PRIMER PROYECTO

SISTEMAS OPERATIVOS

ESTUDIANTES:

Juan Clavijo

Santiago Mesa

Juliana Lugo

DOCENTE:

John Jairo Corredor

Facultad de Ingeniería

FECHA DE ENTREGA:

01 de octubre de 2023

Pdispersa

El código fuente pdispersa.c se encarga de verificar si una matriz es dispersa o no, y para lograrlo, usa los procesos y la función fork() para dividir eficientemente una matriz en partes manejables y realizar un análisis concurrente de elementos diferentes de cero en esas partes. Luego, toma una decisión sobre si la matriz es dispersa o no en función de los resultados recopilados de los procesos hijos. Esto permite analizar matrices grandes de manera eficiente y determinar si cumplen con ciertos criterios para considerarse dispersas.

- **¿Cómo realizó la división de la matriz?**

Para dividir la matriz y realizar un análisis eficiente, se emplea una estrategia de división horizontal o vertical, dependiendo de si el número de filas o columnas es mayor. La elección entre división horizontal o vertical garantiza un proceso paralelo eficiente, ya que se adapta a la geometría de la matriz. En caso de que las dimensiones de la matriz sean muy similares, se utiliza la división horizontal.

División Horizontal: En este enfoque, se divide el trabajo entre múltiples procesos hijos, donde cada uno se encarga de contar los elementos diferentes de cero en un rango específico de filas de la matriz. Estos procesos se ejecutan de manera concurrente, lo que acelera el procesamiento.

División Vertical: En el caso de una división vertical, los procesos hijos se encargan de contar los elementos diferentes de cero en un rango específico de columnas de la matriz. Al igual que en la división horizontal, este enfoque aprovecha la concurrencia para acelerar el conteo.

- **¿Qué información se pasa del padre a los hijos y de los hijos al padre?**

Para llevar a cabo la tarea de contar los elementos diferentes de cero, se pasa información importante entre el proceso padre y los procesos hijos:

Del Padre a los Hijos: El proceso padre divide la matriz en subconjuntos manejables y asigna tareas específicas a cada proceso hijo. Esto incluye información sobre el rango de filas o columnas que debe procesar cada hijo.

De los Hijos al Padre: Cada proceso hijo realiza el conteo de elementos diferentes de cero en su conjunto de filas o columnas asignado. Al completar su tarea, el proceso hijo devuelve el resultado al proceso padre, que acumula estos resultados para tomar una decisión final sobre si la matriz es dispersa o no.

- **¿Cómo se pasan la información: por medio de estructuras, archivos, etc...?**

La información se pasa entre el proceso padre y los procesos hijos utilizando mecanismos de comunicación específicos de la programación en paralelo. En este caso, se utilizan llamadas al sistema y funciones de la biblioteca estándar de C para manejar la comunicación entre procesos:

- Fork(): La función fork() se utiliza para crear procesos hijos. Cada proceso hijo hereda una copia de la memoria del proceso padre y ejecuta una porción específica de código.
- Wait(): La función wait() se utiliza para que el proceso padre espere a que los procesos hijos terminen su ejecución y recopilen sus resultados. Esto asegura que el proceso padre obtenga los resultados de todos los hijos antes de tomar una decisión.
- Exit(): Los procesos hijos utilizan la función exit() para terminar su ejecución y devolver el resultado al proceso padre. Este resultado se recopila mediante la función WEXITSTATUS() en el proceso padre.

Hdispersa

hdispersa.c utiliza hilos y la biblioteca pthread para llevar a cabo un procesamiento en paralelo de una matriz y determinar si es dispersa en función de un umbral específico. Este enfoque paralelo permite acelerar el procesamiento de matrices grandes y mejora la eficiencia del programa en comparación con un enfoque secuencial.

- **¿Cómo realizó la división de la matriz?**

La división de la matriz en el programa se lleva a cabo mediante el uso de múltiples hilos, con la posibilidad de dividirla tanto horizontal como verticalmente, dependiendo de si el número de filas o columnas es divisible por el número de hilos especificado por el usuario.

