INF-221 Algoritmos y Complejidad, 2020-1

Tarea 1

Profesor: Diego Arroyuelo
Ayudantes: Tomás Berrios, Gabriel Carmona
tomas.berrios@sansano.usm.cl,
gabriel.carmonat@sansano.usm.cl

Fecha de Entrega: 24 de junio, 2020
Plazo máximo de entrega: 5 días.

Reglas del Juego

La presente tarea debe hacerse en grupos de 3 personas. Toda excepción a esta regla debe ser conversada con los ayudantes **ANTES** de comenzar la tarea. No se permiten de ninguna manera grupos de más de 3 personas. Pueden usarse los lenguajes de programación C, C++, Python, y Java.

1. Joins en Bases de Datos [35 %]

En bases de datos relacionales, la operación join (denotada con $\bowtie$) permite combinar tuplas de distintas relaciones, formando tuplas más grandes. Dado que la normalización en bases de datos relacionales genera varias relaciones, la operacion join permite (entre otras cosas) obtener la información original (es decir, antes de la normalización). Esto hace a los joins una de las operaciones más típicas y empleadas en bases de datos relacionales. Por ejemplo, si tenemos las relaciones $R(A, B)$ (es decir, que tiene atributos A y B) y $S(B, C)$ (con atributos B y C), la operación $R(A, B) \bowtie S(B, C)$ produce como resultado una relación $T(A, B, C)$. En este caso, el atributo B es conocido como *atributo de join*: es ese atributo el que se usa para combinar tuplas de R y S. Vamos a considerar únicamente el caso de join natural en esta tarea. Esto eso, un join que se realiza sólo teniendo en cuenta la igualdad de los valores de los atributos de join. Por cada tupla $(a_i, b_j) \in R$ y $(b_j, c_k) \in S$, el join produce la tupla (a_i, b_j, c_k) en T. Considere el siguiente ejemplo de join:

R

$(A$	$B)$
1	2
2	4
4	1
5	2
4	5
3	4
3	5

$\bowtie$

S

$(B$	$C)$
3	1
2	3
5	3

$=$

T

$(A$	B	$C)$
1	2	3
5	2	3
4	5	3
3	5	3

La operación join está definida no sólo para relaciones binarias (como lo hemos explicado), sino que puede extenderse a relaciones de cualquier aridad, y cualquier cantidad de atributos de join. Por ejemplo, podemos tener $R(A, B, C) \bowtie S(A, C, D)$, que producirá la relación $T(A, B, C, D)$. La tarea consiste en implementar un algoritmo de fuerza bruta para resolver la operación join. Los atributos de cada relación serán representados por números enteros.

Formato de Entrada

La entrada de datos será a través de la entrada estándar (`stdin`). La primera línea de la entrada contiene un único número entero `N`, indicando la cantidad de atributos de la relación R. Puede asumir que $1 \leq \texttt{N} \leq 100$. La siguiente línea de la entrada contiene `N` valores enteros, separados entre si por un único espacio en blanco, indicando los atributos de la relación R. Los valores están ordenados de forma creciente. La siguiente línea contiene un valor entero `NR`, tal que $1 \leq \texttt{NR} \leq 10,000$, e indica la cantidad de tuplas de la relación R. Luego, le siguen `NR` líneas, cada una con `N` valores enteros (separados entre sí por un único espacio), correspondientes a las tuplas de la relación R. A continuación, le sigue una línea que contiene un único número entero `M`, indicando la cantidad de atributos de la relación S. Puede asumir que $1 \leq \texttt{M} \leq 100$. La siguiente línea de la entrada contiene `M` valores enteros, separados entre si por un único espacio en blanco, indicando los atributos de la relación S. Los valores están ordenados de forma creciente. La siguiente línea contiene un valor entero `NS`, tal que $1 \leq \texttt{NS} \leq 10,000$, e indica la cantidad de tuplas de la relación S. Luego, le siguen `NS` líneas, cada una con `M` valores enteros (separados entre sí por un único espacio), correspondientes a las tuplas de la relación S.

