Implementación del algoritmo

Primero hablaremos sobre el tipo de problemas de programación lineal que el programa acepta, después pasaremos a hablar del algoritmo y los detalles de su implementación, y en la última parte hablaremos de cómo utilizar el algoritmo.

Restricciones sobre el problema de programación que se puede resolver

1. El problema puede optimizar a un máximo o a un mínimo
2. El problema acepta variables positivas y variables libres.
3. Las expresión de la función objetivo debe ser lineal.
4. Las restricciones deben ser de la forma , donde exp es una combinación lineal de las variables y b es un coeficiente no negativo.

Dicho esto, no se aceptan valores absolutos ni funciones máximo/mínimo dentro de la función objetivo o las restricciones. Tampoco se acepta que las restricciones tengan variables de ambos lados de la desigualdad ni que el coeficiente del lado derecho sea negativo.

En otras palabras, el problema que se acepta es de la forma:

$$Min/Máx\ z = c * x^T$$

sujeto a

$$A_{-} x^T \leq b^{-}$$

$$A_{=} x^T = b^{=}$$

$$A_{+} x^T \geq b^{+}$$

donde x, c son vectores con coeficientes en los reales, con x las variables de decisión. b^{-}, $b^{=}$ y b^{+} son vectores no negativos de tamaño ajustado a las restricciones. Por otro lado, A es una matriz con coeficientes en los reales que contiene los coeficientes de las restricciones.

Detalles de implementación

Se implementa el algoritmo simplex utilizando el método de las dos fases. El algoritmo se desarrolló en C++ y se desarrolló una interfaz de usuario en C#.

Los archivos incluidos son:

- **lectura.h:** contiene las funciones que ayudan a la lectura del problema y su conversión a una matriz simplex que representa el problema en su forma estándar (método principal es *read*);
- **simplex.h:** contiene las funciones principales del algoritmo simplex y del método de las dos fases (método principal es *twoPhaseSimplexMethod*);

- **controlador.h:** se encarga de coordinar la ejecución de los programas anteriores y desplegar un resultado (método principal es *main*);
- **library.h:** tiene funciones generales que ayudaron en alguno(s) de los programas;
- **fraction.h:** implementa una estructura (clase pública) para trabajar con fracciones y que los resultados sean precisos si no se trabaja con coeficientes reales.

El algoritmo siempre recurre al método de las dos fases, aún si no hay necesidad, por simplicidad del código. La ejecución sigue la idea que ya anticipan los archivos mencionados:

1. se lee de un archivo de texto el problema y se construye una matriz que represente el problema en su forma estándar
2. se transforma el método a la primera fase y se ejecuta el algoritmo simplex
3. regresa la matriz a sus dimensiones iniciales y continua con la segunda fase si el espacio de soluciones es no vacío.
4. Si se llegó a un óptimo se despliega el valor de la función objetivo y el estado de las variables. Si no, se da un mensaje de que el espacio de solución era vacío o que no era acotado.

Limitaciones de la implementación

Se trabaja sólo con coeficientes racionales. Inicialmente se prefirió así para que los resultados fueran exactos, lo cuál no sería siempre el caso si se trabaja con números flotantes. Trabajar con racionales abarca la mayoría de los problemas e incluso si se trabaja con números reales, se pueden aproximar con racionales.

Por otro lado, al código no le hacen falta muchas modificaciones para trabajar con variables de tipo float o double. Pensando en usar también estas variables fue que todas las funciones de la clase simplex y read trabajan con variables de tipo genérico. Sin embargo, la falta de tiempo ya no permitió terminar los detalles para incluir este otro tipo de variables.

Otra cosa a tomar en cuenta es que pudieran ocurrir desbordamientos. La clase fraction trabaja con el numerador y denominador siendo enteros signados de 64 bits. Si una multiplicación, suma o resta incurriera en un desbordamiento, el programa no lo detectaría y generaría resultados erróneos.

Uso del programa

Para usar la interfaz gráfica

Opción 1: con la terminal

1. Descarga InterfazSimplex.zip
2. Descomprime dicha carpeta
3. Abre la terminal y, usando el comando `cd`, sitúate en la carpeta ..\InterfazSimplex\bin\Debug
4. Corre el comando `.\InterfazSimplex` (si se tiene un sistema operativo MacOS el comando es `./InterfazSimplex`)

Opción 2: con Visual Studio

1. Descarga InterfazSimplex.zip
2. Descomprime dicha carpeta
3. Abre InterfazSimplex.sln
4. Dale click al botón de Iniciar

Para no usar la interfaz, y sólo utilizarlo mediante la terminal y archivos de texto

Para el sistema operativo **Windows**:

1. Siga el formato del archivo problem.txt para describir su problema en un archivo de texto con el nombre de su preferencia. La carpeta ejemplo.txt contiene un ejemplo de un problema de programación y el archivo ejemploResp.txt contiene su solución.
2. Posiciónese en la terminal en la carpeta donde tiene el archivo controlador.exe
   Esto se hace con el comando :
   cd path
   donde path es la dirección de la carpeta donde tiene el archivo controlador.exe.
3. Ejecute en la terminal el comando:
   .\controlador < input.txt > output.txt
   Esto generará en la misma carpeta el archivo output.txt con la descripción de la solución del problema.
   Los archivos no tienen que llamarse input.txt ni output.txt, pueden tener el nombre que usted desee. Se les dio ese nombre para ejemplificar el comando.

La única diferencia para **MacOs** es el paso 3. El comando a ejecutar es:
./controlador < input.txt > output.txt