
UBA FACULTAD DE INGENIERÍA

66.20 Organización de Computadoras

Trabajo Práctico 2

2^{do} Cuatrimestre 2020

Integrantes:

Bacigalupo, Ivan	98064
ibacigaluppo@fi.uba.ar	
Carballo, Matías	93762
mcarballo@fi.uba.ar	
Marshall, Juan Patricio	95471
jmarshall@fi.uba.ar	

Índice

1. Introducción	2
2. Compilación	2
3. Desarrollo y Análisis	2
3.1. Hipótesis	2
3.2. Simulacion	3
3.3. Analisis de la simulacion	3
4. Conclusión	5
5. Código del script	5
5.1. correr-simulacion.sh	5
6. Salida de las corridas	5
6.1. Direct mapping:	5
6.2. 2WSA:	100
6.3. 4WSA:	194
7. Enunciado	289

1. Introducción

El trabajo práctico consistió en un análisis teórico con una posterior simulación de un benchmark para comparar distintos tipos de caches. Se realizaron en total 24 corridas del programa, variando entre los 3 tipos de cache presentadas (DM, 2WSA, 4WSA) y cambiando las dimensiones de la matriz cuadrada, input del programa, entre 1 y 8. Para las corridas, se utilizó el mismo ambiente MIPS que en el TP1, pero instalando manualmente una versión arreglada de valgrind. Que nos permitió correr cachegrind y cg annotate, para así poder sacar datos reales de las simulaciones. Para generar los diferentes inputs, hicimos un mini script de python para generar los txts correspondientes. Al igual que generamos un bash script que ejecuta todas las simulaciones necesarias, escribiendo a cada uno de los archivos correspondientes.

2. Compilación

Para poder correr todas las simulaciones, basta con primero correr el Makefile provisto por la cátedra para generar el ejecutable mmult, y luego ejecutar el script de correr-simulacion.sh `./make tp1 ./correr-simulacion.sh`

3. Desarrollo y Análisis

3.1. Hipótesis

Primeramente identifiquemos el experimento. El programa en C, se encarga de multiplicar matrices dado un input, que tiene las dimensiones de la matriz y todos sus elementos. Suponemos de por sí, que este programa va a variar su comportamiento a nivel performance, tanto en tiempo como en memoria, tiempos de accesos, performances de caches, a medida que cambiemos las dimensiones de la matriz ingresada como input.

A su vez, se quiere estudiar el comportamiento del programa, como benchmark, frente a 3 configuraciones distintas de cache.

Direct Mapping es la cache donde las lecturas son más rápidas, al mapearse direcciones de forma directa 1 a 1 y no se necesitan políticas de borrado (LRU, por ejemplo). Pero esto obviamente tiene un contraefecto muy fuerte: si en la ejecución de un programa, se producen conflictos de manera permanente, el miss rate es más grande de lo que uno supondría. Para poner un ejemplo, un programa que simplemente carga 2 valores en memoria, pero ambos colisionan a la misma dirección en la cache, la carga de cualquiera de esos 2 valores, va a resultar en el pisado del otro. Por lo que, a pesar de tener todos los demás espacios de cache libre, lo que haría posible guardar ambos datos en cache, nos quedamos siempre con 1 solo de los 2.

Esto resulta que en casos generales, pocos casos de uso real aplican caches de este estilo.

En contraparte, las caches n-asociativas, como lo son las de 2 WSA y 4WSA, se basan en que para un mismo valor de "hash" de la cache, tienen n posiciones dedicadas para mantener datos. Casos donde se producen colisiones permanentes, como el descrito en el analisis del Direct Mapping, desencadenaria en que se cachearan ambos datos. En otras palabras, en las caches n-asociativas, se permiten n colisiones/conflictos. Para un mismo valor de hash de la cache, se mantienen n valores. Esto sí obliga a aplicar politicas de borrado para decidir en casos de que ya haya n valores con ese hash cacheados, y estemos queriendo cargar un valor mas. Obviamente, eso tambien tiene un costo, y como bien sabemos, no existen las balas de plata en la programacion. Al tener multiples valores para un mismo "hash", el tiempo necesario para saber si el dato que buscamos esta cacheado o no, aumenta en consideracion a un mapeo directo. Ahora necesitamos iterar por los distintos datos cargados, para saber realmente si nuestro dato estaba cacheado previamente o no.

Partiendo de caches de igual tamaño, si usamos direct mapping, cada posicion de la cache equivale a 1 valor del hash (bits mas significativos de la direccion de memoria). Mientras que en cada nivel de asociatividad nuevo, estamos bajando la cantidad de hashes posibles del cache, ya que ahora para 1 mismo valor del hash hay n posiciones posibles.

3.2. Simulacion

Para la simulacion, se realizaron 24 corridas en total, iterando por el combo del nivel de asociatividad 1(DM), 2(2WSA), 4(4WSA), y las dimensiones nXn de la matriz de input, variando de 1 a 8 monoincrementalmente.

Ejecutando el script bash `./correr-simulacion.sh` post compilacion, se generan todos los outputs necesarios para contrastar nuestra hipotesis a la simulacion.

```
/opt/valgrind/bin/valgrind -log-file=cgrind11x1.txt" --I1 = 32768, 4, 32-  
-D1 = 32768, 1, 32--tool = cachegrind/tmp/02-mmult < /root/CARPETA/tp2-  
2020-2q-src/input11x1.txt > cgrind11x1.txt; /opt/valgrind/bin/cg_annotatecachegrind.out.*  
/root/CARPETA/tp2-2020-2q-src/main.c >> cgrind11x1.txt;
```

3.3. Analisis de la simulacion

Luego de recopilar todos los datos de las corridas, armamos una tabla con los valores que mas nos importan: miss count y miss rate de la cache L1D, para cada combinacion posible.

	DM		2WSA		4WSA	
n	miss rate	miss count	miss rate	miss count	miss rate	miss count
1	1.80%	1,353,997	1.80%	1,315,006	1.80%	1,314,783
2	1.80%	1,354,483	1.80%	1,314,921	1.80%	1,314,795
3	1.80%	1,354,211	1.80%	1,314,930	1.80%	1,314,800
4	1.80%	1,354,045	1.80%	1,314,942	1.80%	1,314,813
5	1.80%	1,354,198	1.80%	1,315,099	1.80%	1,314,830
6	1.80%	1,354,213	1.80%	1,315,141	1.80%	1,314,850
7	1.80%	1,354,741	1.80%	1,315,005	1.80%	1,314,875
8	1.80%	1,354,925	1.80%	1,315,252	1.80%	1,314,908

Como bien podemos ver, hay algo muy raro. Señal de que algo anda mal. El miss rate no varia ni entre tipo de cache, ni entre corridas de diferente input. Por otra parte, el miss count parece crecer de manera constante de la mano de agrandar las dimensiones de la matriz de input.

Pero entonces, ¿que paso? Viendo el output del cg annotate contra nuestro archivo main.c, identificamos que algo raro esta ocurriendo en la linea 180.

```

1      .      .      .      .      .      .      .      .
2      .      .      .      .      #if 1      .      .      .
3      .      .      .      .      if (1) {      .      .      .
4      .      .      .      .      size_t j;      .      .      .
5      2      0      0      0      0      0      0      1
6      0      0      0      0      size_t dim = 1024*1024*10;
7      9      1      1      3      0      0      1
8      0      0      0      int *v = malloc(dim*sizeof(int));
9      83,886,088      2      2      31,457,282      20,480      0      10,485,761
10     0      0      for (j = 0; j < dim; ++j)
11     62,914,560      0      0      20,971,520      0      0      10,485,760
12     1,328,640      1,310,720      v[j] = -1;
13     6      1      1      3      2      2      0
14     0      0      free(v);
15     .      .      .      .      .      .      .
16     .      .      .      .      }
17     .      .      .      .      .      .      .
18     .      .      .      .      #endif

```

```

1  #if 1
2  if (1) {
3      size_t j;
4      size_t dim = 1024*1024*10;
5      int *v = malloc(dim*sizeof(int));
6      for (j = 0; j < dim; ++j)
7          v[j] = -1;
8      free(v);
9  }
10 #endif

```

Podemos ver con claridad en el output del cg annotate, que la mayoría de todos los misses de toda ejecucion, van a ocurrir siempre en este bloque

de código. Sin importar el input, ni del tipo de cache.

Probablemente en futuras reentregas sabremos si tenemos que eliminar esta parte del código, que no parece hacer nada útil para la multiplicación de matrices, y volver a analizar con eso.

4. Conclusión

Como no logramos identificar diferencias entre los resultados de las corridas, no podemos por el momento contrastar nuestras hipótesis iniciales.

En una futura reentrega esperamos poder realizar las correcciones necesarias para poder realizar un análisis más exhaustivo y cumplir con los objetivos del trabajo práctico.

5. Código del script

5.1. correr-simulacion.sh

6. Salida de las corridas

6.1. Direct mapping:

```
1 1 30
2 == Cachegrind, a cache and branch-prediction profiler
3 ==598== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
   Nethercote et al.
4 ==598== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
5 ==598== Command: /tmp/02-mmult
6 ==598== Parent PID: 597
7 ==598==
8 --598-- Warning: Cannot auto-detect cache config, using
   defaults.
9 --598--           Run with -v to see.
10 ==598==
11 ==598== I   refs:           147,098,909
12 ==598== I1  misses:           2,390
13 ==598== L1i misses:          2,372
14 ==598== I1  miss rate:           0.00%
15 ==598== L1i miss rate:          0.00%
16 ==598==
17 ==598== D   refs:           73,518,202 (52,506,753 rd +
   21,011,449 wr)
18 ==598== D1  misses:           1,353,997 ( 24,755 rd +
   1,329,242 wr)
19 ==598== L1d misses:           1,314,138 ( 2,894 rd +
   1,311,244 wr)
```

```

20 ==598== D1 miss rate:          1.8% (      0.0%  +
    6.3% )
21 ==598== LLd miss rate:          1.8% (      0.0%  +
    6.2% )
22 ==598==
23 ==598== LL refs:              1,356,387 (    27,145 rd  +
    1,329,242 wr)
24 ==598== LL misses:            1,316,510 (     5,266 rd  +
    1,311,244 wr)
25 ==598== LL miss rate:          0.6% (      0.0%  +
    6.2% )
26 -----
27 I1 cache:                    32768 B, 32 B, 4-way associative
28 D1 cache:                    32768 B, 32 B, direct-mapped
29 LL cache:                    524288 B, 32 B, 8-way associative
30 Command:                     /tmp/02-mmult
31 Data file:                   cachegrind.out.598
32 Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds: 0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
39
40 -----
41 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
    DLmw
42 -----
43 147,098,909 2,390 2,372 52,506,753 24,755 2,894 21,011,449
    1,329,242 1,311,244 PROGRAM TOTALS
44 -----
45 -----
46 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
    DLmw          file:function
47 -----
48 146,800,887 29 29 52,428,894 20,485 5 20,971,551
    1,328,640 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
    matrix_multiply
49 -----
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
    DLmw
54 -----
55 -- line 18 -----

```



```

56 . . . . . matrix_t* create_matrix(size_t rows,
57 . . . . . size_t cols);
58 . . . . .
59 . . . . . void destroy_matrix(matrix_t* m);
60 . . . . .
61 . . . . . int print_matrix(FILE* fp, matrix_t*
62 . . . . . m);
63 . . . . .
64 . . . . . int
65 . . . . . main(int argc, char** argv)
66 10 2 2 0 0 0 5
67 0 0 {
68 . . . . . /* matrixes */
69 . . . . .
70 . . . . . matrix_t *a, *b, *c;
71 . . . . .
72 . . . . . /* n (dimension) and block size */
73 . . . . .
74 . . . . . size_t n, bs;
75 . . . . .
76 . . . . . /* line buffer (init to null to
77 . . . . . simplify freeing on error) */
78 1 0 0 0 0 0 1
79 0 0 0 char *line = NULL;
80 . . . . .
81 . . . . . /* line parsing auxiliar pointers
82 . . . . . */
83 . . . . .
84 . . . . . char *nptr, *endptr;
85 . . . . .
86 . . . . . /* auxiliar variables */
87 . . . . .
88 . . . . . long l;
89 . . . . .
90 . . . . . double e;
91 2 1 1 0 0 0 1
92 0 0 size_t lineno = 1;
93 . . . . .
94 . . . . . struct timespec t0;
95 . . . . .
96 . . . . . struct timespec t1;
97 . . . . .
98 . . . . . double dt;
99 . . . . .
100 . . . . . size_t i;

```

```

81      .      .      .      .      .      .      .      .
82      25      4      4      .      9      0      0      1
83      0      0      0      for(;; !feof(stdin)); lineno++) {
84      10      1      1      4      0      0      6
85      0      0      0      a=b=c=NULL;
86      .      .      .      .      .      .      .
87      18      3      3      .      8      1      0      2
88      0      0      0      line = read_line(stdin);
89      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .
91      2      1      1      .      /* parse dimension */
92      1      0      0      1
93      10      1      1      0      nptr = line;
94      0      1      1      0      3      1      0      1
95      8      1      1      0      l = strtol(nptr, &endptr, 10);
96      0      0      0      3      0      0      0
97      .      .      .      .      if (errno) {
98      .      .      .      .      perror("");
99      .      .      .      .      goto _exit_main;
100     .      .      .      .      }
101     4      2      2      .      2      0      0      0
102     0      0      0      if (nptr == endptr) {
103     .      .      .      .      fprintf(stderr, "missing
104     .      .      .      .      dimension");
105     .      .      .      .      goto _exit_main;
106     .      .      .      .      }
107     3      2      2      .      1      0      0      0
108     0      0      0      if (l < 1) {
109     .      .      .      .      fprintf(stderr, "invalid
110     .      .      .      .      dimension");
111     .      .      .      .      goto _exit_main;
112     .      .      .      .      }
113     2      1      1      .      1      0      0      1
114     0      0      0      n = (size_t) l;
115     .      .      .      .      .      .      .
116     .      .      .      .      #if 0

```

```

107         . . . . /* parse block size */
108         . . . .
109         . . . . nptr = endptr;
110         . . . . l = strtol(nptr, &endptr, 10);
111         . . . . if (errno) {
112         . . . .     . . . . perror("");
113         . . . .     . . . . goto _exit_main;
114         . . . .     . . . . }
115 -- line 75 -----
116 -- line 83 -----
117         . . . . }
118         . . . . bs = (size_t) 1;
119         . . . .
120         . . . . if (n % bs) {
121         . . . .     . . . . fprintf(stderr, "block size
doesn't match");
122         . . . .     . . . . goto _exit_main;
123         . . . .     . . . . }
124         . . . . #else
125         2 0 0 1 0 0 1
126         0 0 0 bs = n;
127         . . . . #endif
128         . . . .
129         . . . . /* load matrix a */
130         11 1 1 5 1 0 1
131         0 0 0 if (!(a = create_matrix(n, n)))
132         . . . .     . . . . goto _exit_main;
133         . . . .
134         20 3 3 7 0 0 2
135         0 0 0 for (i=0; i < n*n; i++) {
136         2 0 0 1 0 0 1
137         0 0 0 nptr = endptr;
138         9 1 1 3 0 0 1
139         0 0 0 e = strtod(nptr, &endptr);

```

```

134      8      1      1      0      3      0      0      0
      0      0      0      if (errno) {
135      .      .      .      .      .      .      .
      .      .      .      perror("");
136      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
137      .      .      .      .      .      .      .
      .      .      .      }
138      4      2      2      2      0      0      0
      0      0      0      if (nptr == endptr) {
139      .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "missing A
matrix element");
140      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
141      .      .      .      .      .      .      .
      .      .      .      }
142      7      1      1      4      0      0      1
      0      0      0      a->array[i] = e;
143      .      .      .      .      .      .      .
      .      .      .      }
144      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
145      .      .      .      .      .      .      .
      .      .      .      /* load matrix b */
146      11     1      1      5      0      0      1
      0      0      0      if (!(b = create_matrix(n, n)))
147      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
148      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
149      20     2      2      7      0      0      2
      0      0      0      for (i=0; i < n*n; i++) {
150      2      1      1      1      0      0      1
      0      0      0      nptr = endptr;
151      9      1      1      3      0      0      1
      0      0      0      e = strtod(nptr, &endptr);
152      8      1      1      3      0      0      0
      0      0      0      if (errno) {
153      .      .      .      .      .      .      .
      .      .      .      perror("");
154      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
155      .      .      .      .      .      .      .
      .      .      .      }
156      4      1      1      2      0      0      0
      0      0      0      if (nptr == endptr) {
157      .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "missing B
matrix element");
158      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
159      .      .      .      .      .      .      .
      .      .      .      }

```

```

160         7    1    1    0    4    0    0    1
           0        0    b->array[i] = e;
161     .    .    .    .    .    .    .    .
162     .    .    .    .    .    .    .    .
           .    .    .
163     8    1    1    0    2    0    0    0
           0        0    clock_gettime(CLOCK_REALTIME, &t0
           );
164     .    .    .    .    .    .    .    .
           .    .    .
165     .    .    .    .    .    .    .    .
           .    .    .    /* multiply matrixes */
166     13   2    2    0    6    1    1    1
           0        0    if (!(c = matrix_multiply(a, b, bs
           )))
167     .    .    .    .    .    .    .    .
           .    .    .    goto _exit_main;
168     .    .    .    .    .    .    .    .
           .    .    .
169     8    1    1    0    2    0    0    0
           0        0    clock_gettime(CLOCK_REALTIME, &t1
           );
170     .    .    .    .    .    .    .    .
           .    .    .
171     7    1    1    1    2    1    1    1
           1        1    dt = (float) (t1.tv_sec - t0.
           tv_sec);
172     12   1    1    0    5    2    2    1
           0        0    dt = dt + ((float)(t1.tv_nsec - t0
           .tv_nsec)) / 1.0e9;
173     .    .    .    .    .    .    .    .
           .    .    .
174     13   3    2    0    5    2    2    0
           0        0    if(print_matrix(stdout, c) == -1)
175     .    .    .    .    .    .    .    .
           .    .    .    goto _exit_main;
176     .    .    .    .    .    .    .    .
           .    .    .
177     12   1    1    0    7    0    0    0
           0        0    fprintf(stderr, "time: %g\n", dt);
178     .    .    .    .    .    .    .    .
           .    .    .
179     6    1    1    0    3    0    0    0
           0        0    free(line);
180     6    1    1    0    3    0    0    0
           0        0    destroy_matrix(a);
181     6    1    1    0    3    0    0    0
           0        0    destroy_matrix(b);
182     6    0    0    0    3    0    0    0
           0        0    destroy_matrix(c);
183     .    .    .    .    .    .    .    .
           .    .    .    }

```

```

184      .      .      .      .      .      .      .
185      3      1      1      .      0      0      0      0
186      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .
192      .      .      .      .      .      .      .
193      .      .      .      .      .      .      .
194      6      1      1      .      2      0      0      0
195      0      .      .      .      .      .      .      .
196      .      .      .      .      .      .      .
197      .      .      .      .      .      .      .
198      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .
202      .      .      .      .      .      .      .
203      3      1      1      .      2      0      0      1
204      0      .      .      .      .      .      .      .
205      11     1      1      .      5      0      0      1
206      0      .      .      .      .      .      .      .
207      9      2      2      .      3      0      0      1
208      0      .      .      .      .      .      .      .

```

```

209      16      2      2      5      0      0      2
          0      0      0      for(i=0; i<n; i++)
210      16      1      1      5      0      0      2
          0      0      0      for(j=0; j<n; j++)
211      11      1      1      5      0      0      2
          0      0      0      mr->array[i*n+j] = 0.0;
212      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
213      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
214      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
215      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
216      2      0      0      0      0      0      1
          0      0      0      size_t dim = 1024*1024*10;
217      9      1      1      3      0      0      1
          0      0      0      int *v = malloc(dim*sizeof(int));
218 83,886,088      2      2 31,457,282 20,480      0 10,485,761
          0      0      0      for (j = 0; j < dim; ++j)
219 62,914,560      0      0 20,971,520      0      0 10,485,760
          1,328,640 1,310,720      v[j] = -1;
220      6      1      1      3      2      2      0
          0      0      0      free(v);
221      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
222      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
223      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
224      17      2      2      6      0      0      2
          0      0      0      for(kk=0; kk<en; kk+=bs)
225      17      2      2      6      0      0      2
          0      0      0      for(jj=0; jj<en; jj+=bs)
226      16      1      1      5      0      0      2
          0      0      0      for(i=0; i<n; i++)
227      21      2      2      8      0      0      2
          0      0      0      for(j=jj; j<jj+bs; j++) {
228      11      1      1      6      1      1      1
          0      0      0      sum = mr->array[i*n+j];
229      21      3      3      8      0      0      2
          0      0      0      for(k=kk; k<kk+bs; k++)
230      24      2      2      13      2      2      1
          0      0      0      sum += m1->array[i*n+k] *
          m2->array[k*n+j];
231      11      1      1      6      0      0      1
          0      0      0      mr->array[i*n+j] = sum;
232      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
233      1      0      0      1      0      0      0
          0      0      0      return mr;
234      6      1      1      2      0      0      0
          0      0      0      }

```

```

235      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .
238      .      .      .      .      .      .      .
239      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .
242      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .
245      .      .      .      .      .      .      .
246      .      .      .      .      .      .      .
247      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .
251      .      .      .      .      .      .      .
252      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .

