Proyecto final

"*N* cuerpos: colisión de galaxias"

Materia: Programación Básica

Alumnos:

-Luis Alberto Pérez Martínez

-Emmanuel Judá Rodríguez Nachez

-José Pablo Cuevas Cázares

Fecha de entrega: 30/10/18

Docente: Alma González

Descripción del programa

Se dividió el programa en funciones, las cuales controlaban un aspecto distinto de la interacción entre cuerpos. En la función main se realizó la lectura de datos, los cuales fueron generados usando la función random de python. Se generaron un total de 408 partículas, las cuales fueron divididas a la mitad en dos esferas en el espacio. Usando la función random de python, se generaron posiciones aleatorias para cada partícula, estableciendo una condición para que dichas posiciones aleatorias no salieran de un espacio esférico de un tamaño previamente determinado, generando con esto dos esferas, cada una con 204 partículas respectivamente, distribuidas dentro de ella de manera aleatoria. Para el programa, se decidió que la fuerza que afectaría a las partículas sería la gravitacional, por tanto se utilizó la ley de gravitación universal de Newton como ecuación principal para regir las interacciones entre los cuerpos. Como dicha ley es de carácter vectorial, se generaron las posiciones aleatorias en cada eje coordenado, teniendo tres coordenadas espaciales para cada partícula, permitiéndonos calcular la fuerza en cada eje coordenado. También se generaron velocidades iniciales de cada partícula, y se decidió dejar que, en una de las esferas, todas las partículas tuvieran velocidad inicial de 0, y la otra con una velocidad inicial determinada. Una vez que se generaron las posiciones y velocidades iniciales, se guardaron en dos archivos de texto respectivamente, para así poder leerlos y utilizarlos en el código main de c. Para poder tener un mejor orden en el proyecto, se creó una librería especial, la cual contenía todas las funciones que se usaron en el programa. En el código principal, se declararon primero las librerías necesarias típicas que se usaron durante el curso, con la adición de la nueva librería previamente creada. A continuación se declararon las variables enteras del programa, así como los archivos de lectura de posiciones y velocidades iniciales, y también los archivos de posiciones y velocidades en cada instante. La variable n es el número de partículas, j es la variable que controla la j-ésima partícula del sistema, i es la variable usada en los ciclos for para asignarles valores a los arreglos, t es el tiempo total de evolución del sistema y c ,ch y cp son las variables que se utilizaron para imprimir los archivos de texto, como en el proyecto de segundo parcial. A

