

# Proyecto 4: Otro SIMPLEX más

## Parte 1

**Miércoles 12 de Noviembre**

### I. DESCRIPCIÓN GENERAL

Se debe programar el algoritmo de SIMPLEX visto en clases para resolver problemas de Programación Lineal (LP). En esta primera versión, sólo se manejarán restricciones de la forma  $\leq$ , pero su interfaz debe estar preparada para las otras posibilidades (*i.e.*,  $=$  y  $\geq$ ). La salida será un documento  $\LaTeX$  de gran calidad<sup>1</sup> que incluya el problema original, las tablas Símplex inicial y final, la solución final y, si el usuario lo solicita, todas las tablas intermedias. Habrá una interfaz gráfica preparada con GTK y Glade para el ingreso de los problemas. Toda la programación debe realizarse en C sobre Linux. No se pueden cambiar las especificaciones de este documento.

### II. ENTRADA

Usando una interfaz gráfica apropiada, el usuario ingresará el modelo del problema que deba ser resuelto. Se manejarán un máximo de 15 variables y 15 restricciones, y un mínimo de 2 variables y 2 restricciones. La interfaz gráfica sólo desplegará *widgets* para la cantidad de variables y restricciones indicadas por el usuario.

Los datos a ingresar son:

- **Nombre del problema.**
- **Cantidad de Variables.**
- **Cantidad de Restricciones.**
- **Nombres de las variables.** Dado que la salida final es código  $\LaTeX$  estos nombres se pueden interpretar como hileras  $\LaTeX$  que serán insertadas en el documento final. Por defecto, estos nombres serán  $x_1$ ,  $x_2$ , etc. - nótese el uso de subíndices, pero el usuario podrá editarlas.
- **Función objetivo.** Hacer una interfaz robusta que solicite el verbo (Maximizar o Minimizar) y los coeficientes de la ecuación lineal. Estos coeficientes son números reales que por defecto valen cero.
- **Restricciones.** Con una interfaz idéntica a la de la Función Objetivo, se solicitan los coeficientes de la ecuación lineal asociada a cada restricción. Para esta versión todas las restricciones serán desigualdades  $\leq$ , pero la interfaz mostrará desde ya las opciones “ $\geq$ ” y “ $=$ ” (aunque deshabilitadas). Los lados derechos son números reales mayores o iguales a cero.

Cuando el usuario considere que su problema ya está listo, presionará algún botón para arrancar la ejecución del algoritmo. También habrá forma de indicar si el usuario desea las tablas intermedias (por defecto no se mostrarán).

### III. EJECUCIÓN

Se implementará el algoritmo Símplex visto en clase, aunque sólo para restricciones de la forma “ $\leq$ ”. Si incluye otras restricciones o utiliza otro algoritmo su trabajo será calificado con un cero.

Se deben manejar apropiadamente los siguientes casos:

1. **Maximización o Minimización.** Si fuera de Minimización serán manejados con el segundo algoritmo visto en clases. Esto será evidente en las tablas intermedias.
2. **Soluciones Múltiples.** Si su programa detecta que hay soluciones múltiples, debe pivotar una vez más para tener una segunda solución. Ambas tablas finales serán mostradas, junto con una explicación detallada de la situación. Se deben reportar las dos soluciones halladas y al menos otras 3 soluciones adicionales, junto con la ecuación que permite hallar otras soluciones.
3. **Problemas no acotados.** Reportarlos y explicar detalladamente la situación en el documento.
4. **Problemas degenerados.** Reportarlos y explicar detalladamente la situación en el documento. Mencionar en el documento cómo resolvieron los empates de fracciones. Es improbable que su programa se encicle por esta situación.

**Trabajo extra opcional 1:** haga que su programa maneje apropiadamente “*La Pesadilla de Dantzig*” o similares.

5. **Problemas no factibles.** Para esta versión esta situación nunca se presentará.