```

```

char*
read_line(FILE *fp)
{
#define DEF_LINE_SZ 1024

int c;
size_t len = 0, tam = DEF_LINE_SZ;
char* str;

str = malloc(tam);
if (!str) {
perror("");
return NULL;
}

while (EOF != (c=fgetc(fp)) && c !=
'\n') {
str[len++] = c;
if (len==tam-1) {
str = realloc(str, tam *= 2);
if (!str) {
perror("");
return NULL;
}
}
}
}

```



```

261      .      .      .      .      .      .      .
262      8      1      1      .      2      0      0      0
          0      0      0      if (c != EOF)
263      7      0      0      2      0      0      2
          0      0      str[len++]='\n';
264      .      .      .      .      .      .      .
265      12     1      1      .      4      0      0      4
          0      0      0      str[len++]='\0';
266      2      0      0      2      0      0      0
          0      0      return str;
267      12     1      1      4      0      0      0
          0      0      }
268      .      .      .      .      .      .      .
269      .      .      .      .      .      .      .
270      .      .      .      .      .      .      .
271      .      .      .      matrix_t*
272      .      .      .      create_matrix(size_t rows, size_t
          cols)
273      30     1      1      0      0      0      15
          0      0      0      {
274      .      .      .      matrix_t * m;
275      .      .      .      .      .      .      .
276      30     2      2      9      1      0      3
          0      0      if (!(m = malloc(sizeof(matrix_t))))
          {
277      .      .      .      .      .      .      .
278      .      .      .      perror("");
279      .      .      .      .      .      .      .
          .      .      .      return NULL;
          .      .      .      .      .      .      .
          .      .      .      }
          .      .      .      .      .      .      .
280      9      1      1      6      0      0      3
          0      0      0      m->rows = rows;
281      9      1      1      6      0      0      3
          0      0      0      m->cols = cols;
282      51     2      2      21     0      0      3
          0      0      if (!(m->array = malloc(sizeof(
          double)*rows*cols))) {
283      .      .      .      .      .      .      .
          .      .      .      free(m);
284      .      .      .      .      .      .      .
          .      .      .      perror("");
285      .      .      .      .      .      .      .
          .      .      .      return NULL;

```

```

286      .      .      .      .      .      .      .      .
287      .      .      .      .      .      .      .      .
288      3      1      1      .      3      0      0      0
289      0      .      0      return m;
18      18     1      1      .      6      0      0      0
290      0      .      0      }
291      .      .      .      .      .      .      .      .
292      .      .      .      .      .      .      .      .
293      .      .      .      .      .      .      .      .
27      27     1      1      .      0      0      0      12
294      0      .      0      {
9      9      0      0      .      3      0      0      0
295      0      .      0      if (!m) return;
24      24     1      1      .      12     0      0      0
296      0      .      0      free(m->array);
24      24     1      1      .      9      0      0      0
297      0      .      0      free(m);
18      18     0      0      .      6      0      0      0
298      0      .      0      }
299      .      .      .      .      .      .      .      .
300      .      .      .      .      .      .      .      .
301      .      .      .      .      .      .      .      .
10      10     2      2      .      0      0      0      5
302      0      .      0      {
303      .      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .      .
3      3      1      1      .      2      0      0      1
305      0      .      0      n = m->rows;
13      13     1      1      .      6      1      1      0
306      0      .      0      if (fprintf(fp, "%lu", (unsigned
long) m->rows) < 0) {
307      .      .      .      .      .      .      .      .
308      .      .      .      .      .      .      .      .
309      .      .      .      .      .      .      .      .
16      16     2      2      .      5      0      0      2
310      0      .      0      for(i=0; i<n; i++)
16      16     2      2      .      5      0      0      2
311      0      .      0      for (j=0; j<n; j++)
22      22     3      3      .      10     0      0      0
312      0      .      0      if (fprintf(fp, " %g", m->array[
i*n+j]) < 0) {

```

```

312         . . . . . perror("");
313         . . . . . return -1;
314         . . . . . }
315     10 1 1 4 0 0 0
316         0 0 if (fprintf(fp, "\n") < 0) {
317         . . . . . perror("");
318         . . . . . return -1;
319         . . . . . }
320     1 1 1 0 0 0 0
321         0 0 return 0;
322     6 1 1 2 0 0 0
323         0 0 }
-----
323 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
324         DLmw
-----
325 146,801,827 120 119 52,429,222 20,498 12 20,971,672
      1,328,643 1,310,721 events annotated

```

```

1 2 81 112 14 18
2 ind, a cache and branch-prediction profiler
3 ==601== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
      Nethercote et al.
4 ==601== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
      copyright info
5 ==601== Command: /tmp/02-mmult
6 ==601== Parent PID: 597
7 ==601==
8 --601-- Warning: Cannot auto-detect cache config, using
      defaults.
9 --601--          Run with -v to see.
10 ==601==
11 ==601== I   refs:      147,109,308
12 ==601== I1  misses:      2,401
13 ==601== LLi misses:      2,383
14 ==601== I1  miss rate:      0.00%
15 ==601== LLi miss rate:      0.00%
16 ==601==
17 ==601== D   refs:      73,522,032 (52,509,330 rd +
      21,012,702 wr)
18 ==601== D1  misses:      1,354,483 ( 25,153 rd +
      1,329,330 wr)
19 ==601== LLd misses:      1,314,145 ( 2,898 rd +
      1,311,247 wr)
20 ==601== D1  miss rate:      1.8% ( 0.0% +

```

```

21      6.3% )
==601== LLd miss rate:          1.8% (          0.0%  +
      6.2% )
22 ==601==
23 ==601== LL refs:             1,356,884 (    27,554 rd  +
      1,329,330 wr)
24 ==601== LL misses:          1,316,528 (    5,281 rd  +
      1,311,247 wr)
25 ==601== LL miss rate:          0.6% (          0.0%  +
      6.2% )
26 -----
27 I1 cache:                    32768 B, 32 B, 4-way associative
28 D1 cache:                    32768 B, 32 B, direct-mapped
29 LL cache:                    524288 B, 32 B, 8-way associative
30 Command:                     /tmp/02-mmult
31 Data file:                   cachegrind.out.601
32 Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds: 0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
39
40 -----
41 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
      DLmw
42 -----
43 147,109,308 2,401 2,383 52,509,330 25,153 2,898 21,012,702
      1,329,330 1,311,247 PROGRAM TOTALS
44 -----
45 -----
46 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
      DLmw          file:function
47 -----
48 146,801,346 29 29 52,429,109 20,487 7 20,971,590
      1,328,640 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
      matrix_multiply
49 -----
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
      DLmw
54 -----
55 -- line 18 -----
56 . . . . .

```

```

.      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
57      .      .      .      .      .      .      .      .
58      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
59      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
60      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
61      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
62      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
63      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
64      .      .      .      .      .      .      .      .
10      2      2      0      0      0      0      5
0      0      0      {
65      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
69      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
70      1      0      0      0      0      0      0      1
0      0      0      char *line = NULL;
71      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
76      2      1      1      0      0      0      0      1
0      0      0      size_t lineno = 1;
77      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .

```

```

82      25      4      4      .      9      0      0      1
      0      0      0      for(;; !feof(stdin); lineno++) {
83      10      1      1      4      0      0      6
      0      0      a=b=c=NULL;
84      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
85      18      3      3      .      8      1      0      2
      0      0      0      line = read_line(stdin);
86      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
87      6      0      0      0      2      0      0      0
      0      0      0      if (!line) goto _exit_main;
88      9      0      0      0      4      0      0      0
      0      0      0      if (line[0] == 0) break;
89      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
91      2      1      1      .      .      .      .      .
      1      0      0      1      nptr = line;
92      10      1      1      0      3      1      0      1
      0      0      0      l = strtol(nptr, &endptr, 10);
93      8      1      1      0      3      0      0      0
      0      0      0      if (errno) {
94      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
97      4      2      2      .      2      0      0      0
      0      0      0      if (nptr == endptr) {
98      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      dimension");
99      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
100     .      .      .      .      .      .      .
      .      .      .      .      .      .      .
101     3      2      2      .      1      0      0      0
      0      0      0      if (l < 1) {
102     .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      dimension");
103     .      .      .      .      .      .      .
      .      .      .      .      .      .      .
104     .      .      .      .      .      .      .
      .      .      .      .      .      .      .
105     2      1      1      .      1      0      0      1
      0      0      0      n = (size_t) l;
106     .      .      .      .      .      .      .
      .      .      .      .      .      .      .
107     .      .      .      .      .      .      .
      .      .      .      .      .      .      .

```

```

108      .      .      .      /* parse block size */
109      .      .      .      nptr = endptr;
110      .      .      .      l = strtol(nptr, &endptr, 10);
111      .      .      .      if (errno) {
112      .      .      .      .      perror("");
113      .      .      .      .      goto _exit_main;
114      .      .      .      .      }
115  -- line 75 -----
116  -- line 83 -----
117      .      .      .      }
118      .      .      .      bs = (size_t) l;
119      .      .      .      .
120      .      .      .      if (n % bs) {
121      .      .      .      .      fprintf(stderr, "block size
doesn't match");
122      .      .      .      .      goto _exit_main;
123      .      .      .      .      }
124      .      .      .      #else
125      2      0      0      1      0      0      1
126      0      0      0      bs = n;
127      .      .      .      #endif
128      .      .      .      .
129      .      .      .      /* load matrix a */
130      11      1      1      5      1      0      1
131      0      0      0      if (!(a = create_matrix(n, n)))
132      .      .      .      .      goto _exit_main;
133      .      .      .      .
134      50      3      3      19      0      0      5
135      0      0      0      for (i=0; i < n*n; i++) {
136      8      0      0      4      0      0      4
137      0      0      0      nptr = endptr;
138      36      1      1      12      0      0      4
139      0      0      0      e = strtod(nptr, &endptr);
140      32      1      1      12      0      0      0
141      0      0      0      if (errno) {

```

```

135         .      .      .      .      .      .      .      .
136         .      .      .      .      .      .      .      .
137         .      .      .      .      .      .      .      .
138         .      .      .      .      .      .      .      .
139         .      .      .      .      .      .      .      .
140         .      .      .      .      .      .      .      .
141         .      .      .      .      .      .      .      .
142         .      .      .      .      .      .      .      .
143         .      .      .      .      .      .      .      .
144         .      .      .      .      .      .      .      .
145         .      .      .      .      .      .      .      .
146         .      .      .      .      .      .      .      .
147         .      .      .      .      .      .      .      .
148         .      .      .      .      .      .      .      .
149         .      .      .      .      .      .      .      .
150         .      .      .      .      .      .      .      .
151         .      .      .      .      .      .      .      .
152         .      .      .      .      .      .      .      .
153         .      .      .      .      .      .      .      .
154         .      .      .      .      .      .      .      .
155         .      .      .      .      .      .      .      .
156         .      .      .      .      .      .      .      .
157         .      .      .      .      .      .      .      .
158         .      .      .      .      .      .      .      .
159         .      .      .      .      .      .      .      .
160         .      .      .      .      .      .      .      .

        perror("");
        goto _exit_main;
    }
16  2  2  8  0  0  0
    0  0  if (nptr == endptr) {
        fprintf(stderr, "missing A
matrix element");
        goto _exit_main;
    }
28  1  1  16  0  0  4
    0  0  a->array[i] = e;
    }
    }
    }
    /* load matrix b */
11  1  1  5  0  0  1
    0  0  if (!(b = create_matrix(n, n)))
        goto _exit_main;
    }
50  2  2  19  0  0  5
    0  0  for (i=0; i < n*n; i++) {
18  1  1  4  0  0  4
    0  0  nptr = endptr;
36  1  1  12  0  0  4
    0  0  e = strtod(nptr, &endptr);
32  1  1  12  0  0  0
    0  0  if (errno) {
        perror("");
        goto _exit_main;
    }
16  1  1  8  0  0  0
    0  0  if (nptr == endptr) {
        fprintf(stderr, "missing B
matrix element");
        goto _exit_main;
    }
28  1  1  16  0  0  4
    1  1  b->array[i] = e;

```



```

161      .      .      .      .      .      .      .      .
162      .      .      .      .      .      .      .      .
163      8      1      1      .      2      0      0      0
           0      0      clock_gettime(CLOCK_REALTIME, &t0
           );
164      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
166      13     2      2      .      6      1      1      1
           0      0      if (!(c = matrix_multiply(a, b, bs
           )))
167      .      .      .      .      .      .      .      .
168      .      .      .      .      goto _exit_main;
169      .      .      .      .      .      .      .      .
170      8      1      1      .      2      0      0      0
           0      0      clock_gettime(CLOCK_REALTIME, &t1
           );
171      7      1      1      .      2      1      1      1
           1      1      dt = (float) (t1.tv_sec - t0.
           tv_sec);
172      12     1      1      .      5      2      2      1
           0      0      dt = dt + ((float)(t1.tv_nsec - t0
           .tv_nsec)) / 1.0e9;
173      .      .      .      .      .      .      .      .
174      13     3      2      .      5      2      2      0
           0      0      if(print_matrix(stdout, c) == -1)
175      .      .      .      .      .      .      .      .
176      .      .      .      .      goto _exit_main;
177      .      .      .      .      .      .      .      .
178      12     1      1      .      7      0      0      0
           0      0      fprintf(stderr, "time: %g\n", dt);
179      .      .      .      .      .      .      .      .
180      6      1      1      .      3      0      0      0
           0      0      free(line);
181      6      1      1      .      3      0      0      0
           0      0      destroy_matrix(a);
182      6      1      1      .      3      0      0      0
           0      0      destroy_matrix(b);
183      6      0      0      .      3      0      0      0
           0      0      destroy_matrix(c);
184      .      .      .      .      .      .      .      .
185      3      1      1      .      0      0      0      0

```

```

186         0      0      return 0;
187         .      .      .      .      .      .
188         .      .      .      .      .      .
189         .      .      .      .      .      .
190         .      .      .      .      .      .
191         .      .      .      .      .      .
192         .      .      .      .      .      .
193         .      .      .      .      .      .
194         6      1      1      2      0      0      0
195         0      0      0      }
196         .      .      .      .      .      .
197         .      .      .      .      .      .
198         .      .      .      .      .      .
199         .      .      .      .      .      .
200         .      .      .      .      .      .
201         .      .      .      .      .      .
202         .      .      .      .      .      .
203         3      1      1      2      0      0      1
204         0      0      0      n = m1->rows;
205         .      .      .      .      .      .
206         .      .      .      .      .      .
207         9      2      2      3      0      0      1
208         0      0      0      en = bs*(n/bs);
209         .      .      .      .      .      .
210         24     2      2      8      0      0      3
211         0      0      0      for(i=0; i<n; i++)
212         48     1      1      16     0      0      6
213         0      0      0      for(j=0; j<n; j++)

```

```

211      44      1      1      20      0      0      8
          0      0      mr->array[i*n+j] = 0.0;
212      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
213      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
214      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
215      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
216      2      0      0      0      0      0      1
          0      0      0      size_t dim = 1024*1024*10;
217      9      1      1      3      0      0      1
          0      0      int *v = malloc(dim*sizeof(int));
218 83,886,088      2      2 31,457,282 20,480      0 10,485,761
          0      0      for (j = 0; j < dim; ++j)
219 62,914,560      0      0 20,971,520      0      0 10,485,760
          1,328,640 1,310,720      v[j] = -1;
220      6      1      1      3      2      2      0
          0      0      free(v);
221      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
222      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
223      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
224      17      2      2      6      0      0      2
          0      0      for(kk=0; kk<en; kk+=bs)
225      17      2      2      6      0      0      2
          0      0      for(jj=0; jj<en; jj+=bs)
226      24      1      1      8      0      0      3
          0      0      for(i=0; i<n; i++)
227      62      2      2      24      0      0      6
          0      0      for(j=jj; j<jj+bs; j++) {
228      44      1      1      24      2      2      4
          0      0      sum = mr->array[i*n+j];
229      124      3      3      48      0      0      12
          0      0      for(k=kk; k<kk+bs; k++)
230      192      2      2      104      3      3      8
          0      0      sum += m1->array[i*n+k] *
          m2->array[k*n+j];
231      44      1      1      24      0      0      4
          0      0      mr->array[i*n+j] = sum;
232      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
233      1      0      0      1      0      0      0
          0      0      return mr;
234      6      1      1      2      0      0      0
          0      0      }
235      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
236      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
237      .      .      .      .      .      .      .
          .      .      .      .      .      .      .

```

	18	3	3	.	read_line(FILE *fp)				8
238		0		0	{				
239
240	#define DEF_LINE_SZ 1024
241
242	6	0	0	.	int c;				
243		0		0	size_t len = 0, tam = DEF_LINE_SZ;				
244	char* str;
245	14	1	1	.	6 1 0				2
246		1		0	str = malloc(tam);				
247	6	0	0	0	2 0 0				0
248		0		0	if (!str) {				
249	perror("");
250	return NULL;
251		.	.	.	}
252	281	5	5	.	94 1 0				19
253		0		0	while (EOF != (c=fgetc(fp)) && c !=				
254		'\n')		{					
255	136	2	2	.	51 0 0				34
256		0		0	str[len++]=c;				
257	85	0	0	0	34 0 0				0
258		0		0	if (len==tam-1) {				
259	str = realloc(str, tam *= 2);
260	if (!str) {
261	perror("");
262	return NULL;
263	}
264	}
265	}
266	}
267	8	1	1	.	2 0 0				0
268		0		0	if (c != EOF)				
269	7	0	0	0	2 0 0				2
270		0		0	str[len++]='\\n';				

```

264      .      .      .      .      .      .      .
265      12      1      1      .      4      0      0      4
266      0      0      0      str[len++]='\0';
267      2      0      0      2      0      0      0
268      0      0      0      return str;
269      12      1      1      4      0      0      0
270      0      0      0      }
271      .      .      .      .      .      .      .
272      .      .      .      .      .      .      .
273      .      .      .      .      .      .      .
274      .      .      .      .      .      .      .
275      .      .      .      .      .      .      .
276      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .
278      .      .      .      .      .      .      .
279      .      .      .      .      .      .      .
280      .      .      .      .      .      .      .
281      .      .      .      .      .      .      .
282      .      .      .      .      .      .      .
283      .      .      .      .      .      .      .
284      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .
286      .      .      .      .      .      .      .
287      .      .      .      .      .      .      .
288      .      .      .      .      .      .      .
289      .      .      .      .      .      .      .

```

```

290      .      .      .      .      .      .      .
291      .      .      .      .      .      .      .
292      .      .      .      .      .      .      .
293      .      .      .      .      .      .      .
294      .      .      .      .      .      .      .
295      .      .      .      .      .      .      .
296      .      .      .      .      .      .      .
297      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .
299      .      .      .      .      .      .      .
300      .      .      .      .      .      .      .
301      .      .      .      .      .      .      .
302      .      .      .      .      .      .      .
303      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .
305      .      .      .      .      .      .      .
306      .      .      .      .      .      .      .
307      .      .      .      .      .      .      .
308      .      .      .      .      .      .      .
309      .      .      .      .      .      .      .
310      .      .      .      .      .      .      .
311      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .
314      .      .      .      .      .      .      .
315      .      .      .      .      .      .      .

```

```

316         0      0      if (fprintf(fp, "\n") < 0) {
317             .      .      .      .      .      .      .
318             .      .      .      .      .      .      .
319             .      .      .      .      .      .      .
320             1      1      1      0      0      0      0
321             0      0      0      0      0      0      0
322             6      1      1      2      0      0      0
323             0      0      0      0      0      0      0
324             }
325
-----
326 Ir      I1mr ILmr Dr      D1mr  DLmr Dw      D1mw
327      DLmw
328
-----
329 146,802,968 120 119 52,429,703 20,500 14 20,971,776
330 1,328,644 1,310,722 events annotated
331

```

```

1 3 10 29 43 26 70 95 29 60 76
2 nd branch-prediction profiler
3 ==605== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
4   Nethercote et al.
5 ==605== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
6   copyright info
7 ==605== Command: /tmp/02-mmult
8 ==605== Parent PID: 597
9 ==605==
10 --605-- Warning: Cannot auto-detect cache config, using
11 defaults.
12 --605-- Run with -v to see.
13 ==605==
14 ==605== I refs: 147,126,965
15 ==605== I1 misses: 2,407
16 ==605== LLi misses: 2,389
17 ==605== I1 miss rate: 0.00%
18 ==605== LLi miss rate: 0.00%
19 ==605==
20 ==605== D refs: 73,528,535 (52,513,741 rd +
21 21,014,794 wr)
22 ==605== D1 misses: 1,354,211 ( 24,912 rd +
23 1,329,299 wr)
24 ==605== LLd misses: 1,314,153 ( 2,901 rd +
25 1,311,252 wr)
26 ==605== D1 miss rate: 1.8% ( 0.0% +
27 6.3% )
28 ==605== LLd miss rate: 1.8% ( 0.0% +
29 6.2% )
30 ==605==
31 ==605== LL refs: 1,356,618 ( 27,319 rd +
32 1,329,299 wr)
33 ==605== LL misses: 1,316,542 ( 5,290 rd +

```

```

1,311,252 wr)
==605== LL miss rate:          0.6% (          0.0%      +
        6.2%  )

-----

I1 cache:          32768 B, 32 B, 4-way associative
D1 cache:          32768 B, 32 B, direct-mapped
LL cache:          524288 B, 32 B, 8-way associative
Command:           /tmp/02-mmult
Data file:          cachegrind.out.605
Events recorded:    Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Events shown:       Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Event sort order:   Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Thresholds:         0.1 100 100 100 100 100 100 100 100
Include dirs:
User annotated:     /root/CARPETA/tp2-2020-2q-src/main.c
Auto-annotation:    off

-----

Ir          I1mr  ILmr  Dr          D1mr   DLmr  Dw          D1mw
          DLmw

-----

147,126,965 2,407 2,389 52,513,741 24,912 2,901 21,014,794
1,329,299 1,311,252  PROGRAM TOTALS

-----

Ir          I1mr  ILmr  Dr          D1mr   DLmr  Dw          D1mw
          DLmw          file:function

-----

146,802,337  29    29 52,429,584 20,491  11 20,971,667
1,328,641 1,310,721  /root/CARPETA/tp2-2020-2q-src/main.c:
matrix_multiply

-----

-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
-----

Ir          I1mr  ILmr  Dr          D1mr   DLmr  Dw          D1mw
          DLmw

-- line 18 -----
.      .      .      .      .      .      .
.      .      .      .      .      .      .
matrix_t* create_matrix(size_t rows,
size_t cols);
.      .      .      .      .      .      .
.      .      .      .      .      .      .
.      .      .      .      .      .      .
.      .      .      .      .      .      .
void destroy_matrix(matrix_t* m);
.      .      .      .      .      .      .

```



```

60      .          .      .      int print_matrix(FILE* fp, matrix_t*
        m);
61      .          .          .          .          .          .
62      .          .          .          .          .          .
63      .          .          .      int
64      .          .          .      main(int argc, char** argv)
10    2      2      0      0      0      5
        0      0 {
65      .          .          .          .          .          .
66      .          .          .          /* matrixes */
67      .          .          .          matrix_t *a, *b, *c;
68      .          .          .          /* n (dimension) and block size */
69      .          .          .          size_t n, bs;
        .          .          .          /* line buffer (init to null to
simplify freeing on error) */
70    1      0      0      0      0      0      1
        0      0      0      char *line = NULL;
71      .          .          .          /* line parsing auxiliar pointers
*/
72      .          .          .          char *nptr, *endptr;
73      .          .          .          /* auxiliar variables */
74      .          .          .          long l;
75      .          .          .          double e;
76    2      1      1      0      0      0      1
        0      0      0      size_t lineno = 1;
77      .          .          .          struct timespec t0;
78      .          .          .          struct timespec t1;
79      .          .          .          double dt;
80      .          .          .          size_t i;
81      .          .          .          .          .          .
82    25     4      4      9      0      0      1
        0      0      0      for(;; !feof(stdin)); lineno++) {
83    10     1      1      4      0      0      6
        0      0      0      a=b=c=NULL;
84      .          .          .          .          .          .