El programa verifica primero si es posible dividir exactamente las filas o columnas entre los hilos. Si es posible, asigna una cantidad igual de filas o columnas a cada hilo. En caso contrario, decide basándose en la dimensión más grande de la matriz y distribuye las filas o columnas en consecuencia.

- **¿Qué información se pasa del padre a los hijos y de los hijos al padre?**

El programa utiliza hilos para procesar la matriz de manera paralela. Para lograrlo, se pasa información relevante a través de la estructura ThreadData, que contiene los siguientes datos:

- inicio: El índice de inicio de las filas o columnas asignadas al hilo.
- fin: El índice de finalización de las filas o columnas asignadas al hilo.
- numcols o numfilas: El número de columnas o filas en la matriz, dependiendo de si se trata de una división horizontal o vertical.
- matriz: Un puntero a la matriz que se procesará en el hilo.

Desde el padre a los hijos, se pasa esta información para que cada hilo sepa cuál es su tarea específica, es decir, qué parte de la matriz debe procesar.

Desde los hijos al padre, se pasa la información sobre la cantidad de elementos diferentes de cero encontrados en las filas o columnas asignadas a cada hilo. Esta información se almacena en variables locales de cada hilo y se recopila en el proceso padre después de que todos los hilos hayan completado su trabajo.

- **¿Cómo se pasan la información: por medio de estructuras, archivos, etc...?**

La información se pasa principalmente a través de estructuras. La estructura ThreadData se utiliza para agrupar los datos necesarios que se pasan de un hilo a otro y del hilo al proceso padre. Cada hilo recibe una copia de esta estructura con los datos específicos de la tarea que debe realizar.

La comunicación entre los hilos y el proceso padre se realiza utilizando las funciones pthread_create para crear los hilos y pthread_join para esperar a que los hilos terminen y obtener sus resultados. La información sobre la cantidad de elementos diferentes de cero encontrados en cada hilo se almacena en variables locales de los hilos y se agrega al contador total en el proceso padre después de la finalización de todos los hilos.

La información se pasa de manera eficiente utilizando estructuras y las funciones proporcionadas por la biblioteca de hilos de POSIX para la creación, gestión y sincronización de hilos.

Plan de Pruebas

Se realizó un plan de pruebas con diversos casos para evaluar el funcionamiento del código tanto para pdispersa.c como para hdispersa.c. Consulte el anexo.

ANEXO

Plan de pruebas: pdispersa.c			
Descripción de caso	**Valores de entrada**	**Resultado esperado**	**Resultado obtenido**
1. Menos argumentos de los requeridos	Argumentos menores a 10, no se le pasa el argumento de porcentaje.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver Figura 1.
2. Funcionamiento con parámetros en un orden	10 argumentos en el orden: filas, columnas, archivo, número de procesos y porcentaje.	Después de pasarle los argumentos en el orden correspondiente, en la última línea nos debe imprimir que la matriz es dispersa.	Ver Figura 2.
3. Funcionamiento con parámetros en orden diferente	10 argumentos en el orden: número de procesos, porcentajes, filas, columnas	Después de pasarle los argumentos en el orden correspondiente, en la última línea nos debe imprimir que la matriz es dispersa.	Ver Figura 3.
4. Fallo de usuario al ingresar número de procesos	Pasar un número impar para el número de procesos.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver figura 4.
5. Fallo de usuario al ingresar número de filas	Pasar mal el número de filas.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver figura 5.
6. Fallo de usuario al ingresar número de procesos mayor al número de núcleos	Pasar mal el número de procesos en este caso 16. **Nota:** Estamos en una maquina virtual que tiene 4 núcleos de procesador.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver figura 6 y 7.
7. Matriz 50x50 al 75%.	Se pasa un archivo de 50x50 que	Al final de la ejecución el	Ver figura 8.