### IV. SALIDA

Se debe generar un archivo “.tex” que será compilado con `pdflatex` o algún equivalente para que se produzca un archivo “.pdf” que debe ser desplegado usando `evince` (o algún equivalente) en modo presentación. **La invocación de todos estos comandos debe realizarse internamente desde su programa, de manera transparente para el usuario.** Deben quedar disponibles tanto el pdf como el fuente de  $\LaTeX$ . Recuerde que su salida final será una **excelente** documento preparado con  $\LaTeX$  (no Beamer).

En su documento deben aparecer los siguientes elementos:

<sup>1</sup>No puede ser una presentación Beamer

1. **Portada:** se identifica claramente a los miembros del grupo, el curso, el semestre y el nombre del proyecto.
2. **Algoritmo Símplex:** una explicación de la historia y propiedades del algoritmo Símplex.
3. **Problema original:** presentar de manera matemática el problema original (incluir el nombre dado al problema por el usuario). Se deben respetar los nombres de variables escogidos por el usuario.
4. **Tabla Inicial:** se muestra la Tabla Símplex inicial. Esta incluye las variables iniciales y las variables de holgura.
5. **Tablas Intermedias:** si el usuario lo solicita, se deben presentar las tablas intermedias de la ejecución del algoritmo. Debe aparecer marcada la columna seleccionada, los cálculos de fracciones y el pivote. Se recomienda el uso de colores y/o figuras geométricas apropiadas (*i.e.*, flechitas, cajitas, etc.). Se le dará mucha importancia a la calidad de estas tablas intermedias.
6. **Tabla Final:** se muestra la última Tabla Símplex encontrada. Recuerden que en el caso de soluciones múltiples habrá 2 tablas finales.
7. **Solución:** se da la solución óptima encontrada con todo detalle. Si hubiera algún caso particular (*e.g.*, soluciones múltiples, problema no acotado, problema no factible, problema degenerado, etc.) deben aparecer las explicaciones apropiadas.

**Trabajo extra opcional 2:** si el problema es de 2 variables muestre un gráfico con todas las restricciones y la región factible rellena en algún color diferente.

**Trabajo extra opcional 3:** si el problema es de 3 variables muestre un gráfico con 3 ejes que muestre el cuerpo Símplex tridimensional.

- Se debe usar GTK y Glade. Es inaceptable hacer una interfaz de texto desplegada en un *widget*.
- No debe dar “Segmentation Fault” bajo ninguna circunstancia.
- La demostración debe hacerse en una máquina que levante Linux de manera real (puede ser dual), es decir no usar máquinas virtuales. No se puede levantar de un disco o memoria externa. No se pueden prestar máquinas entre grupos.

## VI. FECHA DE ENTREGA

Revisiones a las 11:30am el **Miércoles 12 de Noviembre** en la oficina del profesor. Mande además un .tgz con todo lo necesario (fuentes, makefile, readme, etc.) a [torresrojas.cursos.04@gmail.com](mailto:torresrojas.cursos.04@gmail.com). Ponga como subject: [IO] Proyecto 4 - Fulano - Mengano, donde Fulano y Mengano son los miembros del grupo.

## V. REQUISITOS INDISPENSABLES

La ausencia de uno solo de los siguientes requisitos vuelve al proyecto “no revisable” y recibe un 0 de calificación inmediata:

- La colaboración entre grupos se considera fraude académico.
- Todo el código debe estar escrito en C
- Recuerde que usar IA o de proyectos de otras personas para completar cualquier parte de su trabajo es fraude académico y tendrá muy serias consecuencias.
- El proyecto debe compilar y ejecutar en Linux. Todo debe estar **integrado**, explicaciones del tipo “*todo está bien pero no pudimos pegarlo*”<sup>2</sup> provocan la cancelación automática de la revisión.
- Para el momento de la revisión, su programa no debe tener “prints” de depuración. Luce muy poco profesional.
- Todas las interfaces deben ser gráficas, robustas y de mucha calidad.

<sup>2</sup>esto incluye los supuestos casos cuando alguien del grupo de trabajo no hizo su parte – el profesor no está interesado en sus problemas de organización.