

El muchacho de las aguas frescas.

Francisca Álvarez-Zermeño
Instituto Tecnológico Superior Zacatecas Occidente

INTRODUCCIÓN

En este trabajo describiremos de manera muy detallada el uso de software AMPL, donde llegamos a una respuesta mas rápidamente y muy específica, la cual nos ayuda a tomar decisiones mas específicas y con mas amplitud.

RESULTADOS

A continuación presentamos el enunciado y la solución mediante el software AMPL.

Problema

Un muchacho quiere abrir un puesto de bebidas. Su mamá le dice que no puede vender más de 4 galones de bebidas. El muchacho vende limonada y jugo de fruta. Dice que vende la limonada a 2 dolares el galón y el jugo de fruta a 1.5 el galón. La mamá del muchacho tiene solamente 90 rebanadas de limón y 6 libras de azúcar. Encuentra cuantos galones de cada bebida se pueden hacer para hacer la mayor cantidad de ganancia.

Maximizar: $f=2x+1.5y$ Sujeto a: $x+y \leq 4$ $30x+10y \leq 90$
 $x+2y \leq 6$ $x \geq 0$ $y \geq 0$

Solución

El símbolo “#” sirve para anotar comentarios, se debe omitirlo al ingresarlo en AMPL.:

El ; se agrega para cerrar el algoritmo.: #se agrega var a las variables

var limonada;

var fruta;

Enseguida se agrega la función objetivo.: #se agrega “maximize ganancia”

maximize ganancia: $2x+1.5y$

La palabra “subject to”: #se agrega subject to para agregar las restricciones.

Subject to

PRIMERA RESTRICCIÓN

c1: $x+y \leq 4$ #segunda restricción c2: $30x+10y \leq 90$ #tercera restricción c3: $x+2y \leq 6$ #cuarta restricción c4: $x \geq 0$ #quinta restricción c5: $y \geq 0$

La palabra “solve”: #se agrega para encontrar la solución optima

Solve;

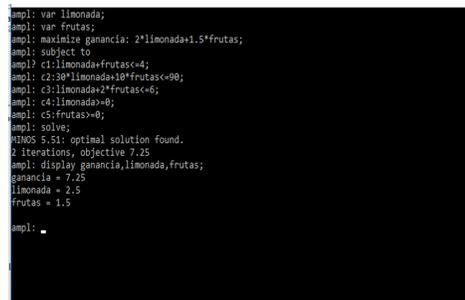
La palabra “display”: #se agrega para encontrar los resultados concretos.

display ganancia,limonada,fruta;

ganancia = 7.5

limonada = 2.5

fruta = 1.5



```
AMPL: var limonada;
AMPL: var fruta;
AMPL: maximize ganancia: 2*limonada+1.5*fruta;
AMPL: subject to
AMPL: c1:limonada+fruta<=4;
AMPL: c2:30*limonada+10*fruta<=90;
AMPL: c3:limonada+2*fruta<=6;
AMPL: c4:limonada>=0;
AMPL: c5:fruta>=0;
AMPL: solve;
AMPL: 5:51: optimal solution found.
2 iterations, objective 7.25
AMPL: display ganancia,limonada,fruta;
ganancia = 7.25
limonada = 2.5
fruta = 1.5
AMPL: .
```

Figure 1. Resultados .

Finalmente se pueden apreciar los resultados en la fig. 1:

CONCLUSIÓN

En la actualidad gracias a nuevos software se pueden realizar de manera mas simple problemas de programación lineal, ya que son muy eficientes y prácticos.

JUEGO DE PING-PONG

INTRODUCCIÓN

En este presente trabajo describiremos de manera muy detallada el uso del software AMPL, donde llegamos a una respuesta mas rápidamente y muy específica, la cual nos ayuda a tomar decisiones mas específicas y con mas amplitud.

RESULTADOS

A continuación presentamos el enunciado y la solución mediante el software AMPL.

Problema

Asume que quieres decidir entre formas alternas de pasar un día de 8 horas, esto es, quieres distribuir tu tiempo. Asume que se te hace 5 veces mas divertido jugar ping-pong que trabajar, pero también sientes que debes trabajar por lo menos 3 veces, tantas como jugaste ping-pong y ahora el problema es cuantas horas jugar y cuantas trabajar para maximizar la función objeto que es diversión.

Maximizar: $f=x+5y$ sujeto a: $x+y \leq 8$ $3y \leq x$ $xy \geq 0$

Solución

El símbolo “#” sirve para anotar comentarios, se debe omitirlo al ingresarlo en AMPL.:

El ; se agrega para cerrar el algoritmo.: #se agrega var con las variables

```
var trabajo;  
var juego;
```

Enseguida se agrega la función objetivo.: #se agrega “maximize ganancia”

```
maximize ganancia: x+5y
```

La palabra “subject to”: #se agrega subject to para agregar las restricciones

```
Subject to
```

PRIMERA RESTRICCIÓN

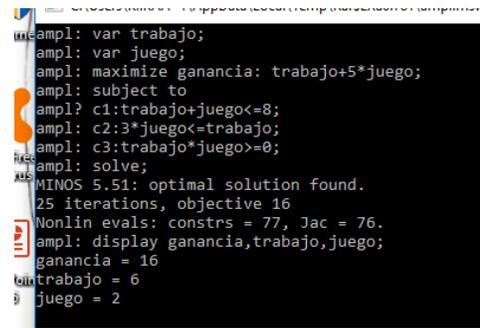
```
c1:x+y<=8 #segunda restricción c2:3y<=x #tercera restricción c3:xy>=0
```

La palabra “solve”: #se agrega para encontrar la solución optima

```
Solve;
```

La palabra “display”: Se agrega para encontrar los resultados concretos.

```
display ganancia,trabajo,juego;  
ganancia =  
trabajo =  
juego =
```



```
ampl: var trabajo;  
ampl: var juego;  
ampl: maximize ganancia: trabajo+5*juego;  
ampl: subject to  
ampl? c1:trabajo+juego<=8;  
ampl: c2:3*juego<=trabajo;  
ampl: c3:trabajo*juego>=0;  
ampl: solve;  
MINOS 5.51: optimal solution found.  
25 iterations, objective 16  
Nonlin evals: conhrs = 77, Jac = 76.  
ampl: display ganancia,trabajo,juego;  
ganancia = 16  
trabajo = 6  
juego = 2
```

Figure 2. Solución Ping-pong

Finalmente se pueden apreciar los resultados en la fig. 2.:

CONCLUSIÓN

En la actualidad gracias a nuevos software se pueden realizar de manera mas simple problemas de programación lineal ya que son muy eficientes y prácticos.