	contiene la matriz. Se usan 4 procesos y el porcentaje de la matriz para que sea dispersa es del 75%.	programa nos tiene que decir que la matriz es dispersa.	
8. Matriz 10x20 al 90%.	Se pasa un archivo con una matriz de 10x20 y se utilizaran 4 procesos. El porcentaje para que la matriz sea considerada dispersa es del 90%.	Al final de la ejecución el programa nos tiene que decir que la matriz es dispersa.	Ver figura 9.
9. Matriz 3x3 que no sea sparse.	Se pasa un archivo que tiene una matriz 3x3 que sabemos que no es dispersa porque solamente uno de esos elementos es 0.	Al final de la ejecución el programa nos tiene que decir que la matriz no es dispersa.	Ver figura 10.

Evidencias de los resultados obtenidos para pdispersa.c

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 3 -c 3 -a mat3x3pd.txt -n 2
no estoy seguro que lo que hayas ingresado sea correcto, REVISA
```

Figura 1. Caso 1 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 3 -c 3 -a mat3x3pd.txt -n 2 -p 50
Número de filas: 3

Número de columnas: 3

se ha reservado la memoria para la matriz con exito!

Los contenidos del archivo se cargaron exitosamente en memoria

El archivo se abrió y cerró exitosamente

La matriz en memoria se ve asi:

1 0 0
0 0 1
1 1 0

el proceso hijo con ID: 3378 encontro 3 elementos distintos de cero entre las filas 1 y 3

el proceso hijo con ID: 3377 encontro 1 elementos distintos de cero entre las filas 0 y 1

el total que escucha el proceso padre es 4

el numero de ceros debe ser 5

el numero de elementos diferentes de cero debe ser 4

La matriz es dispersa.
```

Figura 2. Caso 2 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -n 2 -p 50 -a mat3x3pd.txt -f 3 -c 3
Número de filas: 3

Número de columnas: 3

se ha reservado la memoria para la matriz con exito!

Los contenidos del archivo se cargaron exitosamente en memoria

El archivo se abrió y cerró exitosamente

La matriz en memoria se ve asi:

1 0 0
0 0 1
1 1 0

el proceso hijo con ID: 3386 encontro 1 elementos distintos de cero entre las filas 0 y 1

el proceso hijo con ID: 3387 encontro 3 elementos distintos de cero entre las filas 1 y 3

el total que escucha el proceso padre es 4

el numero de ceros debe ser 5

el numero de elementos diferentes de cero debe ser 4

La matriz es dispersa.
```

Figura 3. Caso 3 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 3 -c 3 -a mat3x3pd.txt -n 3 -p 50
recuerda que necesito que el numero de procesos sea PAR revisa!
```

Figura 4. Caso 4 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 2 -c 3 -a mat3x3pd.txt -n 2 -p 50
Número de filas: 3

Número de columnas: 3

estas seguro de que pusiste el numero de columnas y filas correcto?
```

Figura 5. Caso 5 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 3 -c 3 -a mat3x3pd.txt -n 16 -p 50
Número de filas: 3

Número de columnas: 3

se ha reservado la memoria para la matriz con exito!

Los contenidos del archivo se cargaron exitosamente en memoria

El archivo se abrió y cerró exitosamente

La matriz en memoria se ve asi:

1 0 0
0 0 1
1 1 0

no es posible ejecutar el programa ya que se piden 16 procesos y el computador tiene 4 nucleos, me pides mas procesos que nucleos
```

Figura 6. Caso 6 Plan de pruebas pdispersa.c.

