health Care Services

Ponte Pietro Bucci 41C, 8° piano - 87036 Rende (Cosenza)
mail: **cristian.belfiore@unical.it**

UNIVERSITÀ DELLA CALABRIA
DIPARTIMENTO DI
INGEGNERIA MECCANICA,
ENERGETICA E GESTIONALE
DIMEG

Indice:

- **Introduzione**
- **IBM ILOG CPLEX Optimization Studio**
- **IBM ILOG Script**

IBM ILOG CPLEX Optimization Studio

Pacchetto composto da:

- L'ottimizzatore matematico **CPLEX**
- Il linguaggio di programmazione dichiarativo **OPL**, con costrutti che richiamano l'algebra dei modelli di ottimizzazione
- Il linguaggio di programmazione procedurale **IBM ILOG Script**, utile per modificare il comportamento del risolutore
- Un **IDE**, ambiente di programmazione che supporta il programmatore nello sviluppo del codice

Creazione di un nuovo progetto OPL:

File → New → OPL Project → necessario a questo punto impostare:

- **Project Name** (obbligatorio, il nome deve essere univoco)
- Possibile introdurre una descrizione del progetto (facoltativa) in **Description**
- Già da qui, se si vuole, è **possibile generare automaticamente altre parti del progetto** utili nel seguito, spuntando:

    - **Create Model**: crea il file dove dovrà essere scritto il modello
    - **Create Data**: crea il file dove dovranno essere inseriti i dati della specifica istanza del problema
    - **Create Settings**: crea il file dal quale è possibile impostare i parametri del risolutore
    - **Add Run Configuration**: genera il file che permette di avviare il risolutore su una istanza del problema

Un primo semplice modello di ottimizzazione:

$$\min 3x + 4y$$

$$\text{s.t.}$$

$$x \geq 15$$

$$x + 2y \geq 16$$

$$x \geq 0, \text{ intero}$$

$$y \geq 0, \text{ intero}$$

Struttura del file .mod:

```
1  /*********************************************
2   * OPL 22.1.0.0 Model
3   * Author: Win10
4   * Creation Date: 16 gen 2023 at 09:55:03
5   *********************************************/
6
7
8
9  dvar int x;
10 dvar int y;
11
12 maximize 3*x+4*y;
13
14 subject to {
15   x <= 15;
16   x+2*y <= 16;
17   }
```

Dichiarazione di **coefficienti** dal valore noto o di altre **strutture dati** necessarie

Dichiarazione delle **variabili decisionali**

variabili continue: tipo ***float***
variabili intere: tipo ***int***
variabili binarie: tipo ***boolean***

non negatività: tipi ***float+*** o ***int+***

Funzione obiettivo

maximize

o

minimize

Vincoli

subject to {…}

o

constraints {…}

Inizializzazione diretta: consiste nel riportare **direttamente nel corpo del codice che definisce il modello** il valore dei coefficienti di costo, dei coefficienti delle risorse e dei coefficienti tecnologici e, più in generale, di tutti i dati, senza fare uso del file .dat.

Tale **approccio è sconsigliato**: il modello risultante **manca di generalità**, esso è costruito per risolvere una specifica istanza del problema e, qualora si volesse risolvere una istanza differente dello stesso problema, sarebbe necessario modificare tutto il codice.

Utilizzo dei file .mod e .dat:

Inizializzazione indiretta dei dati: i dati vengono definiti nel file .mod, ma il loro valore viene configurato nel file .dat

In questo modo il **modello risulta del tutto generale** e lo stesso potrà essere utilizzato per risolvere differenti istanze dello stesso problema semplicemente generando diversi file .dat, uno per ogni istanza del problema che si vuole affrontare.

```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: Win10
 * Creation Date: 24 gen 2023 at 11:43:26
 *********************************************/

int n = ...;  // N oggetti
range Oggetti = 1..n;
float Peso [Oggetti] = ...;
float Valore [Oggetti] = ...;

int Capacita = ...;

dvar boolean Selezionato [Oggetti];

maximize sum (i in Oggetti) Valore[i]*Selezionato[i];

subject to {

  sum (i in Oggetti) Peso[i]*Selezionato[i] <= Capacita;

}
```