```

```

85      18      3      3      8      2      0      2
      0      0      line = read_line(stdin);
86      .      .      .      .      .      .      .
      .      .      .
87      6      0      0      2      0      0      0
      0      0      if (!line) goto _exit_main;
88      9      0      0      4      0      0      0
      0      0      if (line[0] == 0) break;
89      .      .      .      .      .      .      .
      .      .      .
90      .      .      .      .      .      .      .
      .      .      .      /* parse dimension */
91      2      1      1      1      0      0      1
      1      0      nptr = line;
92      10     1      1      3      1      0      1
      0      0      l = strtol(nptr, &endptr, 10);
93      8      1      1      3      0      0      0
      0      0      if (errno) {
94      .      .      .      .      .      .      .
      .      .      .      perror("");
95      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
96      .      .      .      .      .      .      .
      .      .      .      }
97      4      2      2      2      0      0      0
      0      0      if (nptr == endptr) {
98      .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "missing
      dimension");
99      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
100     .      .      .      .      .      .      .
      .      .      .      }
101     3      2      2      1      0      0      0
      0      0      if (l < 1) {
102     .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "invalid
      dimension");
103     .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
104     .      .      .      .      .      .      .
      .      .      .      }
105     2      1      1      1      0      0      1
      0      0      n = (size_t) l;
106     .      .      .      .      .      .      .
      .      .      .      #if 0
107     .      .      .      .      .      .      .
      .      .      .      /* parse block size */
108     .      .      .      .      .      .      .
      .      .      .      nptr = endptr;
109     .      .      .      .      .      .      .
      .      .      .      l = strtol(nptr, &endptr, 10);
110     .      .      .      .      .      .      .
      .      .      .      if (errno) {

```

```

111         . . . . .
112         . . . . . perror("");
113         . . . . . goto _exit_main;
114         . . . . . }
115 -- line 75 -----
116 -- line 83 -----
117         . . . . . }
118         . . . . . bs = (size_t) 1;
119         . . . . .
120         . . . . . if (n % bs) {
121         . . . . .     fprintf(stderr, "block size
122         . . . . .     doesn't match");
123         . . . . .     goto _exit_main;
124         . . . . . }
125         . . . . . #else
126         . . . . .     2 0 0 1 0 0 1
127         . . . . .     0 0 0 bs = n;
128         . . . . . #endif
129         . . . . .
130         . . . . . /* load matrix a */
131         . . . . . 11 1 1 5 1 0 1
132         . . . . . 0 0 0 if (!(a = create_matrix(n, n)))
133         . . . . .     goto _exit_main;
134         . . . . .
135         . . . . . 100 3 3 39 0 0 10
136         . . . . . 0 0 0 for (i=0; i < n*n; i++) {
137         . . . . . 18 0 0 9 0 0 9
138         . . . . . 0 0 0 nptr = endptr;
139         . . . . . 81 1 1 27 0 0 9
140         . . . . . 0 0 0 e = strtod(nptr, &endptr);
141         . . . . . 72 1 1 27 0 0 0
142         . . . . . 0 0 0 if (errno) {
143         . . . . .     perror("");
144         . . . . .     goto _exit_main;
145         . . . . . }
146         . . . . . 36 2 2 18 0 0 0

```

```

139         . . . . . if (nptr == endptr) {
        . . . . . fprintf(stderr, "missing A
        matrix element");
140     . . . . .
        . . . . . goto _exit_main;
141     . . . . .
        . . . . . }
142 63     1     1     36     0     0     9
        1     1     a->array[i] = e;
143     . . . . .
        . . . . . }
144     . . . . .
        . . . . .
145     . . . . .
        . . . . . /* load matrix b */
146 11     1     1     5     0     0     1
        0     0     if (!(b = create_matrix(n, n)))
147     . . . . .
        . . . . . goto _exit_main;
148     . . . . .
        . . . . .
149 100    2     2     39     0     0     10
        0     0     for (i=0; i < n*n; i++) {
150 18     1     1     9     0     0     9
        0     0     nptr = endptr;
151 81     1     1     27     0     0     9
        0     0     e = strtod(nptr, &endptr);
152 72     1     1     27     0     0     0
        0     0     if (errno) {
153     . . . . .
        . . . . . perror("");
154     . . . . .
        . . . . . goto _exit_main;
155     . . . . .
        . . . . . }
156 36     1     1     18     0     0     0
        0     0     if (nptr == endptr) {
157     . . . . .
        . . . . . fprintf(stderr, "missing B
        matrix element");
158     . . . . .
        . . . . . goto _exit_main;
159     . . . . .
        . . . . .
        . . . . . }
160 63     1     1     36     0     0     9
        1     1     b->array[i] = e;
161     . . . . .
        . . . . . }
162     . . . . .
        . . . . .
163 8     1     1     2     0     0     0
        0     0     clock_gettime(CLOCK_REALTIME, &to
        );

```

```

164      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .
166      13      2      2      .      /* multiply matrixes */
          0      0      if (!(c = matrix_multiply(a, b, bs
          )))
167      .      .      .      .      .      .      .
          .      .      goto _exit_main;
168      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
169      8      1      1      .      2      0      0      0
          0      0      clock_gettime(CLOCK_REALTIME, &t1
          );
170      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
171      7      1      1      .      2      1      1      1
          1      1      dt = (float) (t1.tv_sec - t0.
          tv_sec);
172      12      1      1      .      5      2      2      1
          0      0      dt = dt + ((float)(t1.tv_nsec - t0
          .tv_nsec)) / 1.0e9;
173      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
174      13      3      2      .      5      2      2      0
          0      0      if(print_matrix(stdout, c) == -1)
175      .      .      .      .      .      .      .
          .      .      goto _exit_main;
176      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
177      12      1      1      .      7      0      0      0
          0      0      fprintf(stderr, "time: %g\n", dt);
178      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
179      6      1      1      .      3      0      0      0
          0      0      free(line);
180      6      1      1      .      3      0      0      0
          0      0      destroy_matrix(a);
181      6      1      1      .      3      0      0      0
          0      0      destroy_matrix(b);
182      6      0      0      .      3      0      0      0
          0      0      destroy_matrix(c);
183      .      .      .      .      .      .      .
          .      .      .      }
184      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
185      3      1      1      .      0      0      0      0
          0      0      return 0;
186      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
187      .      .      .      .      .      .      .
          .      .      .      _exit_main:
188      .      .      .      .      .      .      .
          .      .      .      fprintf(stderr, " at line %u\n", (

```

```

189         unsigned) lineno);
190         .      .      .      .      .      .      .
191         .      .      .      free(line);
192         .      .      .      .      .      .      .
193         .      .      .      destroy_matrix(a);
194         .      .      .      .      .      .      .
195         .      .      .      destroy_matrix(b);
196         .      .      .      .      .      .      .
197         .      .      .      destroy_matrix(c);
198         .      .      .      .      .      .      .
199         .      .      .      exit(1);
200         6      1      1      2      0      0      0
201         0      0      0      }
202         .      .      .      .      .      .      .
203         .      .      .      .      .      .      .
204         .      .      .      matrix_t*
205         .      .      .      .      .      .      .
206         .      .      .      matrix_multiply(matrix_t* m1,
207         .      .      .      matrix_t* m2, int bs)
208         11     2      2      0      0      0      6
209         0      0      0      {
210         .      .      .      .      .      .      .
211         .      .      .      size_t n, en, i, j, k, kk, jj;
212         .      .      .      .      .      .      .
213         .      .      .      double sum;
214         .      .      .      .      .      .      .
215         .      .      .      matrix_t* mr;
216         .      .      .      .      .      .      .
217         3      1      1      2      0      0      1
218         0      0      0      n = m1->rows;
219         .      .      .      .      .      .      .
220         11     1      1      5      0      0      1
221         0      0      0      if(!(mr = create_matrix(n,n)))
222         return NULL;
223         .      .      .      .      .      .      .
224         .      .      .      .      .      .      .
225         9      2      2      3      0      0      1
226         0      0      0      en = bs*(n/bs);
227         .      .      .      .      .      .      .
228         .      .      .      .      .      .      .
229         32     2      2      11     0      0      4
230         0      0      0      for(i=0; i<n; i++)
231         96     1      1      33     0      0      12
232         0      0      0      for(j=0; j<n; j++)
233         99     1      1      45     0      0      18
234         1      1      1      mr->array[i*n+j] = 0.0;
235         .      .      .      .      .      .      .
236         .      .      .      .      .      .      .
237         .      .      .      .      .      .      .
238         .      .      .      #if 1
239         .      .      .      .      .      .      .

```

```

215         .         .         .         if (1) {
216         .         .         .         size_t j;
217         2         0         0         0         0         0         1
218         0         0         0         size_t dim = 1024*1024*10;
219         9         1         1         3         0         0         1
220         0         0         0         int *v = malloc(dim*sizeof(int));
221         83,886,088         2         2         31,457,282         20,480         0         10,485,761
222         0         0         0         for (j = 0; j < dim; ++j)
223         62,914,560         0         0         20,971,520         0         0         10,485,760
224         1,328,640         1,310,720         v[j] = -1;
225         6         1         1         3         2         2         0
226         0         0         0         free(v);
227         .         .         .         .         .         .         .
228         .         .         .         .         .         .         .
229         .         .         .         #endif
230         .         .         .         .         .         .         .
231         17         2         2         6         0         0         2
232         0         0         0         for(kk=0; kk<en; kk+=bs)
233         17         2         2         6         0         0         2
234         0         0         0         for(jj=0; jj<en; jj+=bs)
235         32         1         1         11         0         0         4
236         0         0         0         for(i=0; i<n; i++)
237         123         2         2         48         0         0         12
238         0         0         0         for(j=jj; j<jj+bs; j++) {
239         99         1         1         54         3         3         9
240         0         0         0         sum = mr->array[i*n+j];
241         369         3         3         144         0         0         36
242         0         0         0         for(k=kk; k<kk+bs; k++)
243         648         2         2         351         6         6         27
244         0         0         0         sum += m1->array[i*n+k] *
245         m2->array[k*n+j];
246         99         1         1         54         0         0         9
247         0         0         0         mr->array[i*n+j] = sum;
248         .         .         .         .         .         .         .
249         .         .         .         .         .         .         .
250         1         0         0         1         0         0         0
251         0         0         0         return mr;
252         6         1         1         2         0         0         0
253         0         0         0         }
254         .         .         .         .         .         .         .
255         .         .         .         .         .         .         .
256         .         .         .         char*
257         .         .         .         .         .         .         .
258         .         .         .         read_line(FILE *fp)
259         18         3         3         0         0         0         8
260         0         0         0         {
261         .         .         .         .         .         .         .
262         .         .         .         #define DEF_LINE_SZ 1024
263         .         .         .         .         .         .         .

```

```

241      .      .      .      .      .      .      .
242      6      0      0      .      int c;
243      0      0      0      .      0      0      0      4
244      .      .      .      .      size_t len = 0, tam = DEF_LINE_SZ;
245      .      .      .      .      char* str;
246      .      .      .      .      .      .      .
247      14     1     1     .      6      1      0      2
248      1      0      0      0      str = malloc(tam);
249      6      0      0      0      2      0      0      0
250      0      0      0      0      if (!str) {
251      .      .      .      .      .      .      .
252      .      .      .      .      perror("");
253      .      .      .      .      .      .      .
254      .      .      .      .      return NULL;
255      .      .      .      .      .      .      .
256      .      .      .      .      }
257      .      .      .      .      .      .      .
258      581    5     5     .      194     1      0      39
259      0      0      0      0      while (EOF != (c=fgetc(fp)) && c !=
260      '\n') {
261      296    2     2     .      111     0      0      74
262      1      1     .      .      str[len++] = c;
263      185    0     0     .      74      0      0      0
264      0      0      0      0      if (len==tam-1) {
265      .      .      .      .      .      .      .
266      .      .      .      .      str = realloc(str, tam *= 2);
267      .      .      .      .      .      .      .
268      .      .      .      .      if (!str) {
269      .      .      .      .      .      .      .
270      .      .      .      .      perror("");
271      .      .      .      .      .      .      .
272      .      .      .      .      return NULL;
273      .      .      .      .      .      .      .
274      .      .      .      .      }
275      .      .      .      .      .      .      .
276      .      .      .      .      }
277      .      .      .      .      .      .      .
278      .      .      .      .      }
279      .      .      .      .      .      .      .
280      8      1     1     .      2      0      0      0
281      0      0      0      0      if (c != EOF)
282      7      0      0      0      2      0      0      2
283      0      0      0      0      str[len++] = '\n';
284      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .
286      12     1     1     .      4      0      0      4
287      0      0      0      0      str[len++] = '\0';
288      2      0      0      0      2      0      0      0
289      0      0      0      0      return str;
290      12     1     1     .      4      0      0      0

```



```

268     . . . . .
269     . . . . .
270     . . . . .
271     . . . . . matrix_t*
272     . . . . . create_matrix(size_t rows, size_t
273         cols)
274     30 1 1 0 0 0 15
275     0 0 {
276     . . . . . matrix_t * m;
277     . . . . .
278     30 2 2 9 1 0 3
279     0 0 if (!(m = malloc(sizeof(matrix_t))))
280     {
281     . . . . .
282     . . . . . perror("");
283     . . . . .
284     . . . . . return NULL;
285     . . . . .
286     . . . . . }
287     . . . . .
288     9 1 1 6 0 0 3
289     0 0 m->rows = rows;
290     9 1 1 6 0 0 3
291     0 0 m->cols = cols;
292     51 2 2 21 0 0 3
293     0 0 if (!(m->array = malloc(sizeof(
294         double)*rows*cols))) {
295     . . . . .
296     . . . . . free(m);
297     . . . . .
298     . . . . . perror("");
299     . . . . .
300     . . . . . return NULL;
301     . . . . .
302     . . . . . }
303     . . . . .
304     . . . . .
305     3 1 1 3 0 0 0
306     0 0 return m;
307     18 1 1 6 0 0 0
308     0 0 }
309     . . . . .
310     . . . . .
311     . . . . . void
312     . . . . .
313     . . . . . destroy_matrix(matrix_t* m)

```

```

293 27 1 1 0 0 0 12
      0 0 0 {
294 9 0 0 3 0 0 0
      0 0 if (!m) return;
295 24 1 1 12 0 0 0
      0 0 free(m->array);
296 24 1 1 9 0 0 0
      0 0 free(m);
297 18 0 0 6 0 0 0
      0 0 }
298 . . . . .
299 . . . . .
      . int
300 . . . . .
      . print_matrix(FILE* fp, matrix_t* m)
301 10 2 2 0 0 0 5
      0 0 {
302 . . . . .
      . size_t i, j;
303 . . . . .
      . size_t n;
304 3 1 1 2 0 0 1
      0 0 n = m->rows;
305 13 1 1 6 1 1 0
      0 0 if (fprintf(fp, "%lu", (unsigned
      long) m->rows) < 0) {
306 . . . . .
      . perror("");
307 . . . . .
      . return -1;
308 . . . . .
      . }
309 32 2 2 11 0 0 4
      0 0 for(i=0; i<n; i++)
310 96 2 2 33 0 0 12
      0 0 for (j=0; j<n; j++)
311 198 3 3 90 1 0 0
      0 0 if (fprintf(fp, " %g", m->array[i
      *n+j]) < 0) {
312 . . . . .
      . perror("");
313 . . . . .
      . return -1;
314 . . . . .
      . }
315 10 1 1 4 0 0 0
      0 0 if (fprintf(fp, "\n") < 0) {
316 . . . . .
      . perror("");
317 . . . . .
      . return -1;
318 . . . . .
      . }

```

```

319         1      1      1      0      0      0      0
320             0      0      return 0;
321         6      1      1      2      0      0      0
322             0      0      }
-----
323 Ir          I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
324         DLmw
-----
325 146,805,085  120  119 52,430,618 20,506   18 20,971,960
      1,328,647 1,310,725  events annotated

```

```

1 4 97 46 101 53 76 89 135 88 113 87 126 89 90 88 142 102
2 er
3 ==611== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
      Nethercote et al.
4 ==611== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
      copyright info
5 ==611== Command: /tmp/02-mmult
6 ==611== Parent PID: 597
7 ==611==
8 --611-- Warning: Cannot auto-detect cache config, using
      defaults.
9 --611--          Run with -v to see.
10 ==611==
11 ==611== I   refs:          147,153,617
12 ==611== I1  misses:          2,405
13 ==611== LLi misses:          2,387
14 ==611== I1  miss rate:          0.00%
15 ==611== LLi miss rate:          0.00%
16 ==611==
17 ==611== D   refs:          73,538,418 (52,520,450 rd  +
      21,017,968 wr)
18 ==611== D1  misses:          1,354,045 (  24,800 rd  +
      1,329,245 wr)
19 ==611== LLd misses:          1,314,166 (  2,907 rd  +
      1,311,259 wr)
20 ==611== D1  miss rate:          1.8% (  0.0%  +
      6.3%  )
21 ==611== LLd miss rate:          1.8% (  0.0%  +
      6.2%  )
22 ==611==
23 ==611== LL refs:          1,356,450 (  27,205 rd  +
      1,329,245 wr)
24 ==611== LL misses:          1,316,553 (   5,294 rd  +
      1,311,259 wr)
25 ==611== LL miss rate:          0.6% (  0.0%  +
      6.2%  )
26 -----
27 I1 cache:          32768 B, 32 B, 4-way associative
28 D1 cache:          32768 B, 32 B, direct-mapped

```

```

29 LL cache:          524288 B, 32 B, 8-way associative
30 Command:          /tmp/02-mmult
31 Data file:        cachegrind.out.611
32 Events recorded:  Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:     Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:       0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:   /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation:  off
39
40 -----
41 Ir          I1mr I1mr Dr          D1mr  DLmr Dw          D1mw
42          DLmw
43 147,153,617 2,405 2,387 52,520,450 24,800 2,907 21,017,968
44 1,329,245 1,311,259 PROGRAM TOTALS
45 -----
46 Ir          I1mr I1mr Dr          D1mr  DLmr Dw          D1mw
47          DLmw          file:function
48 146,804,064 29 29 52,430,421 20,496 16 20,971,794
49 1,328,643 1,310,723 /root/CARPETA/tp2-2020-2q-src/main.c:
50 matrix_multiply
51 -----
52 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
53 -----
54 Ir          I1mr I1mr Dr          D1mr  DLmr Dw          D1mw
55          DLmw
56 -- line 18 -----
57 . . . . .
58 . . . . . matrix_t* create_matrix(size_t rows,
59 . . . . . size_t cols);
60 . . . . .
61 . . . . . void destroy_matrix(matrix_t* m);
62 . . . . .
63 . . . . .
64 . . . . . int print_matrix(FILE* fp, matrix_t*
65 . . . . . m);
66 . . . . .
67 . . . . .

```

```

63      .      .      .      .      int      .      .      .      .
64      .      .      .      .      main(int argc, char** argv)
65      10      2      2      0      0      0      5
66      0      0      0      {
67      .      .      .      .      /* matrixes */
68      .      .      .      .      matrix_t *a, *b, *c;
69      .      .      .      .      /* n (dimension) and block size */
70      .      .      .      .      size_t n, bs;
71      .      .      .      .      /* line buffer (init to null to
72      .      .      .      .      simplify freeing on error) */
73      1      0      0      0      0      0      1
74      0      0      0      char *line = NULL;
75      .      .      .      .      /* line parsing auxiliar pointers
76      .      .      .      .      */
77      .      .      .      .      char *nptr, *endptr;
78      .      .      .      .      /* auxiliar variables */
79      .      .      .      .      long l;
80      .      .      .      .      double e;
81      2      1      1      0      0      0      1
82      0      0      0      size_t lineno = 1;
83      .      .      .      .      struct timespec t0;
84      .      .      .      .      struct timespec t1;
85      .      .      .      .      double dt;
86      .      .      .      .      size_t i;
87      .      .      .      .
88      25      4      4      9      0      0      1
89      0      0      0      for(; !feof(stdin); lineno++) {
90      10      1      1      4      0      0      6
91      0      0      0      a=b=c=NULL;
92      .      .      .      .
93      .      .      .      .
94      18      3      3      8      1      0      2
95      0      0      0      line = read_line(stdin);
96      .      .      .      .
97      .      .      .      .
98      6      0      0      2      0      0      0
99      0      0      0      if (!line) goto _exit_main;
100     9      0      0      4      0      0      0

```

```

89         0      0      if (line[0] == 0) break;
90         .      .      .      .      .
91         .      .      .      .      .
92         2      1      1      1      0      0      1
93         10     1      1      0      3      1      0      1
94         0      1      1      0      1 = strtol(nptr, &endptr, 10);
95         8      1      1      0      3      0      0      0
96         0      0      0      if (errno) {
97         .      .      .      .      .
98         .      .      .      perror("");
99         .      .      .      .      .
100        .      .      .      goto _exit_main;
101        .      .      .      .      .
102        4      2      2      .      }
103        0      0      0      2      0      0      0
104        .      .      .      if (nptr == endptr) {
105        .      .      .      .      .
106        .      .      .      fprintf(stderr, "missing
107        dimension");
108        .      .      .      .      .
109        .      .      .      goto _exit_main;
110        .      .      .      .      .
111        3      2      2      .      }
112        0      0      0      1      0      0      0
113        .      .      .      if (l < 1) {
114        .      .      .      .      .
115        .      .      .      fprintf(stderr, "invalid
116        dimension");
117        .      .      .      .      .
118        .      .      .      goto _exit_main;
119        .      .      .      .      .
120        .      .      .      }
121        2      1      1      .      1      0      0      1
122        0      0      0      n = (size_t) l;
123        .      .      .      .      .
124        .      .      .      #if 0
125        .      .      .      .      .
126        .      .      .      /* parse block size */
127        .      .      .      .      .
128        .      .      .      nptr = endptr;
129        .      .      .      .      .
130        .      .      .      l = strtol(nptr, &endptr, 10);
131        .      .      .      .      .
132        .      .      .      if (errno) {
133        .      .      .      .      .
134        .      .      .      perror("");
135        .      .      .      .      .
136        .      .      .      goto _exit_main;
137        .      .      .      .      .
138        .      .      .      }
139        -- line 75 -----

```

```

115  -- line 83
116      . . . . .
117      . . . . . }
118      . . . . . bs = (size_t) 1;
119      . . . . .
120      . . . . . if (n % bs) {
121      . . . . .     fprintf(stderr, "block size
doesn't match");
122      . . . . .     goto _exit_main;
123      . . . . . }
124      . . . . . #else
125      2 0 0 1 0 0 1
126      0 0 0 bs = n;
127      . . . . . #endif
128      . . . . .
129      . . . . . /* load matrix a */
130      11 1 1 5 1 0 1
131      0 0 0 if (!(a = create_matrix(n, n)))
132      . . . . . goto _exit_main;
133      . . . . .
134      170 3 3 67 0 0 17
135      0 0 0 for (i=0; i < n*n; i++) {
136      32 0 0 16 0 0 16
137      0 0 0 nptr = endptr;
138      144 1 1 48 0 0 16
139      0 0 0 e = strtod(nptr, &endptr);
140      128 1 1 48 0 0 0
141      0 0 0 if (errno) {
142      . . . . . perror("");
143      . . . . . goto _exit_main;
144      . . . . . }
145      64 2 2 32 0 0 0
146      0 0 0 if (nptr == endptr) {
147      . . . . . fprintf(stderr, "missing A
matrix element");
148      . . . . . goto _exit_main;
149      . . . . .
150      . . . . .