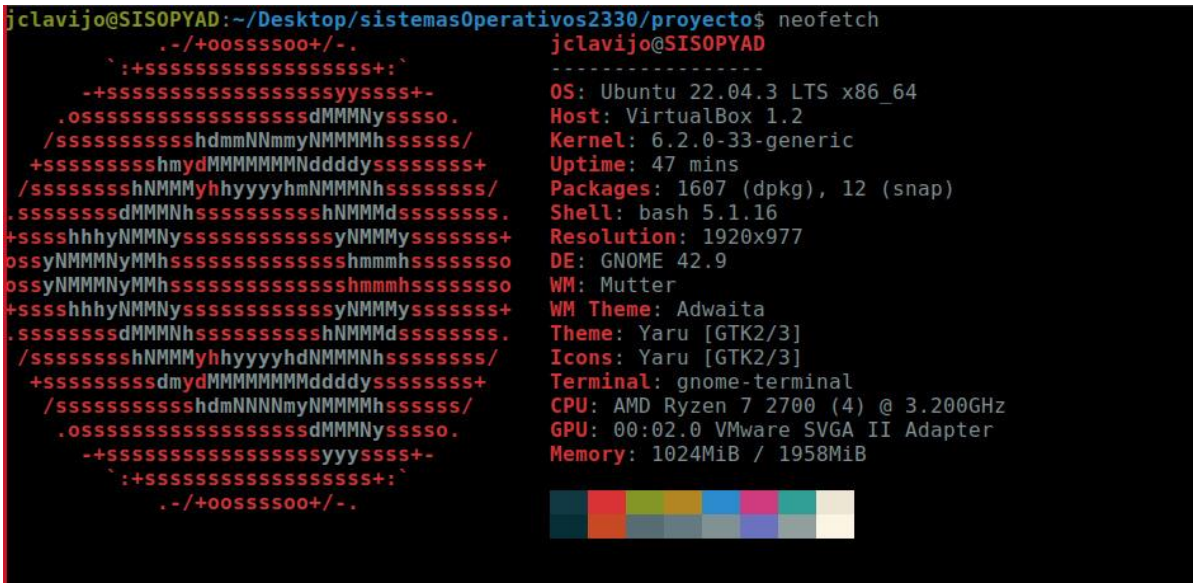

Figura 7. Caso 6 Plan de pruebas pdispersa.c.

Figura 8. Caso 7 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 10 -c 20 -a mat10x20pd.txt -n 4 -p 90
Número de filas: 10

Número de columnas: 20

se ha reservado la memoria para la matriz con exito!

Los contenidos del archivo se cargaron exitosamente en memoria

El archivo se abrió y cerró exitosamente

La matriz en memoria se ve asi:

0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0
1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0

el proceso hijo con ID: 3503 encontro 6 elementos distintos de cero entre las columnas 0 y 5

el proceso hijo con ID: 3504 encontro 8 elementos distintos de cero entre las columnas 5 y 10

el proceso hijo con ID: 3505 encontro 3 elementos distintos de cero entre las columnas 10 y 15

el proceso hijo con ID: 3506 encontro 3 elementos distintos de cero entre las columnas 15 y 20

el numero de ceros debe ser 180

el numero de elementos diferentes de cero debe ser 20

La matriz es dispersa.
```

Figura 9. Caso 8 Plan de pruebas pdispersa.c.

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./pdispersa -f 3 -c 3 -a matnosparse.txt -n 4 -p 10
Número de filas: 3

Número de columnas: 3

se ha reservado la memoria para la matriz con exito!

Los contenidos del archivo se cargaron exitosamente en memoria

El archivo se abrió y cerró exitosamente

La matriz en memoria se ve asi:

1 1 0
1 1 1
1 1 1

el proceso hijo con ID: 3516 encontro 0 elementos distintos de cero entre las filas 0 y 0

el proceso hijo con ID: 3517 encontro 0 elementos distintos de cero entre las filas 0 y 0

el proceso hijo con ID: 3518 encontro 0 elementos distintos de cero entre las filas 0 y 0

el proceso hijo con ID: 3519 encontro 8 elementos distintos de cero entre las filas 0 y 3

el total que escucha el proceso padre es 8

el numero de ceros debe ser 1

el numero de elementos diferentes de cero debe ser 8

La matriz no es dispersa.
```

Figura 10. Caso 9 Plan de pruebas pdispersa.c.