Estratto del file .dat in cui si può osservare come viene assegnato il valore ai coefficienti definiti nel file .mod

```
n = 12;

Capacita = 1000;

Peso = [114, 90, 75, 80, 55, 140, 120, 105, 66, 98, 165, 100];

Valore = [50, 50, 80, 10, 10, 30, 50, 80, 100, 20, 60, 50];
```

Il file .dat può essere configurato in modo tale che la **lettura dei dati avvenga da un file di Excel** e **che i valori della funzione obiettivo e delle variabili decisionali individuate dal risolutore siano riportate in uno stesso o differente file di Excel**

```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Cristian
 * Creation Date: 8 feb 2023 at 12:06:46
 *********************************************/

SheetConnection Sheet1 ("SALPB.xlsx");

NOperazioni from SheetRead(Sheet1, "Foglio1!A2");

NStazioni from SheetRead(Sheet1, "Foglio1!B2");

TempiOperazioni from SheetRead(Sheet1, "Foglio1!B6:B23");

Precedenze from SheetRead(Sheet1, "Foglio2!B5:S22");

TempoCiclo to SheetWrite(Sheet1, "Risultati!A2");

Assegnato to SheetWrite(Sheet1, "Risultati!C8:L25");
```

IBM ILOG CPLEX Optimization Studio

Strutturare i dati: Insiemi, Range, Array

```
14
15 dvar boolean Assegnato [Componenti][Linee];
16 dvar boolean Attiva [Linee];
17
```

Definisce una **matrice** di variabili decisionali binarie di nome ***Assegnato*** che ha tante righe quanti sono gli elementi del *range* ***Componenti*** e tante colonne quanti sono gli elementi del *range* ***Linee*** (|Componenti|*|Linee| elementi).
Gli elementi della matrice sono indicizzati rispetto a tali *range.*

```
2  * OPL 22.1.0.0 Model
3  * Author: Cristian
4  * Creation Date: 2 feb 2023 at
5  ********************************
6  {string} SitiProduttivi = ...;
7  {string} Magazzini = ...;
8  {string} PuntiVendita = ...;
9
```

{tipo} nomeInsieme
[***setof (tipo) nomeInsieme***]

Definisce un ***insieme***, ovvero una collezione di elementi non indicizzati e non duplicati.
Su tali insiemi è possibile eseguire operazioni: ***union***, ***inter***, ***diff***

```
1  /*********************************************
2   * OPL 22.1.0.0 Model
3   * Author: Win10
4   * Creation Date: 20 gen 2023 at 10:05:54
5   *********************************************/
6  int n = ...;
7  int l = ...;
8  range Componenti = 1..n;
9  range Linee = 1..l;
10 int CapLinea = ...;
11 int TempiProduzione [Componenti] = ...;
```

range nomeRange = LB … UB

Un ***range*** definisce un intervallo ordinato di valori interi, utile per:

- indicizzare un array
- specificare l'insieme dei valori che le variabili decisionali possono assumere

Definisce un **vettore** di interi di nome ***TempiProduzione*** che ha tante componenti quanti sono gli elementi del *range* ***Componenti***, nei cui valori è indicizzato

IBM ILOG CPLEX Optimization Studio

Strutturare i dati: Insiemi, Range, Array (Delicius)

ESEMPIO

Delicious, azienda alimentare inglese specializzata nel settore dolciario, sta valutando la possibilità di avviare la produzione di tre nuovi prodotti. L'azienda dispone di due stabilimenti, che possono essere utilizzati entrambi per realizzare tutti i prodotti. I tempi necessari per produrre una confezione di ciascun prodotto, sono evidenziati nella Tabella 3.16, in cui viene riportato anche il profitto unitario.