```

```

142      112      1      1      .      }      64      0      0      16
      3      3      a->array[i] = e;
143      .      .      .      .      .      .      .      .
144      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
145      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
146      11      1      1      .      /* load matrix b */
      0      0      if (!(b = create_matrix(n, n)))
147      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
148      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
149      170      2      2      .      67      0      0      17
      0      0      for (i=0; i < n*n; i++) {
150      32      1      1      .      16      0      0      16
      0      0      nptr = endptr;
151      144      1      1      .      48      0      0      16
      0      0      e = strtod(nptr, &endptr);
152      128      1      1      .      48      0      0      0
      0      0      if (errno) {
153      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
154      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
155      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
156      64      1      1      .      32      0      0      0
      0      0      if (nptr == endptr) {
157      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
158      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
159      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
160      112      1      1      .      64      0      0      16
      4      4      b->array[i] = e;
161      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
162      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
163      8      1      1      .      2      0      0      0
      0      0      clock_gettime(CLOCK_REALTIME, &t0
      );
164      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
166      13      2      2      .      /* multiply matrixes */
      0      0      6      1      1      1
      )))      if (!(c = matrix_multiply(a, b, bs
      )))

```



```

167      .      .      .      .      goto _exit_main;      .
168      .      .      .      .      .      .      .
169      8      1      1      .      2      0      0      0
      0      0      0      clock_gettime(CLOCK_REALTIME, &t1
      );
170      .      .      .      .      .      .      .
171      7      1      1      .      2      1      1      1
      1      1      1      dt = (float) (t1.tv_sec - t0.
      tv_sec);
172      12     1      1      5      2      2      1
      0      0      0      dt = dt + ((float)(t1.tv_nsec - t0
      .tv_nsec)) / 1.0e9;
173      .      .      .      .      .      .      .
174      13     3      2      .      5      2      2      0
      0      0      0      if(print_matrix(stdout, c) == -1)
175      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
176      .      .      .      .      .      .      .
177      12     1      1      .      7      0      0      0
      0      0      0      fprintf(stderr, "time: %g\n", dt);
178      .      .      .      .      .      .      .
179      6      1      1      .      3      0      0      0
      0      0      0      free(line);
180      6      1      1      .      3      0      0      0
      0      0      0      destroy_matrix(a);
181      6      1      1      .      3      0      0      0
      0      0      0      destroy_matrix(b);
182      6      0      0      .      3      0      0      0
      0      0      0      destroy_matrix(c);
183      .      .      .      .      .      .      .
      .      .      .      }
184      .      .      .      .      .      .      .
185      3      1      1      .      0      0      0      0
      0      0      0      return 0;
186      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .
      .      .      .      _exit_main:
188      .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, " at line %u\n", (
      unsigned) lineno);
189      .      .      .      .      .      .      .
      .      .      .      free(line);
190      .      .      .      .      .      .      .
      .      .      .      destroy_matrix(a);
191      .      .      .      .      .      .      .
      .      .      .      destroy_matrix(b);

```

```

192      .      .      .      .      .      .      .      .      .      .
193      .      .      .      .      .      .      .      .      .      .
194      6      1      1      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .      .      .      .
196      .      .      .      .      .      .      .      .      .      .
197      .      .      .      .      .      .      .      .      .      .
198      11      2      2      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .      .      .      .
202      .      .      .      .      .      .      .      .      .      .
203      3      1      1      .      .      .      .      .      .      .
204      .      .      .      .      .      .      .      .      .      .
205      11      1      1      .      .      .      .      .      .      .
206      .      .      .      .      .      .      .      .      .      .
207      9      2      2      .      .      .      .      .      .      .
208      .      .      .      .      .      .      .      .      .      .
209      40      2      2      .      .      .      .      .      .      .
210      160      1      1      .      .      .      .      .      .      .
211      176      1      1      .      .      .      .      .      .      .
212      .      .      .      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .      .      .      .
214      .      .      .      .      .      .      .      .      .      .
215      .      .      .      .      .      .      .      .      .      .
216      2      0      0      .      .      .      .      .      .      .
217      9      1      1      .      .      .      .      .      .      .

```

```

218 83,886,088    2    2 31,457,282 20,480    0 10,485,761
      0          0      for (j = 0; j < dim; ++j)
219 62,914,560    0    0 20,971,520    0    0 10,485,760
      1,328,640 1,310,720      v[j] = -1;
220      6    1    1      3    2    2      0
      0          0      free(v);
221      .    .    .      .    .    .      .
      .    .    .      }
222      .    .    .      .    .    .      .
      .    .    .      #endif
223      .    .    .      .    .    .      .
      .    .    .      .
224      17    2    2      6    0    0      2
      0          0      for(kk=0; kk<en; kk+=bs)
225      17    2    2      6    0    0      2
      0          0      for(jj=0; jj<en; jj+=bs)
226      40    1    1      14    0    0      5
      0          0      for(i=0; i<n; i++)
227      204    2    2      80    0    0      20
      0          0      for(j=jj; j<jj+bs; j++) {
228      176    1    1      96    5    5      16
      0          0      sum = mr->array[i*n+j];
229      816    3    3      320    0    0      80
      0          0      for(k=kk; k<kk+bs; k++)
230      1,536    2    2      832    9    9      64
      0          0      sum += m1->array[i*n+k] * m2
      ->array[k*n+j];
231      176    1    1      96    0    0      16
      0          0      mr->array[i*n+j] = sum;
232      .    .    .      .    .    .      .
      .    .    .      }
233      1    0    0      1    0    0      0
      0          0      return mr;
234      6    1    1      2    0    0      0
      0          0 }
235      .    .    .      .    .    .      .
      .    .    .      .
236      .    .    .      .    .    .      .
      .    .    .      char*
237      .    .    .      .    .    .      .
      .    .    .      read_line(FILE *fp)
238      18    3    3      0    0    0      8
      0          0 {
239      .    .    .      .    .    .      .
      .    .    .      #define DEF_LINE_SZ 1024
240      .    .    .      .    .    .      .
      .    .    .      .
241      .    .    .      .    .    .      .
      .    .    .      int c;
242      6    0    0      0    0    0      4
      0          0      size_t len = 0, tam = DEF_LINE_SZ;
243      .    .    .      .    .    .      .
      .    .    .      char* str;
244      .    .    .      .    .    .      .

```

```

245         14      1      1      .      6      1      0      2
246         1      0      str = malloc(tam);
247         6      0      0      2      0      0      0
248         0      0      if (!str) {
249         .      .      .      .      .      .      .
250         .      .      .      .      .      .      .
251         .      .      .      .      .      .      .
252         .      .      .      .      .      .      .
253         .      .      .      .      .      .      .
254         .      .      .      .      .      .      .
255         .      .      .      .      .      .      .
256         .      .      .      .      .      .      .
257         .      .      .      .      .      .      .
258         .      .      .      .      .      .      .
259         .      .      .      .      .      .      .
260         .      .      .      .      .      .      .
261         .      .      .      .      .      .      .
262         .      .      .      .      .      .      .
263         .      .      .      .      .      .      .
264         .      .      .      .      .      .      .
265         .      .      .      .      .      .      .
266         .      .      .      .      .      .      .
267         .      .      .      .      .      .      .
268         .      .      .      .      .      .      .
269         .      .      .      .      .      .      .
270         .      .      .      .      .      .      .

```

1,001 5 5 334 1 0 67
0 0 while (EOF != (c=fgetc(fp)) && c != '\n') {
520 2 2 195 0 0 130
2 2 str[len++] = c;
325 0 0 130 0 0 0
0 0 if (len == tam - 1) {
. . . str = realloc(str, tam *= 2);
. . . if (!str) {
. . . perror("");
. . . return NULL;
. . . }
. . . }
. . . }
. . . }
8 1 1 2 0 0 0
0 0 if (c != EOF)
7 0 0 2 0 0 2
0 0 str[len++] = '\n';
12 1 1 4 0 0 4
0 0 str[len++] = '\0';
2 0 0 2 0 0 0
0 0 return str;
12 1 1 4 0 0 0
0 0 }
matrix_t*

[illegible]

```

297         0      0      0      free(m);
18      0      0      6      0      0      0
298         0      0      }
299         .      .      .      .      .      .      .
300         .      .      .      .      .      .      .
301         .      .      .      .      .      .      .
10      2      2      0      0      0      0      5
302         0      0      {
303         .      .      .      .      .      .      .
304         .      .      .      .      .      .      .
305         3      1      1      2      0      0      1
13      0      0      0      n = m->rows;
306         0      1      1      6      1      1      0
0      0      0      if (fprintf(fp, "%lu", (unsigned
long) m->rows) < 0) {
307         .      .      .      .      .      .      .
308         .      .      .      .      .      .      .
309         .      .      .      .      .      .      .
40      2      2      14      0      0      0      5
0      0      0      for(i=0; i<n; i++)
310      160     2      2      56      0      0      0      20
0      0      0      for (j=0; j<n; j++)
311      352     3      3      160     0      0      0      0
0      0      0      if (fprintf(fp, " %g", m->array[i
*n+j]) < 0) {
312         .      .      .      .      .      .      .
313         .      .      .      .      .      .      .
314         .      .      .      .      .      .      .
10      1      1      4      0      0      0      0
0      0      0      if (fprintf(fp, "\n") < 0) {
315         .      .      .      .      .      .      .
316         .      .      .      .      .      .      .
317         .      .      .      .      .      .      .
318         .      .      .      .      .      .      .
1      1      1      0      0      0      0      0
0      0      0      return 0;
319      6      1      1      2      0      0      0      0
0      0      0      }
320
321
322

```

```

323 Ir          I1mr I1mr Dr          D1mr  DLmr Dw          D1mw
324      DLmw
-----
325 146,808,382 120 119 52,432,069 20,509 23 20,972,236
      1,328,655 1,310,733 events annotated

1 5 125 118 38 87 87 115 168 73 98 118 86 89 43 46 93 156 132 47
   79 148 79 110 62 57 137
2 017, and GNU GPL'd, by Nicholas Nethercote et al.
3 ==614== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4 ==614== Command: /tmp/02-mmult
5 ==614== Parent PID: 597
6 ==614==
7 --614-- Warning: Cannot auto-detect cache config, using
   defaults.
8 --614-- Run with -v to see.
9 ==614==
10 ==614== I refs: 147,187,039
11 ==614== I1 misses: 2,407
12 ==614== L1i misses: 2,389
13 ==614== I1 miss rate: 0.00%
14 ==614== L1i miss rate: 0.00%
15 ==614==
16 ==614== D refs: 73,550,897 (52,528,979 rd +
   21,021,918 wr)
17 ==614== D1 misses: 1,354,198 ( 24,881 rd +
   1,329,317 wr)
18 ==614== L1d misses: 1,314,183 ( 2,915 rd +
   1,311,268 wr)
19 ==614== D1 miss rate: 1.8% ( 0.0% +
   6.3% )
20 ==614== L1d miss rate: 1.8% ( 0.0% +
   6.2% )
21 ==614==
22 ==614== LL refs: 1,356,605 ( 27,288 rd +
   1,329,317 wr)
23 ==614== LL misses: 1,316,572 ( 5,304 rd +
   1,311,268 wr)
24 ==614== LL miss rate: 0.6% ( 0.0% +
   6.2% )
25 -----
26 I1 cache: 32768 B, 32 B, 4-way associative
27 D1 cache: 32768 B, 32 B, direct-mapped
28 LL cache: 524288 B, 32 B, 8-way associative
29 Command: /tmp/02-mmult
30 Data file: cachegrind.out.614
31 Events recorded: Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
32 Events shown: Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
33 Event sort order: Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
34 Thresholds: 0.1 100 100 100 100 100 100 100 100
35 Include dirs:

```

```

36 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
37 Auto-annotation: off
38
39 -----
40 Ir          I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
41          DLmw
42 -----
43 147,187,039 2,407 2,389 52,528,979 24,881 2,915 21,021,918
44          1,329,317 1,311,268 PROGRAM TOTALS
45 -----
46 Ir          I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
47          DLmw          file:function
48 -----
49 146,806,731 29 29 52,431,722 20,503 23 20,971,983
50          1,328,645 1,310,725 /root/CARPETA/tp2-2020-2q-src/main.c:
51          matrix_multiply
52 -----
53 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
54 -----
55 Ir          I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
56          DLmw
57 -----
58 -- line 18 -----
59 . . . . .
60 . . . matrix_t* create_matrix(size_t rows,
61 . . . size_t cols);
62 . . . . .
63 . . . void destroy_matrix(matrix_t* m);
64 . . . . .
65 . . . int print_matrix(FILE* fp, matrix_t*
66 . . . m);
67 . . . . .
68 . . . int
69 . . . . .
70 . . . main(int argc, char** argv)
71 . . . 10 2 2 0 0 0 5
72 . . . 0 {
73 . . . . .
74 . . . /* matrixes */
75 . . . . .

```



```

66         .      .      .      .      matrix_t *a, *b, *c;
67         .      .      .      .      /* n (dimension) and block size */
68         .      .      .      .      size_t n, bs;
69         .      .      .      .      /* line buffer (init to null to
simplify freeing on error) */
70         1      0      0      0      0      0      0      1
0      0      0      0      char *line = NULL;
71         .      .      .      .      /* line parsing auxiliar pointers
*/
72         .      .      .      .      char *nptr, *endptr;
73         .      .      .      .      /* auxiliar variables */
74         .      .      .      .      long l;
75         .      .      .      .      double e;
76         2      1      1      0      0      0      0      1
0      0      0      0      size_t lineno = 1;
77         .      .      .      .      struct timespec t0;
78         .      .      .      .      struct timespec t1;
79         .      .      .      .      double dt;
80         .      .      .      .      size_t i;
81         25     4      4      9      0      0      0      1
0      0      0      0      for(; !feof(stdin); lineno++) {
82         10     1      1      4      0      0      0      6
0      0      0      0      a=b=c=NULL;
83         .      .      .      .      .      .      .      .
84         18     3      3      8      1      0      0      2
0      0      0      0      line = read_line(stdin);
85         .      .      .      .      .      .      .      .
86         6      0      0      2      0      0      0      0
0      0      0      0      if (!line) goto _exit_main;
87         9      0      0      4      0      0      0      0
0      0      0      0      if (line[0] == 0) break;
88         .      .      .      .      .      .      .      .
89         .      .      .      .      .      .      .      .
90         2      1      1      1      0      0      0      1
1      1      0      0      nptr = line;
91         10     1      1      3      1      0      0      1

```

```

92      0      0      1 = strtol(nptr, &endptr, 10);
      8      1      1      3      0      0      0
93      0      0      if (errno) {
      .      .      .      .      .      .
94      .      .      .      .      .      .
      .      .      .      goto _exit_main;
95      .      .      .      .      .      .
      .      .      .      }
96      4      2      2      2      0      0      0
      0      0      if (nptr == endptr) {
97      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "missing
      dimension");
98      .      .      .      .      .      .
      .      .      .      goto _exit_main;
99      .      .      .      .      .      .
      .      .      .      }
100     3      2      2      1      0      0      0
      0      0      if (l < 1) {
101     .      .      .      .      .      .
      .      .      .      fprintf(stderr, "invalid
      dimension");
102     .      .      .      .      .      .
      .      .      .      goto _exit_main;
103     .      .      .      .      .      .
      .      .      .      }
104     2      1      1      1      0      0      1
      0      0      n = (size_t) 1;
105     .      .      .      .      .      .
      .      .      .      #if 0
106     .      .      .      .      .      .
      .      .      .      /* parse block size */
107     .      .      .      .      .      .
      .      .      .      nptr = endptr;
108     .      .      .      .      .      .
      .      .      .      l = strtol(nptr, &endptr, 10);
109     .      .      .      .      .      .
      .      .      .      if (errno) {
110     .      .      .      .      .      .
      .      .      .      perror("");
111     .      .      .      .      .      .
      .      .      .      goto _exit_main;
112     .      .      .      .      .      .
      .      .      .      }
113 -- line 75 -----
114 -- line 83 -----
115     .      .      .      .      .      .
      .      .      .      }
116     .      .      .      .      .      .
      .      .      .      bs = (size_t) 1;
117     .      .      .      .      .      .
      .      .      .      .
118     .      .      .      .      .      .

```

```

119         .         .         if (n % bs) {
120         .         .         .         fprintf(stderr, "block size
doesn't match");
121         .         .         .         goto _exit_main;
122         .         .         .         }
123         .         .         .         #else
2         0         0         1         0         0         1
124         .         .         .         bs = n;
125         .         .         .         #endif
126         .         .         .
127         .         .         .         /* load matrix a */
11         1         1         5         1         0         1
128         .         .         .         if (!(a = create_matrix(n, n)))
0         0         .         .         .         goto _exit_main;
129         .         .         .
130         .         .         .         260         3         3         103         0         0         26
0         0         0         for (i=0; i < n*n; i++) {
131         .         .         .         50         0         0         25         0         0         25
0         0         0         nptr = endptr;
132         .         .         .         225         1         1         75         0         0         25
0         0         0         e = strtod(nptr, &endptr);
133         .         .         .         200         1         1         75         0         0         0
0         0         0         if (errno) {
134         .         .         .         .         .         .         .         .
135         .         .         .         .         .         .         .         .
136         .         .         .         .         .         .         .         .
137         .         .         .         .         .         .         .         .
100         2         2         50         0         0         0
138         .         .         .         .         .         .         .         .
0         0         .         .         .         .         .         .
139         .         .         .         .         .         .         .         .
matrix element");
140         .         .         .         .         .         .         .         .
141         .         .         .         .         .         .         .         .
175         1         1         100         0         0         25
5         5         .         .         .         .         .         .
142         .         .         .         .         .         .         .         .
143         .         .         .         .         .         .         .         .
144         .         .         .         .         .         .         .         .

```

```

145      11      1      1      .      /* load matrix b */
146      0      0      .      if (!(b = create_matrix(n, n)))
147      .      .      .      goto _exit_main;
148      260     2      2      .      103      0      0      26
149      0      1      1      0      for (i=0; i < n*n; i++) {
150      50      1      1      0      25      0      0      25
151      0      0      .      nptr = endptr;
152      225     1      1      0      75      0      0      25
153      0      0      .      e = strtod(nptr, &endptr);
154      200     1      1      0      75      0      0      0
155      0      0      .      if (errno) {
156      .      .      .      .      perror("");
157      .      .      .      .      goto _exit_main;
158      .      .      .      .      }
159      100     1      1      0      50      0      0      0
160      0      0      .      if (nptr == endptr) {
161      .      .      .      .      fprintf(stderr, "missing B
162      matrix element");
163      .      .      .      .      goto _exit_main;
164      .      .      .      .      }
165      175     1      1      0      100     0      0      25
166      5      .      5      .      b->array[i] = e;
167      .      .      .      .      }
168      .      .      .      .      .
169      8      1      1      .      2      0      0      0
170      0      0      .      clock_gettime(CLOCK_REALTIME, &t0
171      );
172      .      .      .      .      .
173      .      .      .      .      .
174      .      .      .      .      /* multiply matrixes */
175      13      2      2      .      6      1      1      1
176      0      0      .      if (!(c = matrix_multiply(a, b, bs
177      )))
178      .      .      .      .      goto _exit_main;
179      .      .      .      .      .
180      8      1      1      .      2      0      0      0
181      0      0      .      clock_gettime(CLOCK_REALTIME, &t1
182      );
183      .      .      .      .      .

```

```

170         7      1      1      .      2      1      1      1
           1      1      dt = (float) (t1.tv_sec - t0.
           tv_sec);
171     12      1      1      5      2      2      1
           0      0      dt = dt + ((float)(t1.tv_nsec - t0
           .tv_nsec)) / 1.0e9;
172     .      .      .      .      .      .      .
           .      .      .
173     13      3      2      5      2      2      0
           0      0      if(print_matrix(stdout, c) == -1)
174     .      .      .      .      .      .      .
           .      .      .      goto _exit_main;
175     .      .      .      .      .      .      .
           .      .      .
176     12      1      1      7      0      0      0
           0      0      fprintf(stderr, "time: %g\n", dt);
177     .      .      .      .      .      .      .
           .      .      .
178     6      1      1      3      0      0      0
           0      0      free(line);
179     6      1      1      3      0      0      0
           0      0      destroy_matrix(a);
180     6      1      1      3      0      0      0
           0      0      destroy_matrix(b);
181     6      0      0      3      0      0      0
           0      0      destroy_matrix(c);
182     .      .      .      .      .      .      .
           .      .      .      }
183     .      .      .      .      .      .      .
           .      .      .
184     3      1      1      0      0      0      0
           0      0      return 0;
185     .      .      .      .      .      .      .
           .      .      .
186     .      .      .      .      .      .      .
           .      .      .      _exit_main:
187     .      .      .      .      .      .      .
           .      .      .      fprintf(stderr, " at line %u\n", (
           unsigned) lineno);
188     .      .      .      .      .      .      .
           .      .      .      free(line);
189     .      .      .      .      .      .      .
           .      .      .      destroy_matrix(a);
190     .      .      .      .      .      .      .
           .      .      .      destroy_matrix(b);
191     .      .      .      .      .      .      .
           .      .      .      destroy_matrix(c);
192     .      .      .      .      .      .      .
           .      .      .      exit(1);
193     6      1      1      2      0      0      0
           0      0      0 }
194     .      .      .      .      .      .      .
           .      .      .

```

```

195      .      .      .      .      .      .      .
196      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
197      11      2      2      0      0      0      6
      0      .      0      {
198      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
202      3      1      1      2      0      0      1
      0      .      0      n = m1->rows;
203      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
204      11      1      1      5      0      0      1
      0      .      0      if(!(mr = create_matrix(n,n)))
      return NULL;
205      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
206      9      2      2      3      0      0      1
      0      .      0      en = bs*(n/bs);
207      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
208      48      2      2      17      0      0      6
      0      .      0      for(i=0; i<n; i++)
209      240      1      1      85      0      0      30
      0      .      0      for(j=0; j<n; j++)
210      275      1      1      125      0      0      50
      5      .      5      mr->array[i*n+j] = 0.0;
211      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
212      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
214      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
215      2      0      0      0      0      0      1
      0      .      0      size_t dim = 1024*1024*10;
216      9      1      1      3      0      0      1
      0      .      0      int *v = malloc(dim*sizeof(int));
217      83,886,088      2      2      31,457,282      20,480      0      10,485,761
      0      .      0      for (j = 0; j < dim; ++j)
218      62,914,560      0      0      20,971,520      0      0      10,485,760
      1,328,640      1,310,720      v[j] = -1;
219      6      1      1      3      2      2      0
      0      .      0      free(v);
220      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .

```

```

221         . . . . #endif . . .
222         . . . . . . .
223         17 2 2 6 0 0 2
           0 0 for(kk=0; kk<en; kk+=bs)
224         17 2 2 6 0 0 2
           0 0 for(jj=0; jj<en; jj+=bs)
225         48 1 1 17 0 0 6
           0 0 for(i=0; i<n; i++)
226         305 2 2 120 0 0 30
           0 0 for(j=jj; j<jj+bs; j++) {
227         275 1 1 150 7 7 25
           0 0 sum = mr->array[i*n+j];
228         1,525 3 3 600 0 0 150
           0 0 for(k=kk; k<kk+bs; k++)
229         3,000 2 2 1,625 14 14 125
           0 0 sum += m1->array[i*n+k] * m2
           ->array[k*n+j];
230         275 1 1 150 0 0 25
           0 0 mr->array[i*n+j] = sum;
231         . . . . }
232         1 0 0 1 0 0 0
           0 0 return mr;
233         6 1 1 2 0 0 0
           0 0 }
234         . . . .
235         . . . .
236         . . . .
237         18 3 3 0 0 0 8
           0 0 {
238         . . . .
239         . . . . #define DEF_LINE_SZ 1024
240         . . . .
241         . . . . int c;
242         6 0 0 0 0 0 4
           0 0 size_t len = 0, tam = DEF_LINE_SZ;
243         . . . . char* str;
244         14 1 1 6 1 0 2
           1 0 str = malloc(tam);
245         6 0 0 2 0 0 0
           0 0 if (!str) {
246         . . . .
247         . . . . perror("");

```

248	return NULL;	.	.
249	.	.	.	}	.	.	.
250	1,541	5	5	514	1	0	103
	0		0	while (EOF != (c=fgetc(fp)) && c != '\n			
) {						
251	808	2	2	303	0	0	202
	3		3	str[len++]=c;			
252	505	0	0	202	0	0	0
	0		0	if (len==tam-1) {			
253
	.	.	.	str = realloc(str, tam *= 2);			
254
	.	.	.	if (!str) {			
255
	.	.	.	perror("");			
256
	.	.	.	return NULL;			
257
	.	.	.	}			
258
	.	.	.	}			
259
	.	.	.	}			
260

261	8	1	1	2	0	0	0
	0		0	if (c != EOF)			
262	7	0	0	2	0	0	2
	0		0	str[len++]='\n';			
263

264	12	1	1	4	0	0	4
	0		0	str[len++]='\0';			
265	2	0	0	2	0	0	0
	0		0	return str;			
266	12	1	1	4	0	0	0
	0		0	}			
267

268

269
	.	.	.	matrix_t*			
270
	.	.	.	create_matrix(size_t rows, size_t			
	cols)						
271	30	1	1	0	0	0	15
	0		0	{			
272
	.	.	.	matrix_t * m;			
273


```

274      30      2      2      .      9      1      0      3
          0      0      if (!(m = malloc(sizeof(matrix_t))))
          {
275      .      .      .      .      .      .      .      .
          .      .      .      perror("");
276      .      .      .      .      .      .      .      .
          .      .      .      return NULL;
277      .      .      .      .      .      .      .      .
          .      .      .      }
278      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
279      9      1      1      0      6      0      0      3
          0      0      m->rows = rows;
280      9      1      1      0      6      0      0      3
          0      0      m->cols = cols;
281      51      2      2      21      0      0      3
          0      0      if (!(m->array = malloc(sizeof(
double)*rows*cols))) {
282      .      .      .      .      .      .      .      .
          .      .      .      free(m);
283      .      .      .      .      .      .      .      .
          .      .      .      perror("");
284      .      .      .      .      .      .      .      .
          .      .      .      return NULL;
285      .      .      .      .      .      .      .      .
          .      .      .      }
286      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
287      3      1      1      0      3      0      0      0
          0      0      return m;
288      18      1      1      0      6      0      0      0
          0      0      0 }
289      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
290      .      .      .      .      .      .      .      .
          .      .      .      void
291      .      .      .      .      .      .      .      .
          .      .      .      destroy_matrix(matrix_t* m)
292      27      1      1      0      0      0      0      12
          0      0      0 {
293      9      0      0      0      3      0      0      0
          0      0      if (!m) return;
294      24      1      1      0      12      0      0      0
          0      0      free(m->array);
295      24      1      1      0      9      0      0      0
          0      0      free(m);
296      18      0      0      0      6      0      0      0
          0      0      0 }
297      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .      .
          .      .      .      int
299      .      .      .      .      .      .      .      .

```

```

300      .      .      .      print_matrix(FILE* fp, matrix_t* m)
10      2      2      0      0      0      5
      0      0      {
301      .      .      .      .      .      .      .
      .      .      .      size_t i, j;
302      .      .      .      .      .      .      .
      .      .      .      size_t n;
303      3      1      1      2      0      0      1
      0      0      0      n = m->rows;
304      13      1      1      6      1      1      0
      0      0      if (fprintf(fp, "%lu", (unsigned
long) m->rows) < 0) {
305      .      .      .      .      .      .      .
      .      .      .      perror("");
306      .      .      .      .      .      .      .
      .      .      .      return -1;
307      .      .      .      .      .      .      .
      .      .      .      }
308      48      2      2      17      0      0      6
      0      0      0      for(i=0; i<n; i++)
309      240      2      2      85      0      0      30
      0      0      0      for (j=0; j<n; j++)
310      550      3      3      250      47      0      0
      0      0      if (fprintf(fp, " %g", m->array[i
*n+j]) < 0) {
311      .      .      .      .      .      .      .
      .      .      .      perror("");
312      .      .      .      .      .      .      .
      .      .      .      return -1;
313      .      .      .      .      .      .      .
      .      .      .      }
314      10      1      1      4      0      0      0
      0      0      0      if (fprintf(fp, "\n") < 0) {
315      .      .      .      .      .      .      .
      .      .      .      perror("");
316      .      .      .      .      .      .      .
      .      .      .      return -1;
317      .      .      .      .      .      .      .
      .      .      .      }
318      1      1      1      0      0      0      0
      0      0      0      return 0;
319      6      1      1      2      0      0      0
      0      0      0      }
320
321 -----
322 Ir      I1mr ILmr Dr      D1mr      DLmr Dw      D1mw
      DLmw
323 -----
324 146,813,063 120 119 52,434,158 20,563 30 20,972,616
      1,328,661 1,310,739 events annotated

```

```

1  6 68 18 73 48 69 59 126 52 95 128 111 73 82 68 104 68 105 100
   134 76 81 150 129 82 150 77 118 115 171 110 145 53 74 146
   123 104
2  e et al.
3  ==617== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4  ==617== Command: /tmp/02-mmult
5  ==617== Parent PID: 597
6  ==617==
7  --617-- Warning: Cannot auto-detect cache config, using
   defaults.
8  --617--          Run with -v to see.
9  ==617==
10 ==617== I   refs:          147,225,930
11 ==617== I1  misses:          2,407
12 ==617== LLi misses:          2,389
13 ==617== I1  miss rate:          0.00%
14 ==617== LLi miss rate:          0.00%
15 ==617==
16 ==617== D   refs:          73,565,860 (52,539,342 rd  +
   21,026,518 wr)
17 ==617== D1  misses:          1,354,213 (   24,897 rd  +
   1,329,316 wr)
18 ==617== LLd misses:          1,314,203 (   2,925 rd  +
   1,311,278 wr)
19 ==617== D1  miss rate:          1.8% (   0.0%  +
   6.3%  )
20 ==617== LLd miss rate:          1.8% (   0.0%  +
   6.2%  )
21 ==617==
22 ==617== LL refs:          1,356,620 (   27,304 rd  +
   1,329,316 wr)
23 ==617== LL  misses:          1,316,592 (    5,314 rd  +
   1,311,278 wr)
24 ==617== LL  miss rate:          0.6% (   0.0%  +
   6.2%  )
25 -----
26 I1 cache:          32768 B, 32 B, 4-way associative
27 D1 cache:          32768 B, 32 B, direct-mapped
28 LL cache:          524288 B, 32 B, 8-way associative
29 Command:          /tmp/02-mmult
30 Data file:          cachegrind.out.617
31 Events recorded:   Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
32 Events shown:      Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Event sort order:  Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Thresholds:        0.1 100 100 100 100 100 100 100 100
35 Include dirs:
36 User annotated:    /root/CARPETA/tp2-2020-2q-src/main.c
37 Auto-annotation:   off
38
39 -----
40 Ir           I1mr  I1Lmr  Dr           D1mr  DLmr  Dw           D1mw

```

41	DLmw	
42	147,225,930 2,407 2,389 52,539,342 24,897 2,925 21,026,518	
43	1,329,316 1,311,278 PROGRAM TOTALS	
44		
45	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
46	DLmw file:function	
47	146,810,542 29 29 52,433,589 20,511 31 20,972,246	
48	1,328,648 1,310,728 /root/CARPETA/tp2-2020-2q-src/main.c:	
49	matrix_multiply	
50	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c	
51		
52	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
53	DLmw	
54	-- line 18 -----	
55	
56 matrix_t* create_matrix(size_t rows,	
57 size_t cols);	
58 void destroy_matrix(matrix_t* m);	
59 int print_matrix(FILE* fp, matrix_t*	
60 m);	
61 int	
62 main(int argc, char** argv)	
63	10 2 2 0 0 0 5	
64	0 0 {	
65 /* matrixes */	
66 matrix_t *a, *b, *c;	
67 /* n (dimension) and block size */	
68 size_t n, bs;	

```

.          .          /* line buffer (init to null to
simplify freeing on error) */
69      1      0      0      0      0      0      1
.          .          char *line = NULL;
70      .          .          .          .          .          .
.          .          /* line parsing auxiliar pointers
*/
71      .          .          .          .          .          .
.          .          char *nptr, *endptr;
72      .          .          .          .          .          .
.          .          /* auxiliar variables */
73      .          .          .          .          .          .
.          .          long l;
74      .          .          .          .          .          .
.          .          double e;
75      2      1      1      0      0      0      1
.          .          size_t lineno = 1;
76      .          .          .          .          .          .
.          .          struct timespec t0;
77      .          .          .          .          .          .
.          .          struct timespec t1;
78      .          .          .          .          .          .
.          .          double dt;
79      .          .          .          .          .          .
.          .          size_t i;
80      .          .          .          .          .          .
.          .          .          .          .          .
81      25     4      4      9      0      0      1
.          .          for(;; !feof(stdin); lineno++) {
82      10     1      1      4      0      0      6
.          .          a=b=c=NULL;
83      .          .          .          .          .          .
.          .          .          .          .          .
84      18     3      3      8      1      0      2
.          .          line = read_line(stdin);
85      .          .          .          .          .          .
.          .          .          .          .          .
86      6      0      0      2      0      0      0
.          .          if (!line) goto _exit_main;
87      9      0      0      4      0      0      0
.          .          if (line[0] == 0) break;
88      .          .          .          .          .          .
.          .          .          .          .          .
89      .          .          .          .          .          .
.          .          /* parse dimension */
90      2      1      1      1      0      0      1
.          .          nptr = line;
91      10     1      1      3      1      0      1
.          .          l = strtol(nptr, &endptr, 10);
92      8      1      1      3      0      0      0
.          .          if (errno) {
93      .          .          .          .          .          .
.          .          perror("");
94      .          .          .          .          .          .

```

```

95         . . . goto _exit_main;
96         . . . }
97         4 2 2 0 2 0 0 0
98         0 0 if (nptr == endptr) {
99         . . . fprintf(stderr, "missing
100         dimension");
101         . . . goto _exit_main;
102         . . . }
103         3 2 2 0 1 0 0 0
104         0 0 if (l < 1) {
105         . . . fprintf(stderr, "invalid
106         dimension");
107         . . . goto _exit_main;
108         . . . }
109         2 1 1 0 1 0 0 1
110         0 0 n = (size_t) l;
111         . . . #if 0
112         . . . /* parse block size */
113         . . . nptr = endptr;
114         . . . l = strtol(nptr, &endptr, 10);
115         . . . if (errno) {
116         . . .     perror("");
117         . . .     goto _exit_main;
118         . . . }
119         -- line 75 -----
120         -- line 83 -----
121         . . . }
122         . . . bs = (size_t) l;
123         . . .
124         . . . if (n % bs) {
125         . . .     fprintf(stderr, "block size
126         doesn't match");
127         . . .     goto _exit_main;

```

```

121      .      .      .      .      }      .      .      .
122      .      .      .      .      .      .      .      .
123      2      0      0      .      #else      1      0      0      1
124      .      .      .      .      .      .      .      .
125      .      .      .      .      #endif      .      .      .
126      .      .      .      .      .      .      .      .
127      11      1      1      .      /* load matrix a */      5      1      0      1
128      .      .      .      .      .      .      .      .
129      .      .      .      .      .      .      .      .
130      370      3      3      .      147      0      0      37
131      0      .      0      .      for (i=0; i < n*n; i++) {
132      72      0      0      .      36      0      0      36
133      0      .      0      .      nptr = endptr;
134      324      1      1      .      108      0      0      36
135      0      .      0      .      e = strtod(nptr, &endptr);
136      288      1      1      .      108      0      0      0
137      0      .      0      .      if (errno) {
138      .      .      .      .      .      .      .      .
139      .      .      .      .      perror("");
140      .      .      .      .      .      .      .      .
141      .      .      .      .      goto _exit_main;
142      .      .      .      .      .      .      .      .
143      .      .      .      .      }
144      144      2      2      .      72      0      0      0
145      0      .      0      .      if (nptr == endptr) {
146      .      .      .      .      .      .      .      .
147      .      .      .      .      fprintf(stderr, "missing A
matrix element");
148      .      .      .      .      .      .      .      .
149      .      .      .      .      goto _exit_main;
150      .      .      .      .      .      .      .      .
151      .      .      .      .      }
152      252      1      1      .      144      0      0      36
153      8      .      8      .      a->array[i] = e;
154      .      .      .      .      .      .      .      .
155      .      .      .      .      }
156      .      .      .      .      .      .      .      .
157      .      .      .      .      .      .      .      .
158      .      .      .      .      /* load matrix b */
159      11      1      1      .      5      0      0      1
160      0      .      0      .      if (!(b = create_matrix(n, n)))
161      .      .      .      .      .      .      .      .
162      .      .      .      .      goto _exit_main;
163      .      .      .      .      .      .      .      .
164      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
166      .      .      .      .      .      .      .      .
167      .      .      .      .      .      .      .      .
168      .      .      .      .      .      .      .      .
169      .      .      .      .      .      .      .      .
170      .      .      .      .      .      .      .      .
171      .      .      .      .      .      .      .      .
172      .      .      .      .      .      .      .      .
173      .      .      .      .      .      .      .      .
174      .      .      .      .      .      .      .      .
175      .      .      .      .      .      .      .      .
176      .      .      .      .      .      .      .      .
177      .      .      .      .      .      .      .      .
178      .      .      .      .      .      .      .      .
179      .      .      .      .      .      .      .      .
180      .      .      .      .      .      .      .      .
181      .      .      .      .      .      .      .      .
182      .      .      .      .      .      .      .      .
183      .      .      .      .      .      .      .      .
184      .      .      .      .      .      .      .      .
185      .      .      .      .      .      .      .      .
186      .      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .      .
192      .      .      .      .      .      .      .      .
193      .      .      .      .      .      .      .      .
194      .      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .      .
196      .      .      .      .      .      .      .      .
197      .      .      .      .      .      .      .      .
198      .      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .      .
202      .      .      .      .      .      .      .      .
203      .      .      .      .      .      .      .      .
204      .      .      .      .      .      .      .      .
205      .      .      .      .      .      .      .      .
206      .      .      .      .      .      .      .      .
207      .      .      .      .      .      .      .      .
208      .      .      .      .      .      .      .      .
209      .      .      .      .      .      .      .      .
210      .      .      .      .      .      .      .      .
211      .      .      .      .      .      .      .      .
212      .      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .      .
214      .      .      .      .      .      .      .      .
215      .      .      .      .      .      .      .      .
216      .      .      .      .      .      .      .      .
217      .      .      .      .      .      .      .      .
218      .      .      .      .      .      .      .      .
219      .      .      .      .      .      .      .      .
220      .      .      .      .      .      .      .      .
221      .      .      .      .      .      .      .      .
222      .      .      .      .      .      .      .      .
223      .      .      .      .      .      .      .      .
224      .      .      .      .      .      .      .      .
225      .      .      .      .      .      .      .      .
226      .      .      .      .      .      .      .      .
227      .      .      .      .      .      .      .      .
228      .      .      .      .      .      .      .      .
229      .      .      .      .      .      .      .      .
230      .      .      .      .      .      .      .      .
231      .      .      .      .      .      .      .      .
232      .      .      .      .      .      .      .      .
233      .      .      .      .      .      .      .      .
234      .      .      .      .      .      .      .      .
235      .      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .      .
238      .      .      .      .      .      .      .      .
239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .      .
242      .      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      .      .      .      .      .      .      .      .
246      .      .      .      .      .      .      .      .
247      .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
251      .      .      .      .      .      .      .      .
252      .      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .
262      .      .      .      .      .      .      .      .
263      .      .      .      .      .      .      .      .
264      .      .      .      .      .      .      .      .
265      .      .      .      .      .      .      .      .
266      .      .      .      .      .      .      .      .
267      .      .      .      .      .      .      .      .
268      .      .      .      .      .      .      .      .
269      .      .      .      .      .      .      .      .
270      .      .      .      .      .      .      .      .
271      .      .      .      .      .      .      .      .
272      .      .      .      .      .      .      .      .
273      .      .      .      .      .      .      .      .
274      .      .      .      .      .      .      .      .
275      .      .      .      .      .      .      .      .
276      .      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .      .
278      .      .      .      .      .      .      .      .
279      .      .      .      .      .      .      .      .
280      .      .      .      .      .      .      .      .
281      .      .      .      .      .      .      .      .
282      .      .      .      .      .      .      .      .
283      .      .      .      .      .      .      .      .
284      .      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .      .
286      .      .      .      .      .      .      .      .
287      .      .      .      .      .      .      .      .
288      .      .      .      .      .      .      .      .
289      .      .      .      .      .      .      .      .
290      .      .      .      .      .      .      .      .
291      .      .      .      .      .      .      .      .
292      .      .      .      .      .      .      .      .
293      .      .      .      .      .      .      .      .
294      .      .      .      .      .      .      .      .
295      .      .      .      .      .      .      .      .
296      .      .      .      .      .      .      .      .
297      .      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .      .
299      .      .      .      .      .      .      .      .
300      .      .      .      .      .      .      .      .
301      .      .      .      .      .      .      .      .
302      .      .      .      .      .      .      .      .
303      .      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .      .
305      .      .      .      .      .      .      .      .
306      .      .      .      .      .      .      .      .
307      .      .      .      .      .      .      .      .
308      .      .      .      .      .      .      .      .
309      .      .      .      .      .      .      .      .
310      .      .      .      .      .      .      .      .
311      .      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .      .
314      .      .      .      .      .      .      .      .
315      .      .      .      .      .      .      .      .
316      .      .      .      .      .      .      .      .
317      .      .      .      .      .      .      .      .
318      .      .      .      .      .      .      .      .
319      .      .      .      .      .      .      .      .
320      .      .
```

```

148      370      2      2      .      147      0      0      37
      0      0      0      for (i=0; i < n*n; i++) {
149      72      1      1      36      0      0      36
      0      0      0      nptr = endptr;
150      324      1      1      108      0      0      36
      0      0      0      e = strtod(nptr, &endptr);
151      288      1      1      108      0      0      0
      0      0      0      if (errno) {
152      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
153      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
154      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
155      144      1      1      72      0      0      0
      0      0      0      if (nptr == endptr) {
156      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
157      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
158      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
159      252      1      1      144      0      0      36
      9      9      9      b->array[i] = e;
160      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
161      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
162      8      1      1      2      0      0      0
      0      0      0      clock_gettime(CLOCK_REALTIME, &t0
      );
163      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
164      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
165      13      2      2      6      1      1      1
      0      0      0      if (!(c = matrix_multiply(a, b, bs
      )))
166      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
167      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
168      8      1      1      2      0      0      0
      0      0      0      clock_gettime(CLOCK_REALTIME, &t1
      );
169      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
170      7      1      1      2      1      1      1
      1      1      1      dt = (float) (t1.tv_sec - t0.
      tv_sec);
171      12      1      1      5      2      2      1
      0      0      0      dt = dt + ((float)(t1.tv_nsec - t0

```



```

172         .tv_nsec)) / 1.0e9;
173     13      3      2      5      2      2      0
174         0      0      if(print_matrix(stdout, c) == -1)
175         .      .      goto _exit_main;
176     12      1      1      7      0      0      0
177         0      0      fprintf(stderr, "time: %g\n", dt);
178     6      1      1      3      0      0      0
179         0      1      1      3      0      0      0
180     6      1      1      3      0      0      0
181         0      0      0      destroy_matrix(a);
182     6      0      0      3      0      0      0
183         0      0      0      destroy_matrix(b);
184         .      .      .      3      0      0      0
185         .      .      .      destroy_matrix(c);
186         .      .      .      }
187     3      1      1      0      0      0      0
188         0      0      return 0;
189         .      .      .
190         .      .      .      _exit_main:
191         .      .      .      fprintf(stderr, " at line %u\n", (
192         unsigned) lineno);
193         .      .      .      free(line);
194         .      .      .      destroy_matrix(a);
195         .      .      .      destroy_matrix(b);
196         .      .      .      destroy_matrix(c);
197         .      .      .      exit(1);
198     6      1      1      2      0      0      0
199         0      0      }
200         .      .      .
201         .      .      .      matrix_t*
202         .      .      .      matrix_multiply(matrix_t* m1,
203         matrix_t* m2, int bs)
204     11      2      2      0      0      0      6

