Análisis y Diseño de Algoritmos II

Jesús Alexander Aranda Ph.D   Robinson Duque, Ph.D
Juan Francisco Díaz, Ph. D

Universidad del Valle

jesus.aranda@correounivalle.edu.co
robinson.duque@correounivalle.edu.co
juanfco.diaz@correounivalle.edu.co
Programa de Ingeniería de Sistemas
Escuela de Ingeniería de Sistemas y Computación

Overview

Tipos de Variables y Problemas

Recordemos...

Tipo de variables:

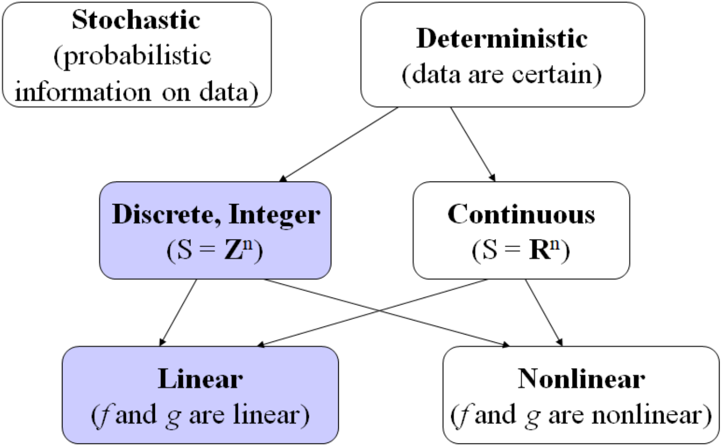
- x_j puede ser continua
- x_j puede ser binaria (0 o 1)
- x_j puede ser entera (1, 2, 3,... ,N)
- x_j puede ser discreta (e.g., tomar valores 10mm, 20mm, 30mm)

Tipos de Variables y Problemas

Diferentes nombres especializados se le dan al problema NLP:

- **Programación Lineal (LP):** cuando todas las funciones (objetivo y restricciones) son lineales
- **Programación Entera (IP):** un LP con la restricción que todas las variables deben ser enteras
- **Programación binaria:** caso especial de IP donde las variables son 0 o 1
- **Programación Entera Mixta (MIP):** un IP donde algunas de las variables son enteras y otras continuas
- **MINLP:** un MIP con funciones no lineales
- **Programación Cuadrática (QP):** cuando una función objetivo es una función cuadrática en x y todas las restricciones son lineales

Tipos de Variables y Problemas



Programación Binaria

Varios problemas de programación de enteros se pueden llevar a una forma donde cada una de las variables toma los valores 0 o 1.

La clase anterior se evidenció un ejemplo al crear un modelo IP y su implementación en MiniZinc que representara una instancia SAT:

$$C = (a_1 \vee a_2 \vee \neg a_4) \wedge (a_2 \vee \neg a_3) \wedge (a_1 \vee \neg a_2) \wedge (a_3 \vee a_5 \vee a_6 \vee \neg a_7)$$

Programación Entera Mixta (MIP)

Es muy común encontrar problemas de optimización (o de satisfacción) donde la solución dependa de un conjunto de variables binarias y contínuas (o enteras). Este tipo de programación recibe el nombre de programación entera mixta (MIP). Ejemplo:

Satisfacción de una restricción entre dos restricciones:
Dadas dos restricciones de desigualdad:

$$g_1(x) \leq 0, g_2(x) \leq 0, x \in R^n$$

Se requiere que al menos una de las restricciones g_i esté satisfecha.

Programación Entera Mixta (MIP)

Esto se puede formular en una forma adecuada para la optimización numérica como:

$$g_1(x) \leq M * y$$

$$g2(x) \leq M * (1 - y)$$

$$y \in \{0, 1\}$$

donde M es un número constante grande (Big M). Así, hay un total de $n + 1$ variables, donde n son continuas y 1 es binaria.

¿Puede ver porqué una de las dos restricciones siempre se satisface?