Plan de pruebas: hdispersa.c			
Descripción de caso	**Valores de entrada**	**Resultado esperado**	**Resultado obtenido**
1. Menos argumentos de los requeridos	Argumentos menores a 10, no se le pasa el argumento de porcentaje.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver figura 11.
2. Funcionamiento con parámetros en un orden	10 argumentos en el orden: filas, columnas, archivo, número de procesos y porcentaje.	Nos dice que la matriz es	Ver figura 12.
3. Funcionamiento con parámetros en un orden diferente	10 argumentos en el orden: número de procesos, porcentajes, filas, columnas	Después de pasarle los argumentos en el orden correspondiente, en la última línea nos debe imprimir que la matriz es dispersa.	Ver figura 13.
4. Fallo de usuario al ingresar número de filas	Pasar mal el número de filas.	Imprime línea que indica que hay algo mal y no deja ejecutar el programa.	Ver figura 14.
5. Ingresar número de procesos mayor al número de núcleos	Pasar el número de procesos en este caso 16. **Nota:** Estamos en una máquina virtual que tiene 4 núcleos de procesador.	Se ejecuta el programa y muestra que la matriz es sparse	Ver figura 15.
6. Matriz 50x50 al 75%.	Se pasa un archivo de 50x50 que contiene la matriz. Se usan 4 procesos y el porcentaje de la matriz para que sea dispersa es del 75%.	Al final de la ejecución el programa nos tiene que decir que la matriz es dispersa.	Ver figura 16.
7. Matriz 10x20 al 90%	Se pasa un archivo con una matriz de 10x20 y se	Al final de la ejecución el programa nos tiene	Ver figura 17.

	utilizaran 4 procesos. El porcentaje para que la matriz sea considerada dispersa es del 90%.	que decir que la matriz es dispersa.	
8. Matriz 3x3 que no sea sparse	Se pasa un archivo que tiene una matriz 3x3 que sabemos que no es dispersa porque solamente uno de esos elementos es 0.	Al final de la ejecución el programa nos tiene que decir que la matriz no es dispersa.	Ver figura 18.