```

```

198      .      .      .      .      .      .      .      .
199      .      .      .      .      size_t n, en, i, j, k, kk, jj;
200      .      .      .      .      double sum;
201      .      .      .      .      matrix_t* mr;
202      .      .      .      .      .      .      .      .
203      3      1      1      .      2      0      0      1
204      0      0      0      n = m1->rows;
205      .      .      .      .      .      .      .      .
206      11     1      1      .      5      0      0      1
207      0      0      0      if(!(mr = create_matrix(n,n)))
208      return NULL;
209      .      .      .      .      .      .      .      .
210      9      2      2      .      3      0      0      1
211      0      0      0      en = bs*(n/bs);
212      .      .      .      .      .      .      .      .
213      56     2      2      .      20     0      0      7
214      0      0      0      for(i=0; i<n; i++)
215      336     1      1      120     0      0      42
216      0      0      0      for(j=0; j<n; j++)
217      396     1      1      180     0      0      72
218      8      8      mr->array[i*n+j] = 0.0;
219      .      .      .      .      .      .      .      .
220      .      .      .      .      .      .      .      .
221      .      .      .      .      .      .      .      .
222      .      .      .      .      .      .      .      .
223      .      .      .      .      .      .      .      .
224      .      .      .      .      .      .      .      .
225      .      .      .      .      .      .      .      .
226      .      .      .      .      .      .      .      .
227      .      .      .      .      .      .      .      .
228      .      .      .      .      .      .      .      .
229      .      .      .      .      .      .      .      .
230      .      .      .      .      .      .      .      .
231      .      .      .      .      .      .      .      .
232      .      .      .      .      .      .      .      .
233      .      .      .      .      .      .      .      .
234      .      .      .      .      .      .      .      .
235      .      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .      .
238      .      .      .      .      .      .      .      .
239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .      .
242      .      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      .      .      .      .      .      .      .      .
246      .      .      .      .      .      .      .      .
247      .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
251      .      .      .      .      .      .      .      .
252      .      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .
262      .      .      .      .      .      .      .      .
263      .      .      .      .      .      .      .      .
264      .      .      .      .      .      .      .      .
265      .      .      .      .      .      .      .      .
266      .      .      .      .      .      .      .      .
267      .      .      .      .      .      .      .      .
268      .      .      .      .      .      .      .      .
269      .      .      .      .      .      .      .      .
270      .      .      .      .      .      .      .      .
271      .      .      .      .      .      .      .      .
272      .      .      .      .      .      .      .      .
273      .      .      .      .      .      .      .      .
274      .      .      .      .      .      .      .      .
275      .      .      .      .      .      .      .      .
276      .      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .      .
278      .      .      .      .      .      .      .      .
279      .      .      .      .      .      .      .      .
280      .      .      .      .      .      .      .      .
281      .      .      .      .      .      .      .      .
282      .      .      .      .      .      .      .      .
283      .      .      .      .      .      .      .      .
284      .      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .      .
286      .      .      .      .      .      .      .      .
287      .      .      .      .      .      .      .      .
288      .      .      .      .      .      .      .      .
289      .      .      .      .      .      .      .      .
290      .      .      .      .      .      .      .      .
291      .      .      .      .      .      .      .      .
292      .      .      .      .      .      .      .      .
293      .      .      .      .      .      .      .      .
294      .      .      .      .      .      .      .      .
295      .      .      .      .      .      .      .      .
296      .      .      .      .      .      .      .      .
297      .      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .      .
299      .      .      .      .      .      .      .      .
300      .      .      .      .      .      .      .      .
301      .      .      .      .      .      .      .      .
302      .      .      .      .      .      .      .      .
303      .      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .      .
305      .      .      .      .      .      .      .      .
306      .      .      .      .      .      .      .      .
307      .      .      .      .      .      .      .      .
308      .      .      .      .      .      .      .      .
309      .      .      .      .      .      .      .      .
310      .      .      .      .      .      .      .      .
311      .      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .      .
314      .      .      .      .      .      .      .      .
315      .      .      .      .      .      .      .      .
316      .      .      .      .      .      .      .      .
317      .      .      .      .      .      .      .      .
318      .      .      .      .      .      .      .      .
319      .      .      .      .      .      .      .      .
320      .      .      .      .      .      .      .      .
321      .      .      .      .      .      .      .      .
322      .      .      .      .      .      .      .      .
323      .      .      .      .      .      .      .      .
324      .      .      .      .      .      .      .      .
325      .      .      .      .      .      .      .      .
326      .      .      .      .      .      .      .      .
327      .      .      .      .      .      .      .      .
328      .      .      .      .      .      .      .      .
329      .      .      .      .      .      .      .      .
330      .      .      .      .      .      .      .      .
331      .      .      .      .      .      .      .      .
332      .      .      .      .      .      .      .      .
333      .      .      .      .      .      .      .      .
334      .      .      .      .      .      .      .      .
335      .      .      .      .      .      .      .      .
336      .      .      .      .      .      .      .      .
337      .      .      .      .      .      .      .      .
338      .      .      .      .      .      .      .      .
339      .      .      .      .      .      .      .      .
340      .      .      .      .      .      .      .      .
341      .      .      .      .      .      .      .      .
342      .      .      .      .      .      .      .      .
343      .      .      .      .      .      .      .      .
344      .      .      .      .      .      .      .      .
345      .      .      .      .      .      .      .      .
346      .      .      .      .      .      .      .      .
347      .      .      .      .      .      .      .      .
348      .      .      .      .      .      .      .      .
349      .      .      .      .      .      .      .      .
350      .      .      .      .      .      .      .      .
351      .      .      .      .      .      .      .      .
352      .      .      .      .      .      .      .      .
353      .      .      .      .      .      .      .      .
354      .      .      .      .      .      .      .      .
355      .      .      .      .      .      .      .      .
356      .      .      .      .      .      .      .      .
357      .      .      .      .      .      .      .      .
358      .      .      .      .      .      .      .      .
359      .      .      .      .      .      .      .      .
360      .      .      .      .      .      .      .      .
361      .      .      .      .      .      .      .      .
362      .      .      .      .      .      .      .      .
363      .      .      .      .      .      .      .      .
364      .      .      .      .      .      .      .      .
365      .      .      .      .      .      .      .      .
366      .      .      .      .      .      .      .      .
367      .      .      .      .      .      .      .      .
368      .      .      .      .      .      .      .      .
369      .      .      .      .      .      .      .      .
370      .      .      .      .      .      .      .      .
371      .      .      .      .      .      .      .      .
372      .      .      .      .      .      .      .      .
373      .      .      .      .      .      .      .      .
374      .      .      .      .      .      .      .      .
375      .      .      .      .      .      .      .      .
376      .      .      .      .      .      .      .      .
377      .      .      .      .      .      .      .      .
378      .      .      .      .      .      .      .      .
379      .      .      .      .      .      .      .      .
380      .      .      .      .      .      .      .      .
381      .      .      .      .      .      .      .      .
382      .      .      .      .      .      .      .      .
383      .      .      .      .      .      .      .      .
384      .      .      .      .      .      .      .      .
385      .      .      .      .      .      .      .      .
386      .      .      .      .      .      .      .      .
387      .      .      .      .      .      .      .      .
388      .      .      .      .      .      .      .      .
389      .      .      .      .      .      .      .      .
390      .      .      .      .      .      .      .      .
391      .      .      .      .      .      .      .      .
392      .      .      .      .      .      .      .      .
393      .      .      .      .      .      .      .      .
394      .      .      .      .      .      .      .      .
395      .      .      .      .      .      .      .      .
396      .      .      .      .      .      .      .      .
397      .      .      .      .      .      .      .      .
398      .      .      .      .      .      .      .      .
399      .      .
```

```

224      17      2      2      6      0      0      2
          0      0      for(jj=0; jj<en; jj+=bs)
225      56      1      1      20      0      0      7
          0      0      for(i=0; i<n; i++)
226      426     2      2      168     0      0      42
          0      0      for(j=jj; j<jj+bs; j++) {
227      396     1      1      216     10     10     36
          0      0      sum = mr->array[i*n+j];
228      2,556    3      3      1,008    0      0      252
          0      0      for(k=kk; k<kk+bs; k++)
229      5,184    2      2      2,808    19     19     216
          0      0      sum += m1->array[i*n+k] * m2
          ->array[k*n+j];
230      396     1      1      216     0      0      36
          0      0      mr->array[i*n+j] = sum;
231      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
232      1      0      0      1      0      0      0
          0      0      return mr;
233      6      1      1      2      0      0      0
          0      0      }
234      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
235      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
236      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
237      18      3      3      0      0      0      8
          0      0      {
238      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
239      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
240      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
241      6      0      0      0      0      0      4
          0      0      size_t len = 0, tam = DEF_LINE_SZ;
242      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
243      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
244      14      1      1      6      1      0      2
          1      0      0      str = malloc(tam);
245      6      0      0      2      0      0      0
          0      0      if (!str) {
246      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
247      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
248      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
249      .      .      .      .      .      .      .
          .      .      .      .      .      .      .
250      2,201    5      5      734     1      0      147

```

```

0      0      0      while (EOF != (c=fgetc(fp))) && c != '\n'
1      1,160    2      2      435      0      0      290
2      4      4      str[len++]=c;
3      725      0      0      290      0      0      0
4      0      0      if (len==tam-1) {
5      .      .      .      .      .      .      .
6      .      .      .      .      str = realloc(str, tam *= 2);
7      .      .      .      .      .      .      .
8      .      .      .      .      if (!str) {
9      .      .      .      .      .      .      .
10     .      .      .      .      perror("");
11     .      .      .      .      .      .      .
12     .      .      .      .      return NULL;
13     .      .      .      .      .      .      .
14     .      .      .      .      .      .      .
15     .      .      .      .      .      .      .
16     .      .      .      .      .      .      .
17     .      .      .      .      .      .      .
18     .      .      .      .      .      .      .
19     .      .      .      .      .      .      .
20     .      .      .      .      .      .      .
21     .      .      .      .      .      .      .
22     .      .      .      .      .      .      .
23     .      .      .      .      .      .      .
24     .      .      .      .      .      .      .
25     .      .      .      .      .      .      .
26     .      .      .      .      .      .      .
27     .      .      .      .      .      .      .
28     .      .      .      .      .      .      .
29     .      .      .      .      .      .      .
30     .      .      .      .      .      .      .
31     .      .      .      .      .      .      .
32     .      .      .      .      .      .      .
33     .      .      .      .      .      .      .
34     .      .      .      .      .      .      .
35     .      .      .      .      .      .      .
36     .      .      .      .      .      .      .
37     .      .      .      .      .      .      .
38     .      .      .      .      .      .      .
39     .      .      .      .      .      .      .
40     .      .      .      .      .      .      .
41     .      .      .      .      .      .      .
42     .      .      .      .      .      .      .
43     .      .      .      .      .      .      .
44     .      .      .      .      .      .      .
45     .      .      .      .      .      .      .
46     .      .      .      .      .      .      .
47     .      .      .      .      .      .      .
48     .      .      .      .      .      .      .
49     .      .      .      .      .      .      .
50     .      .      .      .      .      .      .
51     .      .      .      .      .      .      .
52     .      .      .      .      .      .      .
53     .      .      .      .      .      .      .
54     .      .      .      .      .      .      .
55     .      .      .      .      .      .      .
56     .      .      .      .      .      .      .
57     .      .      .      .      .      .      .
58     .      .      .      .      .      .      .
59     .      .      .      .      .      .      .
60     .      .      .      .      .      .      .
61     .      .      .      .      .      .      .
62     .      .      .      .      .      .      .
63     .      .      .      .      .      .      .
64     .      .      .      .      .      .      .
65     .      .      .      .      .      .      .
66     .      .      .      .      .      .      .
67     .      .      .      .      .      .      .
68     .      .      .      .      .      .      .
69     .      .      .      .      .      .      .
70     .      .      .      .      .      .      .
71     .      .      .      .      .      .      .
72     .      .      .      .      .      .      .
73     .      .      .      .      .      .      .
74     .      .      .      .      .      .      .
75     .      .      .      .      .      .      .
76     .      .      .      .      .      .      .
77     .      .      .      .      .      .      .
78     .      .      .      .      .      .      .
79     .      .      .      .      .      .      .
80     .      .      .      .      .      .      .
81     .      .      .      .      .      .      .
82     .      .      .      .      .      .      .
83     .      .      .      .      .      .      .
84     .      .      .      .      .      .      .
85     .      .      .      .      .      .      .
86     .      .      .      .      .      .      .
87     .      .      .      .      .      .      .
88     .      .      .      .      .      .      .
89     .      .      .      .      .      .      .
90     .      .      .      .      .      .      .
91     .      .      .      .      .      .      .
92     .      .      .      .      .      .      .
93     .      .      .      .      .      .      .
94     .      .      .      .      .      .      .
95     .      .      .      .      .      .      .
96     .      .      .      .      .      .      .
97     .      .      .      .      .      .      .
98     .      .      .      .      .      .      .
99     .      .      .      .      .      .      .
100    .      .      .      .      .      .      .
101    .      .      .      .      .      .      .
102    .      .      .      .      .      .      .
103    .      .      .      .      .      .      .
104    .      .      .      .      .      .      .
105    .      .      .      .      .      .      .
106    .      .      .      .      .      .      .
107    .      .      .      .      .      .      .
108    .      .      .      .      .      .      .
109    .      .      .      .      .      .      .
110    .      .      .      .      .      .      .
111    .      .      .      .      .      .      .
112    .      .      .      .      .      .      .
113    .      .      .      .      .      .      .
114    .      .      .      .      .      .      .
115    .      .      .      .      .      .      .
116    .      .      .      .      .      .      .
117    .      .      .      .      .      .      .
118    .      .      .      .      .      .      .
119    .      .      .      .      .      .      .
120    .      .      .      .      .      .      .
121    .      .      .      .      .      .      .
122    .      .      .      .      .      .      .
123    .      .      .      .      .      .      .
124    .      .      .      .      .      .      .
125    .      .      .      .      .      .      .
126    .      .      .      .      .      .      .
127    .      .      .      .      .      .      .
128    .      .      .      .      .      .      .
129    .      .      .      .      .      .      .
130    .      .      .      .      .      .      .
131    .      .      .      .      .      .      .
132    .      .      .      .      .      .      .
133    .      .      .      .      .      .      .
134    .      .      .      .      .      .      .
135    .      .      .      .      .      .      .
136    .      .      .      .      .      .      .
137    .      .      .      .      .      .      .
138    .      .      .      .      .      .      .
139    .      .      .      .      .      .      .
140    .      .      .      .      .      .      .
141    .      .      .      .      .      .      .
142    .      .      .      .      .      .      .
143    .      .      .      .      .      .      .
144    .      .      .      .      .      .      .
145    .      .      .      .      .      .      .
146    .      .      .      .      .      .      .
147    .      .      .      .      .      .      .
148    .      .      .      .      .      .      .
149    .      .      .      .      .      .      .
150    .      .      .      .      .      .      .
151    .      .      .      .      .      .      .
152    .      .      .      .      .      .      .
153    .      .      .      .      .      .      .
154    .      .      .      .      .      .      .
155    .      .      .      .      .      .      .
156    .      .      .      .      .      .      .
157    .      .      .      .      .      .      .
158    .      .      .      .      .      .      .
159    .      .      .      .      .      .      .
160    .      .      .      .      .      .      .
161    .      .      .      .      .      .      .
162    .      .      .      .      .      .      .
163    .      .      .      .      .      .      .
164    .      .      .      .      .      .      .
165    .      .      .      .      .      .      .
166    .      .      .      .      .      .      .
167    .      .      .      .      .      .      .
168    .      .      .      .      .      .      .
169    .      .      .      .      .      .      .
170    .      .      .      .      .      .      .
171    .      .      .      .      .      .      .
172    .      .      .      .      .      .      .
173    .      .      .      .      .      .      .
174    .      .      .      .      .      .      .
175    .      .      .      .      .      .      .
176    .      .      .      .      .      .      .
177    .      .      .      .      .      .      .
178    .      .      .      .      .      .      .
179    .      .      .      .      .      .      .
180    .      .      .      .      .      .      .
181    .      .      .      .      .      .      .
182    .      .      .      .      .      .      .
183    .      .      .      .      .      .      .
184    .      .      .      .      .      .      .
185    .      .      .      .      .      .      .
186    .      .      .      .      .      .      .
187    .      .      .      .      .      .      .
188    .      .      .      .      .      .      .
189    .      .      .      .      .      .      .
190    .      .      .      .      .      .      .
191    .      .      .      .      .      .      .
192    .      .      .      .      .      .      .
193    .      .      .      .      .      .      .
194    .      .      .      .      .      .      .
195    .      .      .      .      .      .      .
196    .      .      .      .      .      .      .
197    .      .      .      .      .      .      .
198    .      .      .      .      .      .      .
199    .      .      .      .      .      .      .
200    .      .      .      .      .      .      .
201    .      .      .      .      .      .      .
202    .      .      .      .      .      .      .
203    .      .      .      .      .      .      .
204    .      .      .      .      .      .      .
205    .      .      .      .      .      .      .
206    .      .      .      .      .      .      .
207    .      .      .      .      .      .      .
208    .      .      .      .      .      .      .
209    .      .      .      .      .      .      .
210    .      .      .      .      .      .      .
211    .      .      .      .      .      .      .
212    .      .      .      .      .      .      .
213    .      .      .      .      .      .      .
214    .      .      .      .      .      .      .
215    .      .      .      .      .      .      .
216    .      .      .      .      .      .      .
217    .      .      .      .      .      .      .
218    .      .      .      .      .      .      .
219    .      .      .      .      .      .      .
220    .      .      .      .      .      .      .
221    .      .      .      .      .      .      .
222    .      .      .      .      .      .      .
223    .      .      .      .      .      .      .
224    .      .      .      .      .      .      .
225    .      .      .      .      .      .      .
226    .      .      .      .      .      .      .
227    .      .      .      .      .      .      .
228    .      .      .      .      .      .      .
229    .      .      .      .      .      .      .
230    .      .      .      .      .      .      .
231    .     
```

```

276      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .
278      .      .      .      .      .      .      .
279      9      1      1      .      6      0      0      3
280      0      .      .      0      m->rows = rows;
281      9      1      1      .      6      0      0      3
282      0      .      .      0      m->cols = cols;
283      51     2      2      .      21     0      0      3
284      0      .      .      0      if (!(m->array = malloc(sizeof(
285      double)*rows*cols))) {
286      .      .      .      .      free(m);
287      .      .      .      .      perror("");
288      .      .      .      .      return NULL;
289      .      .      .      .      }
290      .      .      .      .      .
291      .      .      .      .      .
292      3      1      1      .      3      0      0      0
293      0      .      .      0      return m;
294      18     1      1      .      6      0      0      0
295      0      .      .      0 }
296      .      .      .      .      .      .      .
297      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .
299      .      .      .      .      void
300      .      .      .      .      destroy_matrix(matrix_t* m)
301      27     1      1      .      0      0      0      12
302      0      .      .      0 {
303      9      0      0      .      3      0      0      0
304      0      .      .      0      if (!m) return;
305      24     1      1      .      12     0      0      0
306      0      .      .      0      free(m->array);
307      24     1      1      .      9      0      0      0
308      0      .      .      0      free(m);
309      18     0      0      .      6      0      0      0
310      0      .      .      0 }
311      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .
314      .      .      .      .      int
315      .      .      .      .      .      .      .
316      .      .      .      .      print_matrix(FILE* fp, matrix_t* m)
317      10     2      2      .      0      0      0      5
318      0      .      .      0 {
319      .      .      .      .      .      .      .
320      .      .      .      .      size_t i, j;
321      .      .      .      .      .      .      .

```

```

303         .      .      .      size_t n;
304         3      1      1      2      0      0      1
305         0      0      0      n = m->rows;
306         13     1      1      6      1      1      0
307         0      0      if (fprintf(fp, "%lu", (unsigned
308         long) m->rows) < 0) {
309         .      .      .      .      .      .      .
310         .      .      .      perror("");
311         .      .      .      .      .      .      .
312         .      .      .      return -1;
313         .      .      .      .      .      .      .
314         .      .      .      }
315         56     2      2      20     0      0      7
316         0      0      for(i=0; i<n; i++)
317         336     2      2      120     0      0      42
318         0      0      for (j=0; j<n; j++)
319         792     3      3      360     40     0      0
320         0      0      if (fprintf(fp, " %g", m->array[i
321         *n+j]) < 0) {
322         .      .      .      .      .      .      .
323         .      .      .      perror("");
324         .      .      .      .      .      .      .
325         .      .      .      return -1;
326         .      .      .      .      .      .      .
327         .      .      .      }
328         10     1      1      4      0      0      0
329         0      0      if (fprintf(fp, "\n") < 0) {
330         .      .      .      .      .      .      .
331         .      .      .      perror("");
332         .      .      .      .      .      .      .
333         .      .      .      return -1;
334         .      .      .      .      .      .      .
335         .      .      .      }
336         1      1      1      0      0      0      0
337         0      0      return 0;
338         6      1      1      2      0      0      0
339         0      0      }

```

```

322 Ir      I1mr ILmr Dr      D1mr  DLmr Dw      D1mw
323      DLmw
324 146,819,332 120 119 52,436,987 20,564 38 20,973,112
      1,328,672 1,310,750 events annotated

```

```

1 7 56 84 81 58 109 114 54 63 131 99 85 146 137 153 103 107 142
   50 174 152 80 61 102 87 44 122 120 57 110 143 111 93 153
   139 116 90 106 116 61 141 128 92 55 30 64 51 74 71 74
2 bVEX; rerun with -h for copyright info
3 ==620== Command: /tmp/02-mmult
4 ==620== Parent PID: 597
5 ==620==

```

```

6  --620-- Warning: Cannot auto-detect cache config, using
   defaults.
7  --620--          Run with -v to see.
8  ==620==
9  ==620== I   refs:      147,273,130
10 ==620== I1  misses:      2,405
11 ==620== LLi misses:      2,387
12 ==620== I1  miss rate:    0.00%
13 ==620== LLi miss rate:    0.00%
14 ==620==
15 ==620== D   refs:      73,583,952 (52,551,913 rd  +
   21,032,039 wr)
16 ==620== D1  misses:      1,354,741 ( 25,330 rd  +
   1,329,411 wr)
17 ==620== LLd misses:      1,314,228 ( 2,936 rd  +
   1,311,292 wr)
18 ==620== D1  miss rate:    1.8% ( 0.0%  +
   6.3%  )
19 ==620== LLd miss rate:    1.8% ( 0.0%  +
   6.2%  )
20 ==620==
21 ==620== LL refs:      1,357,146 ( 27,735 rd  +
   1,329,411 wr)
22 ==620== LL misses:      1,316,615 ( 5,323 rd  +
   1,311,292 wr)
23 ==620== LL miss rate:    0.6% ( 0.0%  +
   6.2%  )
24 -----
25 I1 cache:      32768 B, 32 B, 4-way associative
26 D1 cache:      32768 B, 32 B, direct-mapped
27 LL cache:      524288 B, 32 B, 8-way associative
28 Command:      /tmp/02-mmult
29 Data file:      cachegrind.out.620
30 Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
31 Events shown:   Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
32 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Thresholds:     0.1 100 100 100 100 100 100 100 100
34 Include dirs:
35 User annotated:  /root/CARPETA/tp2-2020-2q-src/main.c
36 Auto-annotation: off
37
38 -----
39 Ir           I1mr ILmr Dr           D1mr   DLmr Dw           D1mw
   DLmw
40 -----
41 147,273,130 2,405 2,387 52,551,913 25,330 2,936 21,032,039
   1,329,411 1,311,292 PROGRAM TOTALS
42 -----
43
44 Ir           I1mr ILmr Dr           D1mr   DLmr Dw           D1mw

```

45	DLmw	file:function	
46	146,815,701	29 29 52,436,124 20,521 41 20,972,595	
47	1,328,651	1,310,731 /root/CARPETA/tp2-2020-2q-src/main.c:	
48	matrix_multiply		
49	-- User-annotated source:	/root/CARPETA/tp2-2020-2q-src/main.c	
51	Ir	I1mr ILmr Dr	D1mr DLmr Dw D1mw
52		DLmw	
53	-- line 18	-----	
54	.	.	.
55	.	.	.
56	.	.	.
57	.	.	.
58	.	.	.
59	.	.	.
60	.	.	.
61	.	.	.
62	10	2 2	0 0 0 5
63	0	0	{
64	.	.	.
65	.	.	.
66	.	.	.
67	.	.	.
68	1	0 0	0 0 0 1
69	0	0	char *line = NULL;
70	.	.	.


```

71      .      .      .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .      .      .
76      .      .      .      .      .      .      .      .      .      .
77      .      .      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .      .      .
82      .      .      .      .      .      .      .      .      .      .
83      .      .      .      .      .      .      .      .      .      .
84      .      .      .      .      .      .      .      .      .      .
85      .      .      .      .      .      .      .      .      .      .
86      .      .      .      .      .      .      .      .      .      .
87      .      .      .      .      .      .      .      .      .      .
88      .      .      .      .      .      .      .      .      .      .
89      .      .      .      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .      .      .      .
91      .      .      .      .      .      .      .      .      .      .
92      .      .      .      .      .      .      .      .      .      .
93      .      .      .      .      .      .      .      .      .      .
94      .      .      .      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .      .      .      .
97      .      .      .      .      .      .      .      .      .      .

/* auxiliar variables */

long l;

double e;

size_t lineno = 1;

struct timespec t0;

struct timespec t1;

double dt;

size_t i;

for(; !feof(stdin); lineno++) {
    a=b=c=NULL;

    line = read_line(stdin);

    if (!line) goto _exit_main;
    if (line[0] == 0) break;

    /* parse dimension */
    nptr = line;
    l = strtol(nptr, &endptr, 10);
    if (errno) {
        perror("");
        goto _exit_main;
    }
    if (nptr == endptr) {
        fprintf(stderr, "missing
dimension");
    }

```

```

98         . . . . . goto _exit_main;
99         . . . . . }
100         3 2 2 0 1 0 0 0
101         0 0 0 if (l < 1) {
102         . . . . . fprintf(stderr, "invalid
103         dimension");
104         . . . . . goto _exit_main;
105         . . . . . }
106         2 1 1 0 1 0 0 1
107         0 0 0 n = (size_t) l;
108         . . . . . #if 0
109         . . . . . /* parse blocksize */
110         . . . . . nptr = endptr;
111         . . . . . l = strtol(nptr, &endptr, 10);
112         . . . . . if (errno) {
113         . . . . . perror("");
114         . . . . . goto _exit_main;
115         . . . . . }
116         -- line 75 -----
117         -- line 83 -----
118         . . . . . }
119         . . . . . bs = (size_t) l;
120         . . . . . if (n % bs) {
121         . . . . . fprintf(stderr, "block size
122         doesn't match");
123         . . . . . goto _exit_main;
124         . . . . . }
125         . . . . . #else
126         2 0 0 1 0 0 1
127         0 0 0 bs = n;
128         . . . . . #endif
129         . . . . .