Programación Entera Mixta (MIP)

O no exclusivo:

Suponga que tiene un modelo de programación lineal con las dos siguientes restricciones: $2x_1 + x_2 \geq 5$, $2x_3 - x_4 \leq 2$ y que quiere modelar lo siguiente:

$$2x_1 + x_2 \geq 5 \text{ o}$$

$$2x_3 - x_4 \leq 2 \text{ o ambas}$$

Se debe agregar una variable y_1 y se requiere que sea binaria:

$$2x_1 + x_2 \geq 5 - M * y_1$$

$$2x_3 - x_4 \leq 2 + M(1 - y_1)$$

Programación Entera Mixta (MIP)

Satisfacción de un conjunto de de restricciones alternativas:
¿Cómo se puede modelar que k de las siguientes p restricciones se satisfagan?

$$g_1(x) \leq 0, g_2(x) \leq 0, ..., g_p(x) \leq 0, x \in R^n$$

Programación Entera Mixta (MIP)

Ejemplo de solución para 3 restricciones: dado un conjunto de restricciones de desigualdad:

$$g_1(x) \leq 0, g_2(x) \leq 0, g_3(x) \leq 0, x \in R^n$$

Se requiere que al menos una de las restricciones g_i esté satisfecha. Esto se puede formular en una forma adecuada para la optimización numérica como:

$$g_1(x) \leq M * y_1, g2(x) \leq M * y_2, g_3(x) \leq M * y_3,$$

$$y_1 + y_2 + y_3 \leq 2$$

$$y_i \in \{0, 1\}, i = 1, 2, 3$$

donde M es un valor constante grande (Big M). Así, hay un total de $n + 3$ variables, donde n son continuas y 3 son binarias.

Programación Entera Mixta (MIP)

Sea X_i una proposición lógica y x_i su variable binaria asociada:

- Si X_1 o X_2 son verdaderos, entonces X_3 es verdadero:

Programación Entera Mixta (MIP)

Sea X_i una proposición lógica y x_i su variable binaria asociada:

- Si X_1 o X_2 son verdaderos, entonces X_3 es verdadero:

$$x_1 \leq x_3, x_2 \leq x_3$$

  donde $x_1 = 1$ si X_1 es verdadero y 0 si es falso.

- Si X_3 es verdadero, entonces X_1 o X_2 son verdaderos:

Programación Entera Mixta (MIP)

Sea X_i una proposición lógica y x_i su variable binaria asociada:

- Si X_1 o X_2 son verdaderos, entonces X_3 es verdadero:

$$x_1 \leq x_3, x_2 \leq x_3$$

  donde $x_1 = 1$ si X_1 es verdadero y 0 si es falso.

- Si X_3 es verdadero, entonces X_1 o X_2 son verdaderos:

$$x_1 + x_2 \geq x_3$$

- Si ambos X_1 y X_2 son verdaderos, entonces X_3 es verdadero:

Programación Entera Mixta (MIP)

Sea X_i una proposición lógica y x_i su variable binaria asociada:

- Si X_1 o X_2 son verdaderos, entonces X_3 es verdadero:

$$x_1 \leq x_3, x_2 \leq x_3$$

  donde $x_1 = 1$ si X_1 es verdadero y 0 si es falso.

- Si X_3 es verdadero, entonces X_1 o X_2 son verdaderos:

$$x_1 + x_2 \geq x_3$$

- Si ambos X_1 y X_2 son verdaderos, entonces X_3 es verdadero:

$$1 + x_3 \geq x_1 + x_2$$

Programación Entera Mixta (MIP)

Si...entonces...

Para transformar una restricción:

si $2x_1 + x_2 \leq 5$ entonces $2x_3 - x_4 \geq 2$,

se debe tener en cuenta que: $A \rightarrow B$ es equivalente a $\neg A or B$:

$2x_1 + x_2 > 5$ o $2x_3 - x_4 \geq 2$ o ambas.

Pero hay una complicación añadida. No nos gusta una restricción estricta de la desigualdad ($2x_1 + x_2 > 5$). Por lo tanto, mostraremos cómo llevar a cabo la transformación en el caso especial de que x_1 y x_2 tienen un valor entero:

$2x_1 + x_2 \geq \mathbf{6}$ o $2x_3 - x_4 \geq 2$ o ambas.

Ahora se procede a transformar el "o"...

Programación Entera Mixta (MIP)

Si...entonces...

$2x_1 + x_2 \geq \mathbf{6}$ o $2x_3 - x_4 \geq 2$ o ambas.

Equivale a:

$$2x_1 + x_2 \geq 6 - M * y_1$$

$$2x_3 - x_4 \geq 2 - M(1 - y_1)$$

$$y_1 \in \{0, 1\}$$

Restricciones booleanas en MiniZinc

Las restricciones booleanas se pueden modelar utilizando enteros 0..1 o variables boolenas al igual que restricciones lineales:

```
b1 = (b2 \/ b3)
b1 >= b2 /\ b1 >= b3 /\ b2 + b3 >= b1
b1 = (b2 /\ b3)
b2 >= b1 /\ b3 >= b1 /\ b1+1 >= b2+b3
b1 = (b2 -> b3)
b1 >= 1 - b2 /\ b1 >= b3 /\ 1-b2 + b3 >= b1
```

