Programación II. Curso 2020-2021
Trabajo Obligatorio

Introducción

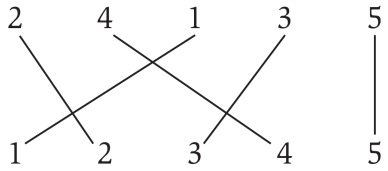
Numerosas aplicaciones web hacen uso de técnicas conocidas como *filtrado colaborativo*, en las que intentan buscar, dentro de su base de datos, otros usuarios que tengan unos gustos similares a los tuyos. Por ejemplo, las páginas web de IMDb (`www.imdb.com`) o FilmAffinity (`www.filmaffinity.es`) usan este tipo de técnicas para recomendarte películas que puedan ser de tu interés en función de las votaciones que hayas hecho a otras películas. Para ello, identifican a posibles usuarios con gustos similares comparando tus calificaciones con las suyas. Algo similar sucede también con metabuscadores de Internet, que intentan sintetizar los resultados buscando las similitudes y diferencias entre los diferentes resultados de búsqueda obtenidos de otros buscadores de Internet. En ambos casos se está tratando con el problema de comparación de dos rankings: *¿cómo se puede cuantificar cómo de similares son dos rankings?*

Considera que se quiere comparar tu ranking de un conjunto de n películas con el ranking del mismo conjunto de otro usuario cualquiera. Una forma natural de poder cuantificar la similitud entre estos dos rankings sería numerar cada una de las películas de 1 a n según tu ranking, y después observar el ranking del otro usuario conforme a la numeración inicial y contar cuántos pares están fuera de orden.

Dicho de otro modo, se está dando un valor a cómo de lejos (o de cerca) se encuentra un vector de enteros de estar ordenado. Este valor se conoce como el *número de inversiones* en un vector de enteros. Si el vector está ordenado, su número de inversiones será 0. Sin embargo, si el vector está ordenador en orden inverso, el número de inversiones será máximo.

Formalmente hablando, dado un vector v de enteros, dos elementos $v[i]$ y $v[j]$ forman una inversión si $i<j \wedge v[i]>v[j]$. Así, el número de inversiones de un vector v de enteros se define como el número de pares (i, j) tal que $0 \leq i<j<\#v \wedge v[i]>v[j]$.

Veámoslo con un ejemplo. Supón tu vector de preferencias $v_0 = [1, 2, 3, 4, 5]$ y el vector de otro usuario como $v = [2, 4, 1, 3, 5]$. Este segundo vector tiene 3 inversiones: $(0, 2)$ (ya que $v[0]>v[2]$), $(1, 2)$ ($v[1]>v[2]$) y $(1, 3)$ ($v[1]>v[3]$). Gráficamente, cada inversión es un cruce de las líneas que unen los mismos números de los vectores:



En este trabajo obligatorio de la asignatura se propone **especificar y diseñar algoritmos para calcular el número de inversiones en un vector de enteros**. Se propone calcular el número de inversiones en un vector **mediante dos estrategias distintas: *Fuerza Bruta* y *Divide y Vencerás***.

El conteo de inversiones puede realizarse de manera natural, como se ha indicado anteriormente, o bien de forma algo más eficiente. Una forma eficiente de contar las inversiones sería mediante el uso de un método de ordenación por mezcla. Si el número de inversiones en la mitad izquierda de un vector v es inv_1 y el número de inversiones en la mitad derecha de v es inv_2, el número total de inversiones sería la suma $inv_1 + inv_2 + inv_{\text{merge}}$, donde inv_{merge} es el número de inversiones resultado de la mezcla de ambas partes.

En concreto, las tareas a realizar son las siguientes:

Tarea 1. Especificar formalmente las funciones declaradas en el fichero «`cuentaInv.hpp`».

Tarea 2. Diseñar e implementar la estrategia de *Fuerza Bruta* para resolver el problema. El coste asintótico en tiempo será $\mathcal{O}(n^2)$, donde n es el número de datos del vector.

Tarea 3. Diseñar e implementar la estrategia de *Divide y Vencerás* para resolver el problema. El coste asintótico en tiempo será $\mathcal{O}(n \cdot log(n))$.

Tarea 4. Ejecutar ambas estrategias para vectores de tamaño entre 100 y 10 000 enteros, generados de forma aleatoria.