Evidencias de los resultados obtenidos para hdispersa.c

Figura 11. Caso 1 Plan de pruebas hdispersa.c

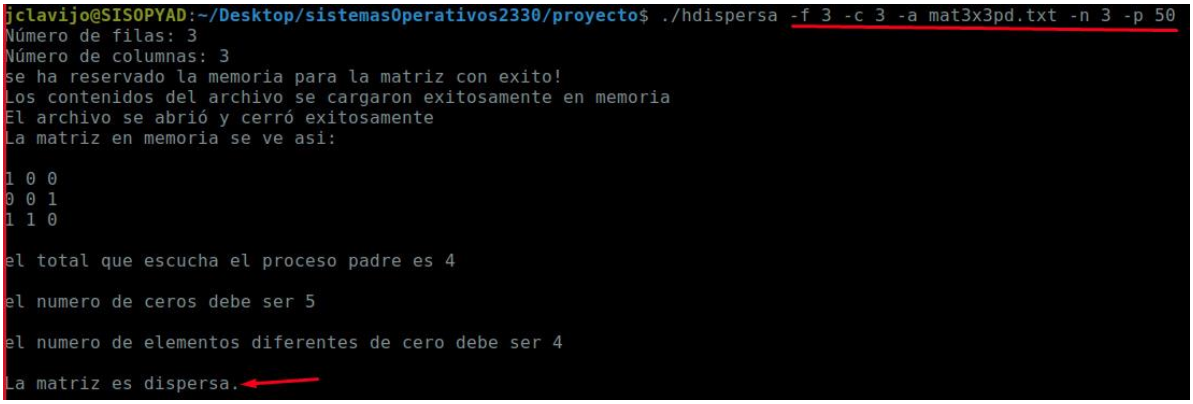

Figura 12. Caso 2 Plan de pruebas hdispersa.c

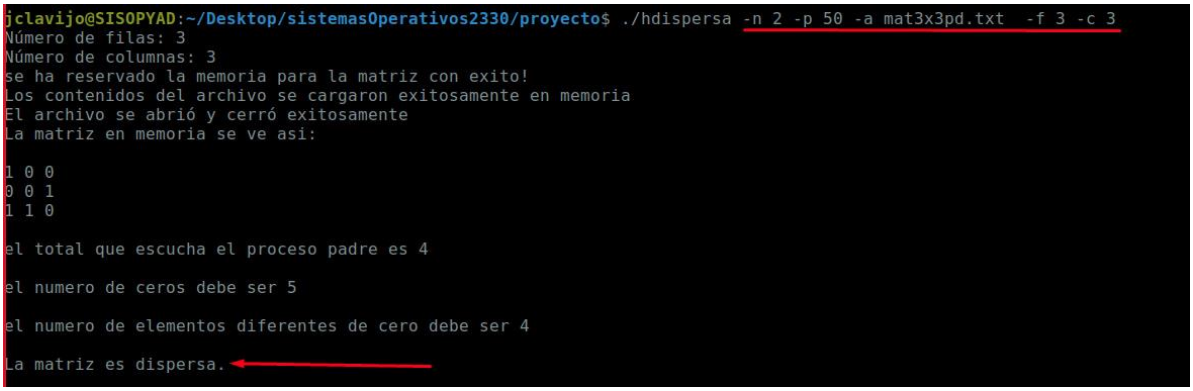

Figura 13. Caso 3 Plan de pruebas hdispersa.c

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./hdispersa -n 2 -p 50 -a mat3x3pd.txt  -f 2 -c 3
Número de filas: 3
Número de columnas: 3
estas seguro de que pusiste el numero de columnas y filas correcto?
```

Figura 14. Caso 4 Plan de pruebas hdispersa.c

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./hdispersa -f 3 -c 3 -a mat3x3pd.txt -n 16 -p 50
Número de filas: 3
Número de columnas: 3
se ha reservado la memoria para la matriz con exito!
Los contenidos del archivo se cargaron exitosamente en memoria
El archivo se abrió y cerró exitosamente
La matriz en memoria se ve asi:

1 0 0
0 0 1
1 1 0

el total que escucha el proceso padre es 4

el numero de ceros debe ser 5

el numero de elementos diferentes de cero debe ser 4

La matriz es dispersa.
```