```

```

125      .      .      .      .      .      .      .      .
126      11      1      1      5      1      0      1
127      0      0      if (!(a = create_matrix(n, n)))
128      .      .      .      .      .      .      .
129      500      3      3      199      0      0      50
130      0      0      0      for (i=0; i < n*n; i++) {
131      441      1      1      147      0      0      49
132      0      0      0      e = strtod(npstr, &endptr);
133      392      1      1      147      0      0      0
134      0      0      if (errno) {
135      .      .      .      .      .      .      .
136      .      .      .      .      .      .      .
137      .      .      .      .      .      .      .
138      .      .      .      .      .      .      .
139      .      .      .      .      .      .      .
140      343      1      1      196      0      0      49
141      11      11      a->array[i] = e;
142      .      .      .      .      .      .      .
143      .      .      .      .      .      .      .
144      11      1      1      5      0      0      1
145      0      0      if (!(b = create_matrix(n, n)))
146      .      .      .      .      .      .      .
147      500      2      2      199      0      0      50
148      0      0      0      for (i=0; i < n*n; i++) {
149      441      1      1      147      0      0      49
150      0      0      0      e = strtod(npstr, &endptr);
151      392      1      1      147      0      0      0
152      0      0      if (errno) {

```

```

151         . . . . .
152         . . . . . perror("");
153         . . . . . goto _exit_main;
154         . . . . . }
155         196 1 1 98 0 0 0
156         0 0 if (nptr == endptr) {
157         . . . . . fprintf(stderr, "missing B
158         . . . . . matrix element");
159         . . . . . goto _exit_main;
160         . . . . . }
161         343 1 1 196 0 0 49
162         11 11 b->array[i] = e;
163         . . . . . }
164         . . . . .
165         8 1 1 2 0 0 0
166         0 0 clock_gettime(CLOCK_REALTIME, &t0
167         );
168         . . . . .
169         . . . . .
170         . . . . . /* multiply matrixes */
171         13 2 2 6 1 1 1
172         0 0 if (!(c = matrix_multiply(a, b, bs
173         )))
174         . . . . . goto _exit_main;
175         . . . . .
176         8 1 1 2 0 0 0
177         0 0 clock_gettime(CLOCK_REALTIME, &t1
178         );
179         . . . . .
180         . . . . .
181         7 1 1 2 1 1 1
182         1 1 dt = (float) (t1.tv_sec - t0.
183         tv_sec);
184         12 1 1 5 2 2 1
185         0 0 dt = dt + ((float)(t1.tv_nsec - t0
186         .tv_nsec)) / 1.0e9;
187         . . . . .
188         . . . . .
189         13 3 2 5 2 2 0
190         0 0 if(print_matrix(stdout, c) == -1)
191         . . . . . goto _exit_main;
192         . . . . .
193         . . . . .
194         . . . . .

```

```

175      12      1      1      7      0      0      0
      0      0      fprintf(stderr, "time: %g\n", dt);
176      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
177      6      1      1      3      0      0      0
      0      0      free(line);
178      6      1      1      3      0      0      0
      0      0      destroy_matrix(a);
179      6      1      1      3      0      0      0
      0      0      destroy_matrix(b);
180      6      0      0      3      0      0      0
      0      0      destroy_matrix(c);
181      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
182      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
183      3      1      1      0      0      0      0
      0      0      return 0;
184      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
185      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
186      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
192      6      1      1      2      1      0      0
      0      0      0 }
193      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
194      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
196      11     2      2      0      0      0      6
      0      0      0 {
197      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
198      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .
      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .
      .      .      .      .      .      .      .

```

```

201      3      1      1      0      2      0      0      1
          0      0      n = m1->rows;
202      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
203      11     1      1      5      0      0      1
          0      0      if(!(mr = create_matrix(n,n)))
          return NULL;
204      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
205      9      2      2      3      0      0      1
          0      0      en = bs*(n/bs);
206      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
207      64     2      2      23     0      0      8
          0      0      for(i=0; i<n; i++)
208      448     1      1      161     0      0      56
          0      0      for(j=0; j<n; j++)
209      539     1      1      245     0      0      98
          11     11      mr->array[i*n+j] = 0.0;
210      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
211      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
212      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
214      2      0      0      0      0      0      1
          0      0      size_t dim = 1024*1024*10;
215      9      1      1      3      0      0      1
          0      0      int *v = malloc(dim*sizeof(int));
216      83,886,088 2      2 31,457,282 20,480      0 10,485,761
          0      0      for (j = 0; j < dim; ++j)
217      62,914,560 0      0 20,971,520      0      0 10,485,760
          1,328,640 1,310,720      v[j] = -1;
218      6      1      1      3      2      2      0
          0      0      free(v);
219      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
220      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
221      .      .      .      .      .      .      .      .
          .      .      .      .      .      .      .      .
222      17     2      2      6      0      0      2
          0      0      for(kk=0; kk<en; kk+=bs)
223      17     2      2      6      0      0      2
          0      0      for(jj=0; jj<en; jj+=bs)
224      64     1      1      23     0      0      8
          0      0      for(i=0; i<n; i++)
225      567     2      2      224     0      0      56
          0      0      for(j=jj; j<jj+bs; j++) {
226      539     1      1      294     13     13     49
          0      0      sum = mr->array[i*n+j];
227      3,969     3      3      1,568     0      0      392

```

```

228      0      0      for(k=kk; k<kk+bs; k++)
8,232      2      2      4,459      26      26      343
0      0      sum += m1->array[i*n+k] * m2
->array[k*n+j];
229      539      1      1      294      0      0      49
0      0      mr->array[i*n+j] = sum;
230      .      .      .      .      .      .      .
.      .      .      .      .      .      .
231      1      0      0      1      0      0      0
0      0      0      return mr;
232      6      1      1      2      0      0      0
0      0      0      }
233      .      .      .      .      .      .      .
.      .      .      .      .      .      .
234      .      .      .      .      .      .      .
.      .      .      char*
235      .      .      .      .      .      .      .
.      .      .      read_line(FILE *fp)
236      18      3      3      0      0      0      8
0      0      0      {
237      .      .      .      .      .      .      .
.      .      .      .      .      .      .
238      .      .      .      .      .      .      .
.      .      .      .      .      .      .
239      .      .      .      .      .      .      .
.      .      .      .      .      .      .
240      6      0      0      0      0      0      4
0      0      0      size_t len = 0, tam = DEF_LINE_SZ;
241      .      .      .      .      .      .      .
.      .      .      char* str;
242      .      .      .      .      .      .      .
.      .      .      .      .      .      .
243      14      1      1      6      1      0      2
1      0      0      str = malloc(tam);
244      6      0      0      2      0      0      0
0      0      0      if (!str) {
245      .      .      .      .      .      .      .
.      .      .      perror("");
246      .      .      .      .      .      .      .
.      .      .      return NULL;
247      .      .      .      .      .      .      .
.      .      .      .      .      .      .
248      .      .      .      .      .      .      .
.      .      .      .      .      .      .
249      2,981      5      5      994      1      0      199
0      0      while (EOF != (c=fgetc(fp)) && c != '\n'
') {
250      1,576      2      2      591      0      0      394
6      6      str[len++]=c;
251      985      0      0      394      0      0      0
0      0      if (len==tam-1) {
252      .      .      .      .      .      .      .
.      .      .      str = realloc(str, tam *= 2);
253      .      .      .      .      .      .      .

```

```

254         . . . . if (!str) {
255         . . . . . perror("");
256         . . . . . return NULL;
257         . . . . }
258         . . . . }
259         . . . . }
260     8 1 1 0 2 0 0 0
261     0 0 0 if (c != EOF)
262     7 0 0 2 0 0 2
263     0 0 0 str[len++]='\n';
264     . . . . .
265     12 1 1 4 0 0 4
266     0 0 0 str[len++]='\0';
267     2 0 0 2 0 0 0
268     0 0 0 return str;
269     12 1 1 4 0 0 0
270     0 0 0 }
271     . . . . .
272     . . . . .
273     . . . . .
274     . . . . .
275     . . . . .
276     . . . . .
277     . . . . .
278     . . . . .
279     . . . . .
280     . . . . .
281     . . . . .
282     . . . . .
283     . . . . .
284     . . . . .
285     . . . . .
286     . . . . .
287     . . . . .
288     . . . . .
289     . . . . .
290     . . . . .
291     . . . . .
292     . . . . .
293     . . . . .
294     . . . . .
295     . . . . .
296     . . . . .
297     . . . . .
298     . . . . .
299     . . . . .
300     . . . . .
301     . . . . .
302     . . . . .
303     . . . . .
304     . . . . .
305     . . . . .
306     . . . . .
307     . . . . .
308     . . . . .
309     . . . . .
310     . . . . .
311     . . . . .
312     . . . . .
313     . . . . .
314     . . . . .
315     . . . . .
316     . . . . .
317     . . . . .
318     . . . . .
319     . . . . .
320     . . . . .
321     . . . . .
322     . . . . .
323     . . . . .
324     . . . . .
325     . . . . .
326     . . . . .
327     . . . . .
328     . . . . .
329     . . . . .
330     . . . . .
331     . . . . .
332     . . . . .
333     . . . . .
334     . . . . .
335     . . . . .
336     . . . . .
337     . . . . .
338     . . . . .
339     . . . . .
340     . . . . .
341     . . . . .
342     . . . . .
343     . . . . .
344     . . . . .
345     . . . . .
346     . . . . .
347     . . . . .
348     . . . . .
349     . . . . .
350     . . . . .
351     . . . . .
352     . . . . .
353     . . . . .
354     . . . . .
355     . . . . .
356     . . . . .
357     . . . . .
358     . . . . .
359     . . . . .
360     . . . . .
361     . . . . .
362     . . . . .
363     . . . . .
364     . . . . .
365     . . . . .
366     . . . . .
367     . . . . .
368     . . . . .
369     . . . . .
370     . . . . .
371     . . . . .
372     . . . . .
373     . . . . .
374     . . . . .
375     . . . . .
376     . . . . .
377     . . . . .
378     . . . . .
379     . . . . .
380     . . . . .
381     . . . . .
382     . . . . .
383     . . . . .
384     . . . . .
385     . . . . .
386     . . . . .
387     . . . . .
388     . . . . .
389     . . . . .
390     . . . . .
391     . . . . .
392     . . . . .
393     . . . . .
394     . . . . .
395     . . . . .
396     . . . . .
397     . . . . .
398     . . . . .
399     . . . . .
400     . . . . .
401     . . . . .
402     . . . . .
403     . . . . .
404     . . . . .
405     . . . . .
406     . . . . .
407     . . . . .
408     . . . . .
409     . . . . .
410     . . . . .
411     . . . . .
412     . . . . .
413     . . . . .
414     . . . . .
415     . . . . .
416     . . . . .
417     . . . . .
418     . . . . .
419     . . . . .
420     . . . . .
421     . . . . .
422     . . . . .
423     . . . . .
424     . . . . .
425     . . . . .
426     . . . . .
427     . . . . .
428     . . . . .
429     . . . . .
430     . . . . .
431     . . . . .
432     . . . . .
433     . . . . .
434     . . . . .
435     . . . . .
436     . . . . .
437     . . . . .
438     . . . . .
439     . . . . .
440     . . . . .
441     . . . . .
442     . . . . .
443     . . . . .
444     . . . . .
445     . . . . .
446     . . . . .
447     . . . . .
448     . . . . .
449     . . . . .
450     . . . . .
451     . . . . .
452     . . . . .
453     . . . . .
454     . . . . .
455     . . . . .
456     . . . . .
457     . . . . .
458     . . . . .
459     . . . . .
460     . . . . .
461     . . . . .
462     . . . . .
463     . . . . .
464     . . . . .
465     . . . . .
466     . . . . .
467     . . . . .
468     . . . . .
469     . . . . .
470     . . . . .
471     . . . . .
472     . . . . .
473     . . . . .
474     . . . . .
475     . . . . .
476     . . . . .
477     . . . . .
478     . . . . .
479     . . . . .
480     . . . . .
481     . . . . .
482     . . . . .
483     . . . . .
484     . . . . .
485     . . . . .
486     . . . . .
487     . . . . .
488     . . . . .
489     . . . . .
490     . . . . .
491     . . . . .
492     . . . . .
493     . . . . .
494     . . . . .
495     . . . . .
496     . . . . .
497     . . . . .
498     . . . . .
499     . . . . .
500     . . . . .
501     . . . . .
502     . . . . .
503     . . . . .
504     . . . . .
505     . . . . .
506     . . . . .
507     . . . . .
508     . . . . .
509     . . . . .
510     . . . . .
511     . . . . .
512     . . . . .
513     . . . . .
514     . . . . .
515     . . . . .
516     . . . . .
517     . . . . .
518     . . . . .
519     . . . . .
520     . . . . .
521     . . . . .
522     . . . . .
523     . . . . .
524     . . . . .
525     . . . . .
526     . . . . .
527     . . . . .
528     . . . . .
529     . . . . .
530     . . . . .
531     . . . . .
532     . . . . .
533     . . . . .
534     . . . . .
535     . . . . .
536     . . . . .
537     . . . . .
538     . . . . .
539     . . . . .
540     . . . . .
541     . . . . .
542     . . . . .
543     . . . . .
544     . . . . .
545     . . . . .
546     . . . . .
547     . . . . .
548     . . . . .
549     . . . . .
550     . . . . .
551     . . . . .
552     . . . . .
553     . . . . .
554     . . . . .
555     . . . . .
556     . . . . .
557     . . . . .
558     . . . . .
559     . . . . .
560     . . . . .
561     . . . . .
562     . . . . .
563     . . . . .
564     . . . . .
565     . . . . .
566     . . . . .
567     . . . . .
568     . . . . .
569     . . . . .
570     . . . . .
571     . . . . .
572     . . . . .
573     . . . . .
574     . . . . .
575     . . . . .
576     . . . . .
577     . . . . .
578     . . . . .
579     . . . . .
580     . . . . .
581     . . . . .
582     . . . . .
583     . . . . .
584     . . . . .
585     . . . . .
586     . . . . .
587     . . . . .
588     . . . . .
589     . . . . .
590     . . . . .
591     . . . . .
592     . . . . .
593     . . . . .
594     . . . . .
595     . . . . .
596     . . . . .
597     . . . . .
598     . . . . .
599     . . . . .
600     . . . . .
601     . . . . .
602     . . . . .
603     . . . . .
604     . . . . .
605     . . . . .
606     . . . . .
607     . . . . .
608     . . . . .
609     . . . . .
610     . . . . .
611     . . . . .
612     . . . . .
613     . . . . .
614     . . . . .
615     . . . . .
616     . . . . .
617     . . . . .
618     . . . . .
619     . . . . .
620     . . . . .
621     . . . . .
622     . . . . .
623     . . . . .
624     . . . . .
625     . . . . .
626     . . . . .
627     . . . . .
628     . . . . .
629     . . . . .
630     . . . . .
631     . . . . .
632     . . . . .
633     . . . . .
634     . . . . .
635     . . . . .
636     . . . . .
637     . . . . .
638     . . . . .
639     . . . . .
640     . . . . .
641     . . . . .
642     . . . . .
643     . . . . .
644     . . . . .
645     . . . . .
646     . . . . .
647     . . . . .
648     . . . . .
649     . . . . .
650     . . . . .
651     . . . . .
652     . . . . .
653     . . . . .
654     . . . . .
655     . . . . .
656     . . . . .
657     . . . . .
658     . . . . .
659     . . . . .
660     . . . . .
661     . . . . .
662     . . . . .
663     . . . . .
664     . . . . .
665     . . . . .
666     . . . . .
667     . . . . .
668     . . . . .
669     . . . . .
670     . . . . .
671     . . . . .
672     . . . . .
673     . . . . .
674     . . . . .
675     . . . . .
676     . . . . .
677     . . . . .
678     . . . . .
679     . . . . .
680     . . . . .
681     . . . . .
682     . . . . .
683     . . . . .
684     . . . . .
685     . . . . .
686     . . . . .
687     . . . . .
688     . . . . .
689     . . . . .
690     . . . . .
691     . . . . .
692     . . . . .
693     . . . . .
694     . . . . .
695     . . . . .
696     . . . . .
697     . . . . .
698     . . . . .
699     . . . . .
700     . . . . .
701     . . . . .
702     . . . . .
703     . . . . .
704     . . . . .
705     . . . . .
706     . . . . .
707     . . . . .
708     . . . . .
709     . . . . .
710     . . . . .
711     . . . . .
712     . . . . .
713     . . . . .
714     . . . . .
715     . . . . .
716     . . . . .
717     . . . . .
718     . . . . .
719     . . . . .
720     . . . . .
721     . . . . .
722     . . . . .
723     . . . . .
724     . . . . .
725     . . . . .
726     . . . . .
727     . . . . .
728     . . . . .
729     . . . . .
730     . . . . .
731     . . . . .
732     . . . . .
733     . . . . .
734     . . . . .
735     . . . . .
736     . . . . .
737     . . . . .
738     . . . . .
739     . . . . .
740     . . . . .
741     . . . . .
742     . . . . .
743     . . . . .
744     . . . . .
745     . . . . .
746     . . . . .
747     . . . . .
748     . . . . .
749     . . . . .
750     . . . . .
751     . . . . .
752     . . . . .
753     . . . . .
754     . . . . .
755     . . . . .
756     . . . . .
757     . . . . .
758     . . . . .
759     . . . . .
760     . . . . .
761     . . . . .
762     . . . . .
763     . . . . .
764     . . . . .
765     . . . . .
766     . . . . .
767     . . . . .
768     . . . . .
769     . . . . .
770     . . . . .
771     . . . . .
772     . . . . .
773     . . . . .
774     . . . . .
775     . . . . .
776     . . . . .
777     . . . . .
778     . . . . .
779     . . . . .
780     . . . . .
781     . . . . .
782     . . . . .
783     . . . . .
784     . . . . .
785     . . . . .
786     . . . . .
787     . . . . .
788     . . . . .
789     . . . . .
790     . . . . .
791     . . . . .
792     . . . . .
793     . . . . .
794     . . . . .
795     . . . . .
796     . . . . .
797     . . . . .
798     . . . . .
799     . . . . .
800     . . . . .
801     . . . . .
802     . . . . .
803     . . . . .
804     . . . . .
805     . . . . .
806     . . . . .
807     . . . . .
808     . . . . .
809     . . . . .
810     . . . . .
811     . . . . .
812     . . . . .
813     . . . . .
814     . . . . .
815     . . . . .
816     . . . . .
817     . . . . .
818     . . . . .
819     . . . . .
820     . . . . .
821     . . . . .
822     . . . . .
823     . . . . .
824     . . . . .
825     . . . . .
826     . . . . .
827     . . . . .
828     . . . . .
829     . . . . .
830     . . . . .
831     . . . . .
832     . . . . .
833     . . . . .
834     . . . . .
835     . . . . .
836     . . . . .
837     . . . . .
838     . . . . .
839     . . . . .
840     . . . . .
841     . . . . .
842     . . . . .
843     . . . . .
844     . . . . .
845     . . . . .
846     . . . . .
847     . . . . .
848     . . . . .
849     . . . . .
850     . . . . .
851     . . . . .
852     . . . . .
853     . . . . .
854     . . . . .
855     . . . . .
856     . . . . .
857     . . . . .
858     . . . . .
859     . . . . .
860     . . . . .
861     . . . . .
862     . . . . .
863     . . . . .
864     . . . . .
865     . . . . .
866     . . . . .
867     . . . . .
868     . . . . .
869     . . . . .
870     . . . . .
871     . . . . .
872     . . . . .
873     . . . . .
874     . . . . .
875     . . . . .
876     . . . . .
877     . . . . .
878     . . . . .
879     . . . . .
880     . . . . .
881     . . . . .
882     . . . . .
883     . . . . .
884     . . . . .
885     . . . . .
886     . . . . .
887     . . . . .
888     . . . . .
889     . . . . .
890     . . . . .
891     . . . . .
892     . . . . .
893     . . . . .
894     . . . . .
895     . . . . .
896     . . . . .
897     . . . . .
898     . . . . .
899     . . . . .
900     . . . . .
901     . . . . .
902     . . . . .
903     . . . . .
904     . . . . .
905     . . . . .
906     . . . . .
907     . . . . .
908     . . . . .
909     . . . . .
910     . . . . .
911     . . . . .
912     . . . . .
913     . . . . .
914     . . . . .
915     . . . . .
916     . . . . .
917     . . . . .
918     . . . . .
919     . . . . .
920     . . . . .
921     . . . . .
922     . . . . .
923     . . . . .
924     . . . . .
925     . . . . .
926     . . . . .
927     . . . . .
928     . . . . .
929     . . . . .
930     . . . . .
931     . . . . .
932     . . . . .
933     . . . . .
934     . . . . .
935     . . . . .
936     . . . . .
937     . . . . .
938     . . . . .
939     . . . . .
940     . . . . .
941     . . . . .
942     . . . . .
943     . . . . .
944     . . . . .
945     . . . . .
946     . . . . .
947     . . . . .
948     . . . . .
949     . . . . .
950     . . . . .
951     . . . . .
952     . . . . .
953     . . . . .
954     . . . . .
955     . . . . .
956     . . . . .
957     . . . . .
958     . . . . .
959     . . . . .
960     . . . . .
961     . . . . .
962     . . . . .
963     . . . . .
964     . . . . .
965     . . . . .
966     . . . . .
967     . . . . .
968     . . . . .
969     . . . . .
970     . . . . .
971     . . . . .
972     . . . . .
973     . . . . .
974     . . . . .
975     . . . . .
976     . . . . .
977     . . . . .
978     . . . . .
979     . . . . .
980     . . . . .
981     . . . . .
982     . . . . .
983     . . . . .
984     . . . . .
985     . . . . .
986     . . . . .
987     . . . . .
988     . . . . .
989     . . . . .
990     . . . . .
991     . . . . .
992     . . . . .
993     . . . . .
994     . . . . .
995     . . . . .
996     . . . . .
997     . . . . .
998     . . . . .
999     . . . . .
1000    . . . . .