Tarea 5. Generar gráficas que muestren para cada una de las estrategias el tiempo de ejecución en función del tamaño del vector. Las gráficas se pueden realizar de la manera que se desee. Una posibilidad consiste en guardar las datos que se quieren mostrar en un fichero de texto y después realizar una llamada al sistema desde el programa. Por ejemplo, si se han guardado los siguientes datos en el fichero de texto «`tnumifb.txt`» (cada línea representa un tamaño de vector y el tiempo de ejecución en segundos, separados por un tabulador), que representan los tiempos de ejecución de la estrategia de *Fuerza Bruta*:

```
10      0
1010    0.015
2010    0
3010    0.015
4010    0.031
5010    0.063
```

Y teniendo un fichero de texto similar, «`tnumidv.txt`», que contiene los tiempos de ejecución de la estrategia de *Divide y Vencerás*, entonces la instrucción:

```
system("gnuplot plot_datos_t.plot");
```

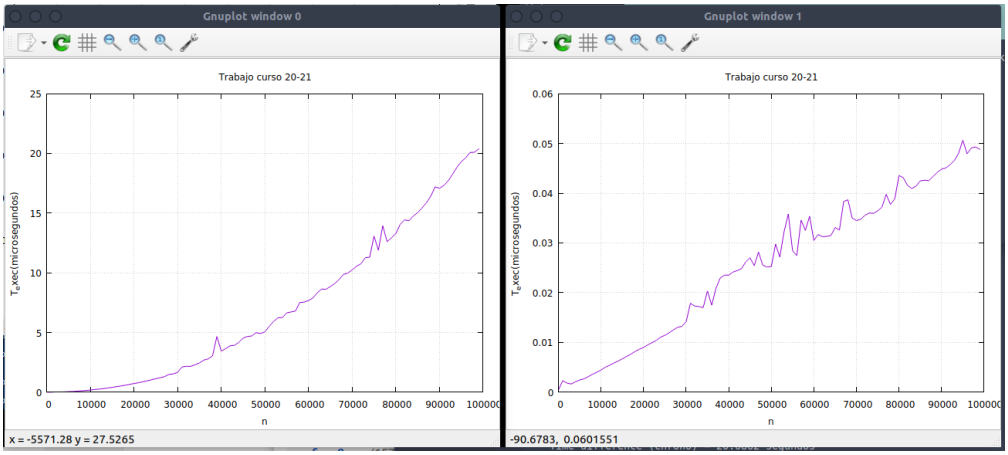


Figura 1: Ventanas de `gnuplot` mostradas tras ejecutar el código «`prueba_gnuplot.cpp`» compilado.

realiza una llamada al sistema que ejecuta el comando `gnuplot` para generar una gráfica mediante el fichero de órdenes de `gnuplot` llamado «`plot_datos_t.plot`». Este fichero de órdenes se encarga de leer los fichero «`tnumifb.txt`» y «`tnumidv.txt`» para generar y mostrar dos gráficas con los datos dados. Como resultado de la ejecución de esta instrucción, deberías de obtener unas ventanas similares a la que se muestra en la Figura 1.

Puedes encontrar más información sobre `gnuplot` en `http://www.gnuplot.info/` y en `http://www.sromero.org/wiki/linux/aplicaciones/gnuplot`. Para instalar `gnuplot` en tu sistema, consulta el Apéndice A. Adicionalmente, se te ha proporcionado el fichero «`prueba_gnuplot.cpp`» para probar si los has instalado correctamente en tu sistema. La ejecución de ese código compilado debería de mostrarte una gráfica idéntica a la mostrada anteriormente.

Tarea 6. Describir brevemente en un fichero «`comentarios.pdf`» las gráficas obtenidas y una **concisa discusión sobre las ventajas de cada estrategia**.

Notas:

- En Moodle se proporciona el fichero «`cuentaInv.hpp`».
- Aunque se puede hacer el uso que se desee de este fichero, **obligatoriamente deberán implementarse las funciones especificadas**.

Forma y fecha de entrega

- El trabajo se realizará individualmente.
- Los apartados relativos al diseño se entregarán vía *Moodle* en una tarea habilitada a tal efecto.

- Se deberá entregar un fichero con nombre «`AAAAAA.zip`» (donde AAAAAA es tu NIA) que contenga los siguientes ficheros:
  - Todos los ficheros `.hpp` y `.cpp`.
  - Un fichero «`Makefile`» que los compile y genere el ejecutable «`costeInv`», que cuenta inversiones para distintos tamaños de vectores y genera las gráficas con los costes en tiempo.
  - El fichero «`comentarios.pdf`», que describe brevemente los costes temporales de las estrategias.
- Las especificaciones formales deben escribirse en el fichero «`cuentaInv.hpp`».
- El último día para entregar el trabajo es el **domingo 6 de junio de 2021.**

Sobre el presente trabajo obligatorio

La evaluación de la asignatura en la primera convocatoria consta de tres pruebas:

Prueba 1. Examen escrito con un peso del 50 %. Para aprobar la asignatura es necesaria una calificación mínima de 4.0 puntos en él.