Restricciones booleanas en MiniZinc

Retomemos el problema para producir alimentos:

```
var int: X;
var int: Y;
var int: C;

constraint 15*X + 20*Y >= 60;
constraint 10*X + 5*Y >= 30;
constraint X>=0;
constraint Y>=0;
constraint C=80*X+50*Y;
solve minimize C;
output ["X=", show(X), "\n Y=", show(Y), "\n C=",
    show(C)];
```

Solución: X=2, Y=2, C=260

Restricciones booleanas en MiniZinc

Asuma que queremos satisfacer las restricciones (proteina o grasa o ambas) tal que se minimice la función objetivo:

```
var int: X;
var int: Y;
var int: C;

constraint 15*X + 20*Y >= 60;
constraint 10*X + 5*Y >= 30;
constraint X>=0;
constraint Y>=0;
constraint C=80*X+50*Y;
solve minimize C;
output ["X=", show(X), "\n Y=", show(Y), "\n C=",
    show(C)];
```

Solución: X=2, Y=2, C=260

Restricciones booleanas en MiniZinc

Opción 1:

```
var int: X;
var int: Y;
var int: C;
var bool: aux;
constraint 15*X + 20*Y >= 60 - 1000 * aux;
constraint 10*X + 5*Y >= 30 - 1000 * (1- aux);
constraint X>=0;
constraint Y>=0;
constraint C=80*X+50*Y;
solve minimize C;

output ["X=", show(X), "\n Y=", show(Y), "\n C=",
    show(C), "\n aux=", show(aux)];
```

Solución: X=0, Y=3, C=150, aux=false

Restricciones booleanas en MiniZinc

Opción 2:

```
var int: X;
var int: Y;
var int: C;
var bool: a;
var bool: b;
constraint a = (15*X + 20*Y >= 60);
constraint b = (10*X + 5*Y >= 30);
constraint a \/ b ;
constraint X>=0;
constraint Y>=0;
constraint C=80*X+50*Y;
solve minimize C;
output ["X=", show(X), "\n Y=", show(Y), "\n C=",
    show(C) , "\n a=", show(a) , "\n b=", show(b)];
```

Solución: X=0, Y=3, C=150, a=true, b=false

Restricciones booleanas en MiniZinc

Opción 3: O Exclusivo

```
var int: X;
var int: Y;
var int: C;
var bool: a;
var bool: b;
constraint a = (15*X + 20*Y >= 60);
constraint b = (10*X + 5*Y >= 30);
constraint a + b = 1;
constraint X>=0;
constraint Y>=0;
constraint C=80*X+50*Y;
solve minimize C;
output ["X=", show(X), "\n Y=", show(Y), "\n C=",
    show(C) , "\n a=", show(a) , "\n b=", show(b)];
```

Solución: X=0, Y=3, C=150, a=true, b=false

Restricciones booleanas en MiniZinc

Asuma el siguiente programa MIP:

$$
\begin{array}{lll}
\texttt{maximize} & \multicolumn{2}{l}{4x_1 + 5x_2 + x_3 + 6x_4} \\
\texttt{subject to} & (1) & 7x_1 + 3x_2 + 4x_3 + 2x_4 \leq 6 \\
 & (2) & 2x_1 + 5x_2 + x_3 + 5x_4 \leq 4 \\
 & (3) & x_1 + 3x_2 + 5x_3 + 2x_4 \leq 7 \\
 & \multicolumn{2}{l}{x_1, x_2, x_3, x_4 \geq 0, x_1, x_2 \ \text{enteros}}
\end{array}
$$

Escriba implementacines en MiniZinc para que:

- Se cumpla exactamente una de las restricciones
- Se cumplan mínimo dos restricciones
- Si se cumple la restricción 3 se debe cumplir la 2
- Se cumpla la restricción 1 o la 3 pero no ambas
- Se cumplan las restricciónes 1 y 3 o 2 y 3, pero nunca se cumplan todas

Problema de la Mochila

Un caminante desea llenar su mochila de capacidad W=6 de forma que se maximice la utilidad de los objetos que empaque. Formule este problema como un MIP teniendo en cuenta la siguiente tabla:

		utility	weight
A	a picture	12	0.2
B	a bottle	7	2
C	another bottle	3	2
D	a pullover	4	0.8
E	chocolate bars	5	1
F	dried fruit	8	1

El objetivo es determinar qué objetos deben tomarse para maximizar la utilidad sin exceder la capacidad de la mochila.

Fin de la Presentación

¿Preguntas?