	Tempi di Lavorazione [h]		
Prodotto	Stabilimento 1	Stabilimento 2	Profitto Unitario [€]
1	2	2	500
2	3	1	400
3	1	4	600

Tabella 3.16. Dati relativi al problema della pianificazione della produzione della Delicious.

Per ogni stabilimento è stata individuata anche la capacità giornaliera, cioè, il numero di ore giornaliere in cui lo stabilimento può essere utilizzato. In particolare, il primo stabilimento può essere utilizzato al massimo per 10 ore al giorno, mentre la capacità giornaliera del secondo stabilimento è pari a 18 ore. Al fine di evitare un'eccessiva diversificazione della produzione e limitare i costi logistici, l'azienda ha deciso che solo uno dei due stabilimenti dovrà essere utilizzato per la produzione e al massimo due dei tre prodotti dovranno essere messi in produzione. L'obiettivo che si vuole raggiungere è quello di massimizzare il profitto giornaliero. Dal momento che bisogna imporre che solo uno dei due stabilimenti può essere utiliz-

* Da:
Caramia M., Giordani S., Guerriero F., Musmanno R., Pacciarelli D., "Ricerca Operativa", Isedi, 2014, Novara. p. 95

IBM ILOG CPLEX Optimization Studio

Strutturare i dati: Insiemi, Range, Array (Delicius)

P = insieme dei prodotti (i).
S = insieme degli stabilimenti (j).
π_i = profitto unitario derivante dalla vendita di un prodotto di tipo i $\in$ P.
T_j = tempo massimo giornaliero per la produzione nello stabilimento j $\in$ S.
M = quantità molto grande rispetto agli altri dati del problema.

k_i = variabile decisionale binaria che vale 1 se il prodotto i $\in$ P viene selezionato, 0 altrimenti.
y_i = variabile decisionale binaria che vale 1 se lo stabilimento j $\in$ S viene selezionato, 0 altrimenti.
x_{ij} = variabile decisionale intera che rappresenta la quantità giornaliera di prodotto i $\in$ P che viene realizzata nello stabilimento j $\in$ S.

Il Modello:

$$max \sum_{i \in P} \sum_{j \in S} \pi_i x_{ij}$$

subject to:

$$\sum_{i \in P} k_i \leq 2$$

$$\sum_{j \in S} y_j = 1$$

$$\sum_{i \in P} t_{ij} x_{ij} \leq T_j y_j \; : \forall j \in S$$

$$\sum_{j \in S} x_{ij} \leq M k_i \; : \forall i \in P$$

$$k_i \in \{0,1\} \; : \forall i \in P$$

$$y_j \in \{0,1\} \; : \forall j \in S$$

$$x_{ij} \geq 0 \text{ intero} \; : \forall i \in P, j \in S$$

```
/*********************************************
 * OPL 22.1.0.0 Model
 * Author: Win10
 * Creation Date: 7 feb 2023 at 10:49:00
 *********************************************/
{string} Prodotti = ...;
{string} Stabilimenti = ...;

int TempiLavorazione [Prodotti][Stabilimenti] = ...;

int ProfittoUnitario [Prodotti] = ...;

int DisponibilitaOrariaStabilimento [Stabilimenti] = ...;

dvar boolean Attivo [Stabilimenti];
dvar boolean Selezionato [Prodotti];
dvar int+ Produzione [Prodotti][Stabilimenti];

maximize sum (p in Prodotti, s in Stabilimenti) ProfittoUnitario[p] * Produzione[p][s];

subject to {

  sum (s in Stabilimenti) Attivo[s] == 1;

  sum (p in Prodotti) Selezionato[p] <= 2;

  forall (s in Stabilimenti)
    sum (p in Prodotti) Produzione [p][s] * TempiLavorazione[p][s] <= DisponibilitaOrariaStabilimento[s] * Attivo[s];

  forall (p in Prodotti)
    sum (s in Stabilimenti) Produzione[p][s] <= 1000 * Selezionato[p];
}
```

```
/*********************************************
 * OPL 22.1.0.0 Data
 * Author: Win10
 * Creation Date: 7 feb 2023 at 10:49:09
 *********************************************/
Prodotti = {"P1", "P2", "P3"};
Stabilimenti = {"S1", "S2"};

TempiLavorazione = [[2, 2]
                    [3, 1]
                    [1, 4]];

ProfittoUnitario = [500, 400, 600];

DisponibilitaOrariaStabilimento = [10, 18];
```