continuación se declararon las variables de masa, el parámetro de iteración h y las variables que se usaron para calcular el vector de distancia entre dos partículas (d, dx, dy y dz). Después se procedió a declarar todos los arreglos necesarios. Se declararon arreglos para la posición en los tres ejes en el instante pasado y en el instante actual, esto con el fin de guardar en arreglos las posiciones después de cada iteración, para así saber las nuevas posiciones después de que las partículas sufrieron los efectos de la fuerza entre ellas. Se realizó lo mismo y se crearon arreglos para las velocidades en el instante pasado y en el instante actual (o sea, en la iteración i-ésima del ciclo for en el cual se encontraban, y a su vez en la iteración j-ésima para saber las posiciones de la partícula después de que haya interactuado con las demás. También se crearon dos arreglos de fuerza, igualmente uno para las iteraciones pasadas y otro para las iteraciones en el instante actual. Sin embargo, a diferencia de los demás arreglos, para calcular las nuevas posiciones y velocidades se requiere la fuerza total que experimenta la partícula en cada eje, por lo que se declararon otras variables de fuerza total, esta vez no arreglos, puesto que la fuerza total no es necesario que se guarde en un arreglo, ya que con cada ciclo ésta conforme cambian las posiciones y velocidades, que a su vez afectan la fuerza en cada instante. Dicha suma de fuerzas inicialmente se igualó a cero, puesto que se asumió que al inicio no había interacción entre partículas. Además, se declaró una variable tipo char, la cual cumplía la función de dar nombres distintos a cada archivo nuevo de velocidades y posiciones en la impresión de los archivos de texto en cada iteración. Enseguida se procedió a abrir el archivo de posiciones y velocidades iniciales, y se leyeron los datos de los mismos usando un ciclo for, guardando en tres arreglos distintos las velocidades y posiciones en cada eje coordenado. A continuación se procedió a abrir un archivo que contenía los valores iniciales de la masa, el tiempo total de evolución y el parámetro de iteración h, los cuales fueron también escaneados a partir de un archivo. Una vez que los arreglos iniciales contuvieron a los valores de los archivos de posiciones y velocidades iniciales, se le dio el valor a la variable ch como el resultado de la división entre el tiempo total y h, con el fin de saber el número de iteraciones que el usuario requería. Por consiguiente, se creó un ciclo while para el tiempo, el cual realiza las operaciones necesarias solamente cuando una variable contador “c”, es menor al número de operaciones, es decir, menor a ch. Dentro del ciclo while, se estableció una condición para que se imprimiera la variable c, esto con el fin de saber en qué iteración estaba el ciclo, para así tener una idea clara de el tiempo que le toma al programa en ser corrido correctamente. A continuación, se inicializa el ciclo for con la variable j, la cual controla la partícula j-ésima del sistema, es decir, cada ciclo calcula la fuerza que siente esa partícula por efecto de todas las demás. Una vez dentro del ciclo for con la variable j, se abre otro ciclo for dentro de éste, en el cual se llama a las funciones distancia y distanciat. En la función distancia, se calcula la distancia entre dos partículas en cada eje coordenado, teniendo como argumentos los arreglos de las posiciones en cada eje. La función distancia resta las coordenadas en un mismo eje de las posiciones guardadas en los arreglos de posiciones, y regresa dicho valor. En el mismo archivo, la función distanciat calcula el valor absoluto del vector de posición, usando el resultado de la resta de las distancias en cada eje. La función se llama tres veces en el ciclo for, porque se calcula la distancia entre las coordenadas en cada eje. Una vez calculadas las distancias, se llama la función de distanciat, para así calcular la magnitud del vector compuesto por las ahora calculadas distancias en cada eje entre dos partículas. Por consiguiente, en el mismo ciclo for, se manda a llamar a la función fuerza. Dicha función esta función toma como argumentos a la masa, la distancia previamente calculada entre dos puntos en cada eje (para

así calcular la fuerza en cada eje coordenado) y el valor de la distancia total. Se estableció una condición para evitar valores nulos con un if, indicando que si el valor de la distancia r es cero, entonces la fuerza también es cero, de lo contrario, se calcula la fuerza en cada eje, la cual toma el valor de G (que en este caso, se decidió mantener las unidades de la constante en años, unidades astronómicas y masas solares, con el fin de mantener el valor de G como 4 veces por pi al cuadrado) y después lo multiplicada por la distancia en un eje en específico y lo dividía por el valor absoluto del vector distancia al cubo. Al igual que con las distancias, esta función se llama tres veces, para cada eje coordenado. A continuación, se termina el ciclo for, y enseguida se abre uno nuevo, en el cual se realiza un sumatorio de las fuerzas en todos los ejes, puesto que para resolver las ecuaciones diferenciales con el método de Leap-frog, la fuerza que aparece en la ecuación es la total en cada eje, por lo cual deben sumarse todas las fuerzas. Una vez calculadas las fuerzas, se cierra el ciclo. Enseguida se llama a la función de posición, al mismo tiempo que se iguala con el arreglo de posición con la variable j, porque se están calculando las nuevas posiciones de la j-ésima partícula. La función posición toma como argumentos el parámetro de iteración h, el arreglo de posición de cada partícula en el instante pasado, la velocidad en el instante pasado y la fuerza en el instante pasado. Dicha función cumple con la siguiente ecuación:

$$x_i(t+h) = x_i(t) + h * v_{x,i}(t) + 1/2 * F_x * h^2 \quad (1)$$

Inmediatamente después de realizar la función de posiciones, se realiza el cálculo de las velocidades, la cual requiere los argumentos de la velocidad en el instante pasado, la fuerza en el instante pasado, la fuerza total (la sumatoria de todas las fuerzas en un solo eje) y el parámetro de iteración h. Dicho arreglo de velocidad cumple con la siguiente ecuación:

$$v_{x,i}(t+h) = v_{x,i}(t) + 1/2 * (F_x(t) + F_x(t+h)) * h \quad (2)$$

Se llama a la función de posiciones tres veces, y también se realiza tres veces el cálculo de las velocidades, porque todo lo anterior se realiza en los tres ejes coordenados.
Cabe señalar que en todo el cuerpo del código, tanto principal como de funciones, todos los arreglos, y en general todas las variables excepto algunas cuantas, se declararon como long double, puesto que los valores generados en python eran muy grandes, lo cual es de esperarse, puesto que se están manejando distancias, masas, fuerzas y tiempos de iteración enormes. Una vez calculadas las nuevas posiciones y velocidades con base en las fuerzas generadas entre partículas, y partiendo del archivo, se procedió a igualar las nuevas posiciones con las posiciones viejas, e igual para las velocidades y fuerzas. Lo anterior se realizó para que cuando se cambiara de partícula (es decir, que el ciclo for de j cambiara), las velocidades, posiciones y fuerzas no fueran las tomadas del archivo, si no las ya previamente calculadas después de que pasó un determinado tiempo en el sistema.
Enseguida se igualaron las fuerzas totales a cero, ya que para que el programa funcionara correctamente, y fuera lo más realista posible, las fuerzas totales se igualan a cero, puesto que como se modifican las velocidades, esto provoca un cambio en las posiciones, y por ende en las distancias, y en última instancia, en las fuerzas que existen en todo el sistema en cada eje. Entonces, si se modifican las fuerzas en cada eje en cada iteración, no sirve de nada guardar la fuerza total de antes, por lo que se necesita igualar a cero la fuerza al final del arreglo de j, para que al pasar a la siguiente partícula la fuerza total no sea la que

experimentó la partícula anterior, si no una nueva. Finalmente, después de realizar todos los cálculos, se realizó un ciclo para pode imprimir el número de archivos que el usuario escogiera. Esto se realizó para imprimir los archivos de posiciones y de velocidades, puesto que lo que es importante para la realización de las gráficas, y poder así observar la evolución del sistema, son las posiciones a cada instante. De manera análoga al proyecto de segundo parcial, se decidió imprimir solamente un número determinado de archivos de texto con las nuevas posiciones, al igual que para las nuevas velocidades. Como el tiempo de evolución del sistema fue de diez mil años, y al parámetro de iteración h se le dio un valor de 0.1, se realizaron un total de cien mil cálculos de posiciones, de los cuales se decidió solamente imprimir un total de 500 archivos de texto, tanto de posiciones como de velocidades. Finalmente, se construyó un programa similar al utilizado en el proyecto de segundo parcial para graficas, modificando el código para que pudiese graficar en tres dimensiones, tomando los datos de los archivos de texto en cada eje, y seleccionando cada fila como las coordenadas de la partícula j-ésima, generando así un total de 500 gráficas, correspondientes a los 500 archivos de texto que se decidió fueran generados para observar la evolución del sistema.

Discusión y análisis de resultados

De los 500 archivos de texto obtenidos, se graficó cada uno de ellos, con el fin de observar de manera paulatina la evolución del sistema a través del tiempo. Como ejemplo, en la iteración número 2000, se obtuvo la siguiente imagen:

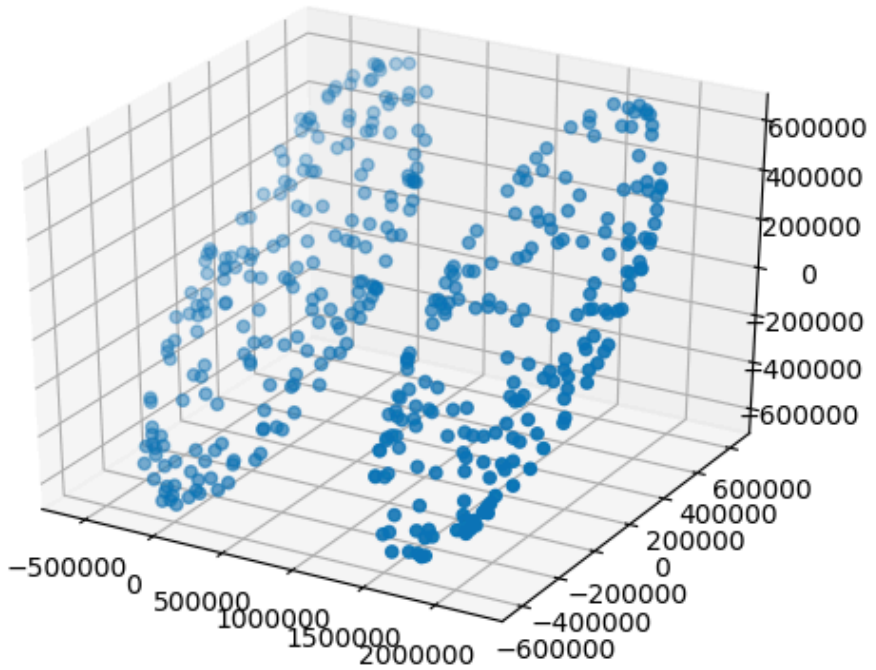

Figura1. Imagen obtenida al graficar los datos obtenidos del décimo archivo de texto, correspondiente a la iteración número 2000. Se observa que, como no ha pasado aún mucho tiempo en el sistema, las posiciones son semejantes a las posiciones iniciales, por lo que no se observa un cambio muy drástico en cuanto a la distribución de partículas.

Al dejar evolucionar al sistema durante más tiempo, se obtuvo la siguiente imagen, correspondiente a la iteración número 55000:

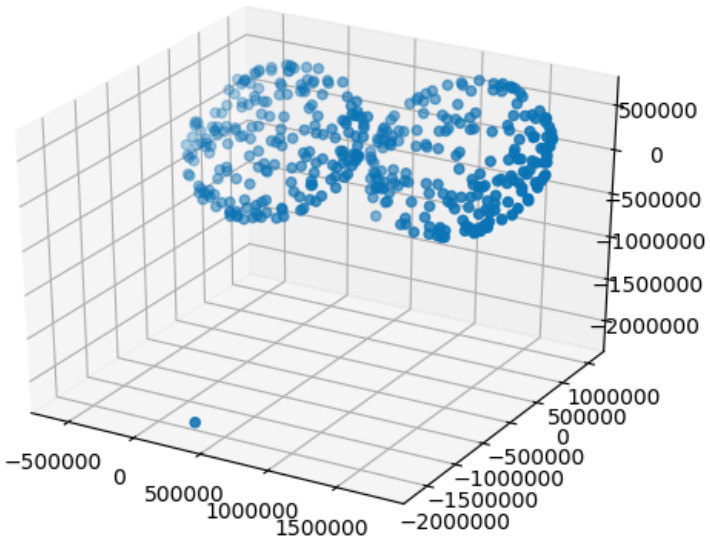

Figura2. Imagen correspondiente al archivo de texto número 275, y a la iteración número 55000.

En la imagen se puede observar cómo, después de que el tiempo avanza, el sistema tiende a colapsar, producto de la fuerza gravitacional existente entre las partículas. También se puede inferir que, si pasa suficiente tiempo, el sistema tenderá a colapsar en un espacio cada vez más pequeño, incrementando así la magnitud de las fuerzas que existen entre las partículas. Finalmente, en la iteración número 100000, correspondiente al archivo de texto número 500, se obtuvo la siguiente imagen:

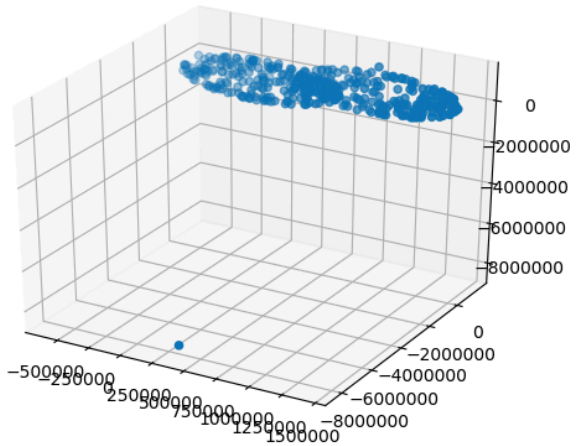

Figura3. Imagen correspondiente al archivo de texto número 500, y a la iteración número 10000. La imagen representa el estado final del sistema.

Como se puede observar en la figura anterior, el sistema tiende al colapso, puesto que como se infirió anteriormente, el carácter atractivo de la fuerza gravitacional tiende a crecer conforme se reducen las distancias entre las partículas, como se puede observar en la ley de gravitación universal, expresada de la siguiente manera:

$$m_p \ddot{\vec{r}}_i = -\sum_{i \neq j}^{N} \frac{G m_p m_p \vec{r_{ij}}}{r_{ij}^3}. \qquad (3)$$

Al analizar la ecuación anterior, observamos que la fuerza que experimenta la partícula j-ésima es la suma de todas las fuerzas del sistema, exceptuando la fuerza que produce ella misma, razón por la cual se estableció esa condición en el código.