Herramienta Gráfica para Resolución de Problemas de Optimización Lineal

5 de junio de 2013

Resumen

Esta herramienta es una implementación del algoritmo Simplex con el método de la gran 'M'. Se desarrolló bajo los lenguajes de programación Python 2.7 y C++, usando las librerías de GTK y Glade

1. Introducción

La siguiente pantalla muestra nuestra interfaz recién iniciada.

Figura 1: Pantalla de inicio

Aquí podemos ver que nuestra interfaz tiene varios elementos que son:

Entrada Es una vista de texto que nos permite capturar el problema a optimizar con un formato amigable al usuario familiarizado con los conceptos de optimización lineal.

Resolver Este botón manda a llamar todo el proceso para solucionar el problema de entrada.

Ejemplo Permite mostrar al usuario un ejemplo del formato de entrada al usuario no familiarizado con la aplicación.

Limpiar Con este botón limpiamos el buffer de entrada y de salida.

Salida Es una vista de texto no modificable donde mostramos la solución óptima, o un mensaje si hubo algún error de formato de entrada o si el problema no tiene solución.

2. Casos de Uso

2.1. Problema con Solución

Ponemos como ejemplo el siguiente problema:

$$\begin{array}{lrcl} \text{Maiximizar} & -0.5x + 3y + z + 4w & = & p \\ & x + y + z + w & \leq & 40 \\ & 2x + y - z - w & \geq & 10 \\ & 2w - y & = & 10 \end{array}$$

En este caso el formato de entrada para nuestro programa será el que se muestra en la Figura 2.

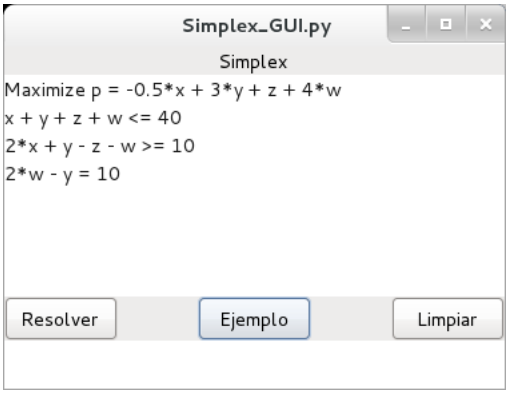

Figura 2: Pantalla de formato de entrada

Es necesario hacer notar que la entrada debe contener la palabra "Maximize" o "Minimize" seguido de la función a optimizar en la primera línea y en las siguientes líneas las restricciones, una por línea. También vemos que la multiplicación debe ser denotada con el operador asterisco (*).

En este momento el usuario tiene la opción de resolver el problema mediante el botón "Resolver" o limpiar el buffer de entrada mediante el botón "Limpiar".

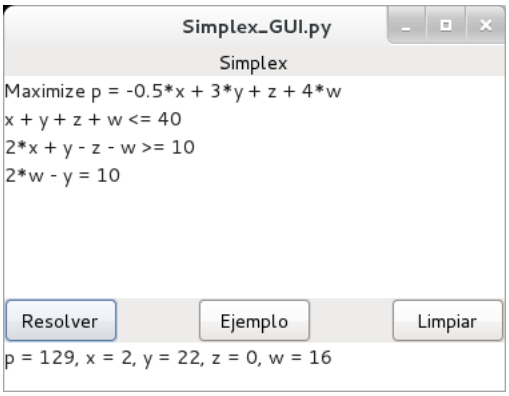

Figura 3: Pantalla de solución

El botón "Resolver" internamente activará el código que analiza la entrada y la traduce para que nuestro código del algoritmo Simplex pueda facilmente operar con ella mediante un formato de matrices y después regresarnos la solución que mostramos en nuestra vista de texto de salida. La Figura 3 presenta el resultado.

2.2. Problema sin Solución

En caso de que el problema no tenga solución óptima se mostrará una pantalla similar a la siguiente:

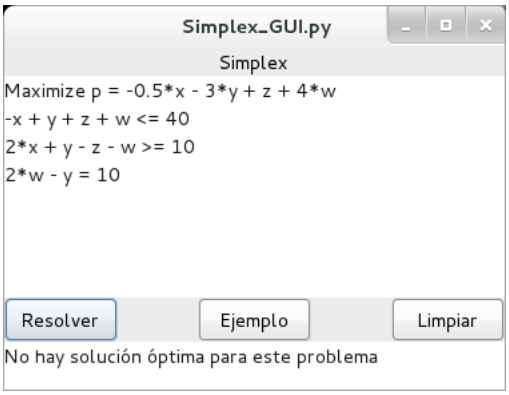

Figura 4: Pantalla de problema sin solución

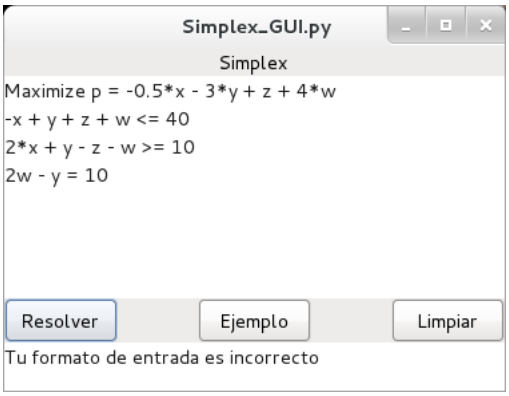

Figura 5: Pantalla de error de formato

2.3. Error de Formato de Entrada

Por último mostramos, en la siguiente figura, el comportamiento de nuestro programa en el caso de que haya un error en el formato de entrada

3. Sobre la Implementación del Algoritmo Simplex

El núcleo del programa trabaja de la siguiente manera:

- Siempre maximiza la función objetivo.
- Implementa el método de la Gran 'M'.
- La función recibe:
  1. Un vector con los coeficientes de la función objetivo, donde el i-ésimo elemento corresponde al coeficiente de la i-ésima variable.
  2. Una matriz, donde el elemento (i, j) corresponde al coeficiente de la j-ésima variable en la i-ésima restricción.
  3. Una matriz, donde el elemento $(i, 0)$ corresponde al coeficiente de la i-ésima restricción y el elemento $(i, 1)$ es un -1 si la restricción es un menor qué, un 0 si es un igual qué o un 1 si es un mayor qué.
- La función regresa:
  1. Un vector donde el primer elemento corresponde al valor de la función objetivo y el elemento $(i+1)$-ésimo corresponde al valor de la i-ésima variable.

La entrada y salida de nuestro código Simplex es analizada por un código intermediaro implementado en Python que se encarga de darle el formato deseado.