Se necessario, l'istruzione ''per ogni'' può essere condizionata in modo molto semplice:
forall (o in Oggetti : condizioneSu_o)
forall (o in Oggetti : condizioneSu_o)

```
// solution (optimal) with objective 7200
Del prodotto P1 non viene avviata la produzione.
Del prodotto P2 viene avviata la produzione.
Del prodotto P3 non viene avviata la produzione.

Lo stabilimento S1 non viene attivato.
Lo stabilimento S2 viene attivato.

Il prodotto P2 è realizzato in 18 pezzi nello stabilimento S2
```

IBM ILOG CPLEX Optimization Studio

<u>Strutturare i dati</u>: Insiemi, Range, Array (Delicius)

Vincolo che traduce simultaneamente una condizione logica ed un vincolo tecnologico

$$\sum_{i \in P} t_{ij} x_{ij} \leq T_j y_j \quad \forall j \in S$$

$j = 1$
Stabilimento 1

$$2x_{11} + 3x_{21} + x_{31} \leq 10\, y_1 \longrightarrow$$

$y_1 = 1 \longrightarrow x_{11}, x_{21}, x_{31} \geq 0$ *ma tali per cui* $2x_{11} + 3x_{21} + x_{31} \leq 10$

Lo stabilimento viene attivato.
In questo è possibile produrre degli articoli soddisfacendo il vincolo di disponibilità temporale.

$y_1 = 0 \longrightarrow x_{11}, x_{21}, x_{31} = 0$

Lo stabilimento non viene attivato. All'interno di questo non può essere prodotto nulla.

IBM ILOG CPLEX Optimization Studio

<u>Strutturare i dati</u>: Insiemi, Range, Array (Delicius)

Medesimo risultato si otterrebbe scrivendo la seguente coppia di gruppi di vincoli:

$$\sum_{i \in P} t_{ij} x_{ij} \leq T_j \qquad \forall j \in S$$

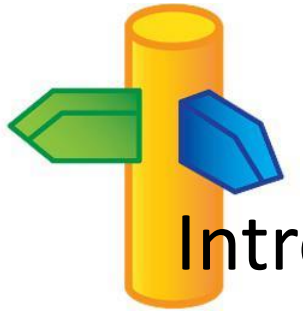
⟶ Traduzione del vincolo tecnologico sulla disponibilità oraria degli stabilimenti

$$\sum_{i \in P} t_{ij} x_{ij} \leq M\, y_j \qquad \forall j \in S$$

⟶ Traduzione della condizione logica

Big M, parametro sufficientemente grande da non costituire una limitazione aggiuntiva non voluta nel caso in cui lo stabilimento j venga attivato ($y_j = 1$)

UNIVERSITÀ DELLA CALABRIA



Introduzione alle funzionalità del pacchetto

IBM ILOG CPLEX Optimization Studio

A cura di:

Cristian Belfiore, Dottore Magistrale in Ingegneria Gestionale

de-Health Lab - Laboratory of Decision Engineering for Health Care Services

Ponte Pietro Bucci 41C, 8° piano - 87036 Rende (Cosenza)
mail: **cristian.belfiore@unical.it**

UNIVERSITÀ DELLA CALABRIA
DIPARTIMENTO DI
INGEGNERIA MECCANICA,
ENERGETICA E GESTIONALE
DIMEG