Un ejemplo (que replica el ejemplo mostrado anteriormente) es el siguiente:

```
2
0 1
7
1 2
2 4
4 1
5 2
4 5
3 4
3 5
2
1 2
3
3 1
2 3
5 3
```

Formato de Salida

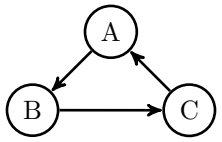
La salida se hará a través de la salida estándar (`stdout`). Debe mostrar el resultado de la operación join sobre las dos relaciones dadas en la entrada. El formato es el siguiente: la primera línea debe contener un valor `P`, indicando la cantidad de atributos de la relación resultante del join. Luego, le sigue una línea que contiene `P` valores enteros, separados entre si por un único espacio, y ordenados de forma creciente. Esos valores indican los atributos de la relación resultante. Luego, le sigue una línea que contiene un único valor entero `NT`, que es la cantidad de tuplas del join. Finalmente, se tienen `NT` líneas, cada una con `P` valores enteros separados por un único espacio, correspondientes a las tuplas del join.

La salida correspondiente al ejemplo mostrado anteriormente es:

```
3
0 1 2
4
1 2 3
5 2 3
4 5 3
3 5 3
```

2. Contando Triángulos en Grafos [15 %]

Sea $G = (V, E)$ un grafo dirigido con conjunto de vértices V y aristas E. En esta parte, debe implementar un algoritmo que use el algoritmo de join de la sección anterior para contar la cantidad de triángulos que tiene G. Esto es una aplicación importante en muchas aplicaciones, como por ejemplo redes sociales y biología. Un triángulo es un patrón del siguiente tipo:



Suponga que los arcos del grafo estarán representado por una relación binaria $E(A, B)$, en donde por cada arco $a \to b$ en el grafo, tenemos la tupla $(a, b) \in E$.

La entrada de datos se hará mediante la entrada estándar (`stdin`). Cada línea contiene dos valores enteros, que representan un arco (dirigido) del grafo. la entrada es terminada con `EOF`. La salida se hará a través de la salida estándar (`stdout`). Se debe imprimir una única línea que contiene la cantidad de triángulos del grafo dado en la entrada.

3. Algoritmo de Multiplicación de Matrices [50 %]

Dadas dos matrices A y B, ambas de dimensión $n \times n$, el algoritmo del tipo Dividir y Conquistar de Strassen permite calcular $A \times B$ en tiempo $\Theta(n^{2,81})$, mientras que el algoritmo tradicional de multiplicación de matrices es de tiempo $\Theta(n^3)$. El objetivo de esta parte de la tarea es estudiar e implementar el algoritmo de Strassen, para poder aplicarlo en las siguientes situaciones:

Problema 1: Suponga que quiere multiplicar dos matrices X e Y, de dimensiones $kn \times n$ y $n \times kn$, respectivamente. Implemente un algoritmo para llevar a cabo dicha multiplicación, usando el algoritmo de Strassen como subrutina.

Problema 2: Idem al problema anterior, pero ahora asumiendo que las matrices tienen dimensión $n \times kn$ y $kn \times n$, respectivamente.

Escriba un informe (idealmente usando LaTeX, aunque no es obligatorio su uso) en el que muestre la comparación del algoritmo de Strassen y el tradicional para multiplicar matrices, para diferentes valores de k y n. Resuma sus experimentos en unos pocos gráficos (su informe no debería tener más de 2 páginas). Agregue observaciones y conclusiones de sus experimentos, así como una explicación clara de cómo se realizaron los experimentos (quien lea su informe, debería ser capaz de replicar sus experimentos sin problemas).

Formato de Entrada

La entrada de datos será a través de la entrada estándar (`stdin`). Para estos dos problemas, la entrada comienza con una línea que contiene dos números enteros, N y k. Luego le siguen los valores que componen a las dos matrices, escritos por filas y los valores separados entre si por un único espacio (primero las filas de la matriz A, y a continuación las de la matriz B). Asuma que las matrices están formadas por valores enteros.

Formato de Salida

La salida se hará a través de la salida estándar (`stdout`). La primera línea de la salida contiene dos números, N y M, indicando las dimensiones de la matriz resultante de la multiplicación de las dos matrices de la entrada. Le siguen N líneas, cada una conteniendo M valores enteros. Cada una de esas líneas es una fila de la matriz.

4. Entrega de la Tarea

La entrega de la tarea debe realizarse enviando un archivo comprimido llamado

`tarea3-apellido1-apellido2-apellido3.tar.gz`

(reemplazando sus apellidos según corresponda), o alternativamente usando formato zip, en el sitio Aula USM del curso, a más tardar el día 24 de junio, 2020, a las 23:59:00 hrs (Chile Continental), el cual contenga:

- Los archivos con el código fuente necesarios para el funcionamiento de la tarea.
- `NOMBRES.txt`, Nombre y ROL de cada integrante del grupo.
- `README.txt`, Instrucciones de compilación en caso de ser necesarias.
- `Makefile`, Instrucciones para compilación automática.