Figura 15. Caso 5 Plan de pruebas hdispersa.c

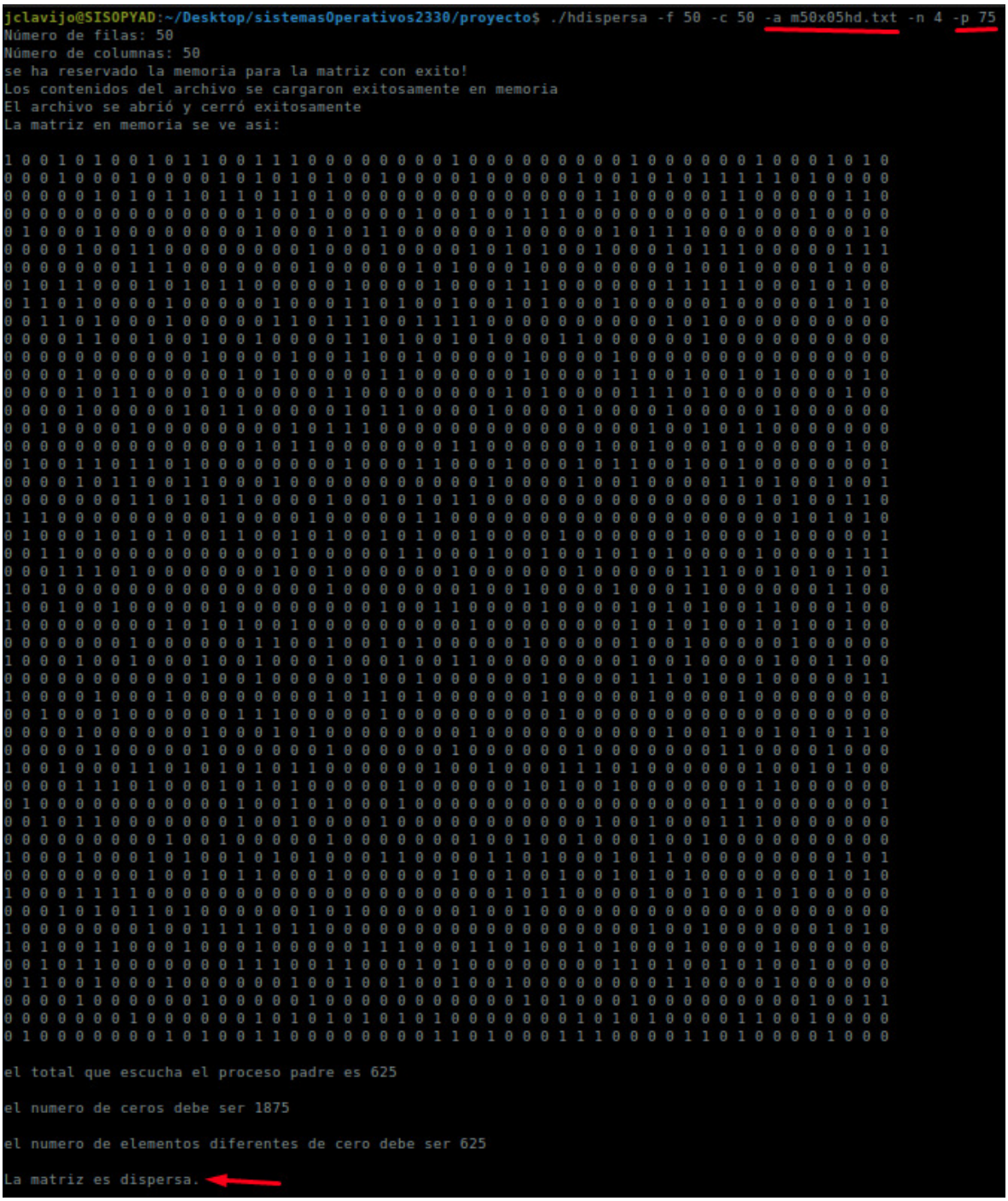

Figura 16. Caso 6 Plan de pruebas hdispersa.c

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./hdispersa -f 10 -c 20 -a m10x20hd.txt -n 5 -p 90
Número de filas: 10
Número de columnas: 20
se ha reservado la memoria para la matriz con exito!
Los contenidos del archivo se cargaron exitosamente en memoria
El archivo se abrió y cerró exitosamente
La matriz en memoria se ve asi:

0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0
0 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

el total que escucha el proceso padre es 20

el numero de ceros debe ser 180

el numero de elementos diferentes de cero debe ser 20

La matriz es dispersa.
```

Figura 17. Caso 7 Plan de pruebas hdispersa.c

```
jclavijo@SISOPYAD:~/Desktop/sistemasOperativos2330/proyecto$ ./hdispersa -f 3 -c 3 -a matnosparse.txt -n 2 -p 10
Número de filas: 3
Número de columnas: 3
se ha reservado la memoria para la matriz con exito!
Los contenidos del archivo se cargaron exitosamente en memoria
El archivo se abrió y cerró exitosamente
La matriz en memoria se ve asi:

1 1 0
1 1 1
1 1 1

el total que escucha el proceso padre es 8

el numero de ceros debe ser 1

el numero de elementos diferentes de cero debe ser 8

La matriz no es dispersa.
```

Figura 18. Caso 8 Plan de pruebas hdispersa.c