Prueba 2. Examen práctico en laboratorio con un peso del 25 %. No se exige calificación mínima.

Prueba 3. El presente trabajo obligatorio con un peso del 25 %. No se exige calificación mínima.

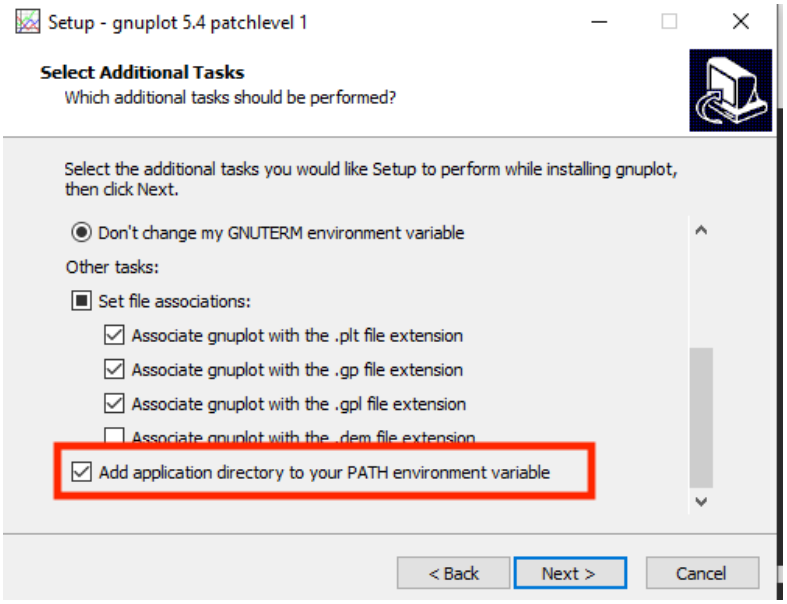
A. Instalación de `gnuplot`

Este apéndice da unas instrucciones breves sobre la instalación de `gnuplot` en sistemas Windows 10, sistemas macOS y sistemas GNU/Linux.

A.1. En Windows 10

Para instalar `gnuplot` en Windows 10, tienes que seguir los siguientes pasos:

1. Descarga la versión de `gnuplot` 5.4.1 del siguiente enlace: `https://sourceforge.net/projects/gnuplot/files/gnuplot/5.4.1/gp541-win64-mingw.exe/download`
2. Realiza el proceso de instalación y en la última ventana selecciona la opción `Add application directory to your PATH environment variable` como se muestra en la siguiente imagen:



3. Reinicia tu sesión de usuario.
4. Compila el fichero de pruebas «`prueba_gnuplot.cpp`» y ejecútalo para visualizar la gráfica de ejemplo. Si todo ha funcionado correctamente, deberás de ver una ventana similar a la Figura 1.

A.2. En macOS

Para instalar `gnuplot` en macOS 10.15 (Catalina) o superior, tienes que seguir los siguientes pasos:

1. La forma más sencilla de instalar `gnuplot` es instalar en primer lugar `Homebrew`, que es un sistema de gestión de paquetes que simplifica la instalación, actualización

y eliminación de paquetes de software de GNU/Linux en entornos macOS. Las instrucciones de instalación y los requisitos mínimos están disponibles en `https://brew.sh/`. Su instalación es tan sencilla como ejecutar en una terminal (pincha en el icono de la lupa en esquina superior derecha [`Spotlight`], busca y ejecuta la aplicación `Terminal`) la siguiente orden:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Esta orden descargará e instalará `Homebrew` en tu sistema macOS, informando en cada paso de instalación de qué está haciendo y del resultado.

2. Tras instalar `Homebrew`, ejecuta la siguiente orden en la terminal para instalar `gnuplot`:

   ```
   brew install gnuplot
   ```

   Verás que automáticamente se descargarán una serie de paquetes de software de Internet y se instalarán en tu sistema, informando apropiadamente en caso de algún error.

3. Compila el fichero de pruebas «`prueba_gnuplot.cpp`» y ejecútalo para visualizar la gráfica de ejemplo. Si todo ha funcionado correctamente, deberás de ver una ventana similar a la Figura 1.

A.3. En GNU/Linux

Para instalar `gnuplot` en GNU/Linux, simplemente instala el paquete `gnuplot` desde tu gestor de paquetes:

- Sistemas basados en Debian: `apt-get install gnuplot`
- Sistemas basados en Fedora/CentOS: `yum install gnuplot`

Después de instalarlo, compila el fichero de pruebas «`prueba_gnuplot.cpp`» y ejecútalo para visualizar la gráfica de ejemplo. Si todo ha funcionado correctamente, deberás de ver una ventana similar a la Figura 1.