```



```

280      0      0      m->cols = cols;
51      2      2      21      0      0      3
      0      0      if (!(m->array = malloc(sizeof(
double)*rows*cols))) {
281      .      .      .      .      .      .
      .      .      .      free(m);
282      .      .      .      .      .      .
      .      .      .      perror("");
283      .      .      .      .      .      .
      .      .      .      return NULL;
284      .      .      .      .      .      .
      .      .      .      }
285      .      .      .      .      .      .
      .      .      .      .      .      .
286      3      1      1      3      0      0      0
      0      0      return m;
287      18      1      1      6      0      0      0
      0      0      }
288      .      .      .      .      .      .
      .      .      .      .      .      .
289      .      .      .      void
290      .      .      .      .      .      .
      .      .      .      destroy_matrix(matrix_t* m)
291      27      1      1      0      0      0      12
      0      0      0      {
292      9      0      0      3      0      0      0
      0      0      if (!m) return;
293      24      1      1      12      0      0      0
      0      0      free(m->array);
294      24      1      1      9      0      0      0
      0      0      free(m);
295      18      0      0      6      0      0      0
      0      0      }
296      .      .      .      .      .      .
      .      .      .      .      .      .
297      .      .      .      .      .      .
      .      .      .      int
298      .      .      .      .      .      .
      .      .      .      print_matrix(FILE* fp, matrix_t* m)
299      10      2      2      0      0      0      5
      0      0      0      {
300      .      .      .      .      .      .
      .      .      .      size_t i, j;
301      .      .      .      .      .      .
      .      .      .      size_t n;
302      3      1      1      2      0      0      1
      0      0      n = m->rows;
303      13      1      1      6      1      1      0
      0      0      if (fprintf(fp, "%lu", (unsigned
long) m->rows) < 0) {
304      .      .      .      .      .      .
      .      .      .      perror("");
305      .      .      .      .      .      .

```

```

306         .      .      .      .      .      .      .      .      .      .
307         .      .      .      .      .      .      .      .      .      .
308         64      2      2      .      23      0      0      .      8
309         0      .      0      .      for(i=0; i<n; i++)
310         448      2      2      .      161      0      0      .      56
311         0      .      0      .      for (j=0; j<n; j++)
312         1,078      3      3      .      490      49      0      .      0
313         0      .      0      .      if (fprintf(fp, " %g", m->array[i*n
314         +j]) < 0) {
315         .      .      .      .      .      .      .      .      .
316         .      .      .      .      perror("");
317         .      .      .      .      .      .      .      .      .
318         .      .      .      .      return -1;
319         .      .      .      .      .      .      .      .      .
320         .      .      .      .      .      .      .      .      .
321         .      .      .      .      .      .      .      .      .
322         .      .      .      .      .      .      .      .      .
323         10      1      1      .      4      0      0      .      0
324         0      .      0      .      if (fprintf(fp, "\n") < 0) {
325         .      .      .      .      .      .      .      .      .
326         .      .      .      .      perror("");
327         .      .      .      .      .      .      .      .      .
328         .      .      .      .      return -1;
329         .      .      .      .      .      .      .      .      .
330         .      .      .      .      .      .      .      .      .
331         .      .      .      .      .      .      .      .      .
332         .      .      .      .      .      .      .      .      .
333         1      1      1      .      0      0      0      .      0
334         0      .      0      .      return 0;
335         6      1      1      .      2      0      0      .      0
336         0      .      0      .      }

```

```

321 Ir      I1mr ILmr Dr      D1mr  DLmr Dw      D1mw
322      DLmw
323 146,827,393 120 119 52,440,658 20,584 48 20,973,736
      1,328,682 1,310,760 events annotated

```

```

1 8 240 184 111 110 129 175 191 230 105 183 118 82 140 181 172
   148 179 136 110 107 151 113 116 165 185 178 143 170 219 133
   177 202 240 232 166 119 161 187 203 268 181 158 128 109
   199 101 128 205 229 254 165 163 208 223 208 251 178 181 135
   176 205 159 197 186
2  PID: 597
3  ==623==
4  --623-- Warning: Cannot auto-detect cache config, using
      defaults.
5  --623--      Run with -v to see.
6  ==623==
7  ==623== I      refs:      147,337,268
8  ==623== I1  misses:      2,404
9  ==623== LLi misses:      2,384
10 ==623== I1  miss rate:    0.00%
11 ==623== LLi miss rate:    0.00%

```

```

12 ==623==
13 ==623== D   refs:          73,608,453  (52,568,667 rd   +
      21,039,786 wr)
14 ==623== D1  misses:       1,354,925  (   25,352 rd   +
      1,329,573 wr)
15 ==623== LLd misses:       1,314,257  (    2,949 rd   +
      1,311,308 wr)
16 ==623== D1  miss rate:         1.8% (    0.0%   +
      6.3%   )
17 ==623== LLd miss rate:         1.8% (    0.0%   +
      6.2%   )
18 ==623==
19 ==623== LL refs:          1,357,329  (   27,756 rd   +
      1,329,573 wr)
20 ==623== LL misses:       1,316,641  (    5,333 rd   +
      1,311,308 wr)
21 ==623== LL miss rate:         0.6% (    0.0%   +
      6.2%   )
22 -----
23 I1 cache:          32768 B, 32 B, 4-way associative
24 D1 cache:          32768 B, 32 B, direct-mapped
25 LL cache:         524288 B, 32 B, 8-way associative
26 Command:          /tmp/02-mmult
27 Data file:         cachegrind.out.623
28 Events recorded:   Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
29 Events shown:      Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
30 Event sort order:  Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
31 Thresholds:        0.1 100 100 100 100 100 100 100 100
32 Include dirs:
33 User annotated:    /root/CARPETA/tp2-2020-2q-src/main.c
34 Auto-annotation:   off
35
36 -----
37 Ir           I1mr  ILmr  Dr           D1mr   DLmr  Dw           D1mw
      DLmw
38 -----
39 147,337,268 2,404 2,384 52,568,667 25,352 2,949 21,039,786
      1,329,573 1,311,308 PROGRAM TOTALS
40 -----
41 -----
42 Ir           I1mr  ILmr  Dr           D1mr   DLmr  Dw           D1mw
      DLmw      file:function
43 -----
44 146,822,412 29 29 52,439,429 20,532 52 20,973,042
      1,328,655 1,310,735 /root/CARPETA/tp2-2020-2q-src/main.c:
      matrix_multiply
45 -----
46 -----

```

```

47 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
48 -----
49 Ir          I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
          DLmw
50
51 -- line 18 -----
52 . . . . . matrix_t* create_matrix(size_t rows,
          size_t cols);
53 . . . . .
54 . . . . . void destroy_matrix(matrix_t* m);
55 . . . . .
56 . . . . . int print_matrix(FILE* fp, matrix_t*
          m);
57 . . . . .
58 . . . . . int
59 . . . . . main(int argc, char** argv)
60 10 2 2 0 0 0 0 5
          0 {
61 . . . . . /* matrixes */
62 . . . . . matrix_t *a, *b, *c;
63 . . . . . /* n (dimension) and block size */
64 . . . . . size_t n, bs;
65 . . . . . /* line buffer (init to null to
          simplify freeing on error) */
66 1 0 0 0 0 0 0 1
          0 char *line = NULL;
67 . . . . . /* line parsing auxiliar pointers
          */
68 . . . . . char *nptr, *endptr;
69 . . . . . /* auxiliar variables */
70 . . . . . long l;
71 . . . . . double e;
72 2 1 1 0 0 0 0 1
          0 size_t lineno = 1;
73 . . . . .

```

```

74      .      .      .      struct timespec t0;
75      .      .      .      struct timespec t1;
76      .      .      .      double dt;
77      .      .      .      size_t i;
78      25      4      4      9      0      0      1
79      0      1      1      0      for(, !feof(stdin); lineno++) {
80      10      1      1      4      0      0      6
81      0      0      0      a=b=c=NULL;
82      .      .      .      .      .      .
83      18      3      3      8      1      0      2
84      0      0      0      line = read_line(stdin);
85      .      .      .      .      .      .
86      6      0      0      2      0      0      0
87      0      0      0      if (!line) goto _exit_main;
88      9      0      0      4      0      0      0
89      0      0      0      if (line[0] == 0) break;
90      .      .      .      .      .      .
91      .      .      .      .      .      .
92      .      .      .      .      .      .
93      2      1      1      1      0      0      1
94      1      1      0      nptr = line;
95      10      1      1      3      1      0      1
96      0      1      0      l = strtol(nptr, &endptr, 10);
97      8      1      1      3      0      0      0
98      0      0      0      if (errno) {
99      .      .      .      .      .      .
      .      .      .      perror("");
      .      .      .      .      .      .
      .      .      .      goto _exit_main;
      .      .      .      .      .      .
      .      .      .      }
      4      2      2      2      0      0      0
      0      0      0      if (nptr == endptr) {
      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "missing
dimension");
      .      .      .      .      .      .
      .      .      .      goto _exit_main;
      .      .      .      .      .      .
      .      .      .      }
      3      2      2      1      0      0      0
      0      0      0      if (l < 1) {
      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "invalid
dimension");
      .      .      .      .      .      .

```

```

100         .         .         goto _exit_main;
101         .         .         }
102         2         1         1         1         0         0         1
103         0         0         n = (size_t) 1;
104         .         .         #if 0
105         .         .         /* parse block size */
106         .         .         nptr = endptr;
107         .         .         l = strtol(nptr, &endptr, 10);
108         .         .         if (errno) {
109         .         .             perror("");
110         .         .             goto _exit_main;
111         .         .         }
112         -- line 75 -----
113         -- line 83 -----
114         .         .         }
115         .         .         bs = (size_t) 1;
116         .         .         if (n % bs) {
117         .         .             fprintf(stderr, "block size
118         .         .             doesn't match");
119         .         .             goto _exit_main;
120         .         .         }
121         .         .         #else
122         2         0         0         1         0         0         1
123         0         0         bs = n;
124         .         .         #endif
125         .         .         /* load matrix a */
126         11         1         1         5         1         0         1
127         0         0         if (!(a = create_matrix(n, n)))
128         .         .             goto _exit_main;
129         .         .

```

```

127      650      3      3      259      0      0      65
        0      0      0      for (i=0; i < n*n; i++) {
128      128      0      0      64      0      0      64
        0      0      0      nptr = endptr;
129      576      1      1      192      0      0      64
        0      0      0      e = strtod(nptr, &endptr);
130      512      1      1      192      0      0      0
        0      0      0      if (errno) {
131      .      .      .      .      .      .      .
        .      .      .      perror("");
132      .      .      .      .      .      .      .
        .      .      .      goto _exit_main;
133      .      .      .      .      .      .      .
        .      .      .      }
134      256      2      2      128      0      0      0
        0      0      0      if (nptr == endptr) {
135      .      .      .      .      .      .      .
        .      .      .      fprintf(stderr, "missing A
        matrix element");
136      .      .      .      .      .      .      .
        .      .      .      goto _exit_main;
137      .      .      .      .      .      .      .
        .      .      .      }
138      448      1      1      256      0      0      64
        15      15      a->array[i] = e;
139      .      .      .      .      .      .      .
        .      .      .      }
140      .      .      .      .      .      .      .
        .      .      .      .
141      .      .      .      .      .      .      .
        .      .      .      /* load matrix b */
142      11      1      1      5      0      0      1
        0      0      0      if (!(b = create_matrix(n, n)))
143      .      .      .      .      .      .      .
        .      .      .      goto _exit_main;
144      .      .      .      .      .      .      .
        .      .      .      .
145      650      2      2      259      0      0      65
        0      0      0      for (i=0; i < n*n; i++) {
146      128      1      1      64      0      0      64
        0      0      0      nptr = endptr;
147      576      1      1      192      0      0      64
        0      0      0      e = strtod(nptr, &endptr);
148      512      1      1      192      0      0      0
        0      0      0      if (errno) {
149      .      .      .      .      .      .      .
        .      .      .      perror("");
150      .      .      .      .      .      .      .
        .      .      .      goto _exit_main;
151      .      .      .      .      .      .      .
        .      .      .      }
152      256      1      1      128      0      0      0
        0      0      0      if (nptr == endptr) {
153      .      .      .      .      .      .      .

```

```

154         .           .           fprintf(stderr, "missing B
matrix element");
155         .           .           goto _exit_main;
156         .           .           }
448         1       1       256       0       0       64
16       16       b->array[i] = e;
157         .           .           }
158         .           .           .
159         8       1       1       2       0       0       0
0       0       clock_gettime(CLOCK_REALTIME, &t0
);
160         .           .           .
161         .           .           .
162         13      2       2       6       1       1       1
0       0       /* multiply matrixes */
if (!(c = matrix_multiply(a, b, bs
)))
163         .           .           .
goto _exit_main;
164         .           .           .
165         8       1       1       2       0       0       0
0       0       clock_gettime(CLOCK_REALTIME, &t1
);
166         .           .           .
167         7       1       1       2       1       1       1
1       1       dt = (float) (t1.tv_sec - t0.
tv_sec);
168         12      1       1       5       2       2       1
0       0       dt = dt + ((float)(t1.tv_nsec - t0
.tv_nsec)) / 1.0e9;
169         .           .           .
170         13      3       2       5       2       2       0
0       0       if(print_matrix(stdout, c) == -1)
171         .           .           .
goto _exit_main;
172         .           .           .
173         12      2       1       7       0       0       0
0       0       fprintf(stderr, "time: %g\n", dt);
174         .           .           .
175         6       1       1       3       0       0       0
0       0       free(line);
176         6       1       1       3       0       0       0
0       0       destroy_matrix(a);
177         6       1       1       3       0       0       0

```



```

178         0      0      destroy_matrix(b);
179         6      0      0      3      0      0      0
180         0      0      destroy_matrix(c);
181         .      .      .      .      .      .      .
182         .      .      .      }
183         .      .      .      .      .      .      .
184         .      .      .      .      .      .      .
185         3      1      1      0      0      0      0
186         0      0      return 0;
187         .      .      .      .      .      .      .
188         .      .      .      .      .      .      .
189         .      .      .      _exit_main:
190         .      .      .      .      .      .      .
191         .      .      .      fprintf(stderr, " at line %u\n", (
192         unsigned) lineno);
193         .      .      .      .      .      .      .
194         .      .      .      free(line);
195         .      .      .      .      .      .      .
196         .      .      .      destroy_matrix(a);
197         .      .      .      .      .      .      .
198         .      .      .      destroy_matrix(b);
199         .      .      .      .      .      .      .
200         .      .      .      destroy_matrix(c);
201         .      .      .      .      .      .      .
202         .      .      .      exit(1);
203         6      1      1      2      0      0      0
204         0      0      }
205         .      .      .      .      .      .      .
206         .      .      .      .      .      .      .
207         .      .      .      matrix_t*
208         .      .      .      .      .      .      .
209         .      .      .      matrix_multiply(matrix_t* m1,
210         matrix_t* m2, int bs)
211         11      2      2      0      0      0      6
212         0      0      {
213         .      .      .      .      .      .      .
214         .      .      .      size_t n, en, i, j, k, kk, jj;
215         .      .      .      .      .      .      .
216         .      .      .      double sum;
217         .      .      .      .      .      .      .
218         .      .      .      matrix_t* mr;
219         .      .      .      .      .      .      .
220         .      .      .      .      .      .      .
221         3      1      1      2      0      0      1
222         0      0      n = m1->rows;
223         .      .      .      .      .      .      .
224         .      .      .      .      .      .      .
225         11      1      1      5      0      0      1
226         0      0      if(!(mr = create_matrix(n,n)))
227         return NULL;
228         .      .      .      .      .      .      .
229         .      .      .      .      .      .      .

```

```

203      9      2      2      3      0      0      1
          0      0      en = bs*(n/bs);
204      .      .      .      .      .      .      .
          .      .      .
205      72      2      2      26      0      0      9
          0      0      for(i=0; i<n; i++)
206      576      1      1      208      0      0      72
          0      0      for(j=0; j<n; j++)
207      704      1      1      320      0      0      128
          15      15      mr->array[i*n+j] = 0.0;
208      .      .      .      .      .      .      .
          .      .      .
209      .      .      .      .      .      .      .
          .      .      .      #if 1
210      .      .      .      .      .      .      .
          .      .      .      if (1) {
211      .      .      .      .      .      .      .
          .      .      .      size_t j;
212      2      0      0      0      0      0      1
          0      0      size_t dim = 1024*1024*10;
213      9      1      1      3      0      0      1
          0      0      int *v = malloc(dim*sizeof(int));
214      83,886,088      2      2      31,457,282      20,480      0      10,485,761
          0      0      for (j = 0; j < dim; ++j)
215      62,914,560      0      0      20,971,520      0      0      10,485,760
          1,328,640      1,310,720      v[j] = -1;
216      6      1      1      3      2      2      0
          0      0      free(v);
217      .      .      .      .      .      .      .
          .      .      .      }
218      .      .      .      .      .      .      .
          .      .      .      #endif
219      .      .      .      .      .      .      .
          .      .      .
220      17      2      2      6      0      0      2
          0      0      for(kk=0; kk<en; kk+=bs)
221      17      2      2      6      0      0      2
          0      0      for(jj=0; jj<en; jj+=bs)
222      72      1      1      26      0      0      9
          0      0      for(i=0; i<n; i++)
223      728      2      2      288      0      0      72
          0      0      for(j=jj; j<jj+bs; j++) {
224      704      1      1      384      17      17      64
          0      0      sum = mr->array[i*n+j];
225      5,824      3      3      2,304      0      0      576
          0      0      for(k=kk; k<kk+bs; k++)
226      12,288      2      2      6,656      33      33      512
          0      0      sum += m1->array[i*n+k] * m2->
          array[k*n+j];
227      704      1      1      384      0      0      64
          0      0      mr->array[i*n+j] = sum;
228      .      .      .      .      .      .      .
          .      .      .      }
229      1      0      0      1      0      0      0

```

```

230         0      0      0      0      0      0      0      0      0      0
        6      1      1      2      0      0      0      0
        0      0      0      0      0      0      0      0
231         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
232         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
233         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
234        18     3      3      0      0      0      0      8
        0      0      0      0      0      0      0      0
235         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
236         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
237         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
238        6      0      0      0      0      0      0      4
        0      0      0      0      0      0      0      0
239         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
240         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
241        14     1      1      6      1      0      0      2
        1      0      0      0      0      0      0      0
242        6      0      0      2      0      0      0      0
        0      0      0      0      0      0      0      0
243         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
244         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
245         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
246         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
247        3,881   5      5      1,294   1      0      259
        0      0      0      0      0      0      0      0
        ' ) {
248        2,056   2      2      771      0      0      514
        8      8      8      8      8      8      8      8
249        1,285   0      0      514      0      0      0
        0      0      0      0      0      0      0      0
250         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
251         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
252         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
253         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
254         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
255         .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .

```

```

256     .   .   .   .   .   .   .   .
257     .   .   .   .   .   .   .   .
258     8   1   1   .   2   0   0   0
259         0   0   0   if (c != EOF)
260         7   0   0   2   0   0   2
261         0   0   0   str[len++]='\n';
262     .   .   .   .   .   .   .   .
263     12  1   1   .   4   0   0   4
264         0   0   0   str[len++]='\0';
265     2   0   0   2   0   0   0
266         0   0   0   return str;
267     12  1   1   4   0   0   0
268         0   0   0 }
269     .   .   .   .   .   .   .   .
270     .   .   .   .   .   .   .   .
271     .   .   .   .   .   .   .   .
272     .   .   .   .   .   .   .   .
273     .   .   .   .   .   .   .   .
274     .   .   .   .   .   .   .   .
275     .   .   .   .   .   .   .   .
276     .   .   .   .   .   .   .   .
277     .   .   .   .   .   .   .   .
278     .   .   .   .   .   .   .   .
279     .   .   .   .   .   .   .   .
280     .   .   .   .   .   .   .   .
281     .   .   .   .   .   .   .   .

```

```

282         .      .      .      .      .      .      .      .      .
283         .      .      .      .      .      .      .      .      .
284         3      1      1      .      3      0      0      .      0
285         .      0      .      0      return m;
286         18     1      1      .      6      0      0      .      0
287         .      0      .      0      }
288         .      .      .      .      .      .      .      .      .
289         .      .      .      .      void
290         .      .      .      .      .      .      .      .      .
291         .      .      .      .      destroy_matrix(matrix_t* m)
292         27     1      1      .      0      0      0      .      12
293         .      0      .      0      {
294         9      0      0      .      3      0      0      .      0
295         .      0      .      0      if (!m) return;
296         24     1      1      .      12     0      0      .      0
297         .      0      .      0      free(m->array);
298         24     1      1      .      9      0      0      .      0
299         .      0      .      0      free(m);
300         18     0      0      .      6      0      0      .      0
301         .      0      .      0      }
302         .      .      .      .      .      .      .      .      .
303         .      .      .      .      .      .      .      .      .
304         .      .      .      .      int
305         .      .      .      .      .      .      .      .      .
306         .      .      .      .      print_matrix(FILE* fp, matrix_t* m)
307         10     2      2      .      0      0      0      .      5
308         .      0      .      0      {
309         .      .      .      .      .      .      .      .      .
310         .      .      .      .      size_t i, j;
311         .      .      .      .      .      .      .      .      .
312         .      .      .      .      size_t n;
313         3      1      1      .      2      0      0      .      1
314         .      0      .      0      n = m->rows;
315         13     1      1      .      6      1      1      .      0
316         .      0      .      0      if (fprintf(fp, "%lu", (unsigned
317         long) m->rows) < 0) {
318         .      .      .      .      .      .      .      .      .
319         .      .      .      .      perror("");
320         .      .      .      .      .      .      .      .      .
321         .      .      .      .      return -1;
322         .      .      .      .      .      .      .      .      .
323         .      .      .      .      }
324         72     2      2      .      26     0      0      .      9
325         .      0      .      0      for(i=0; i<n; i++)
326         576    2      2      .      208    0      0      .      72
327         .      0      .      0      for (j=0; j<n; j++)
328         1,408   3      3      .      640    0      0      .      0
329         .      0      .      0      if (fprintf(fp, " %g", m->array[i*n

```

```

308         . . . . .
309         . . . . . perror("");
310         . . . . . return -1;
311         . . . . . }
312     10 1 1 4 0 0 0
313         0 0 if (fprintf(fp, "\n") < 0) {
314         . . . . . perror("");
315         . . . . . return -1;
316         . . . . . }
317     1 1 1 0 0 0 0
318         0 0 return 0;
319     6 1 1 2 0 0 0
320         0 0 }
321
-----
322 Ir          IImr ILMr Dr          Dimr    DLMr Dw
323         DLMw
324
-----
325 146,837,450 121 119 52,445,273 20,545 59 20,974,500
326 1,328,697 1,310,775 events annotated

```

6.2. 2WSA:

```

1 1 30
2 == Cachegrind, a cache and branch-prediction profiler
3 ==626== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
    Nethercote et al.
4 ==626== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
5 ==626== Command: /tmp/02-mmult
6 ==626== Parent PID: 597
7 ==626==
8 --626-- Warning: Cannot auto-detect cache config, using
    defaults.
9 --626--          Run with -v to see.
10 ==626==
11 ==626== I    refs:          147,098,977
12 ==626== I1  misses:          2,390
13 ==626== LLi misses:          2,372
14 ==626== I1  miss rate:         0.00%
15 ==626== LLi miss rate:         0.00%
16 ==626==
17 ==626== D    refs:          73,518,213   (52,506,759 rd   +
    21,011,454 wr)

```

```

18 ==626== D1 misses:      1,315,006 (    3,709 rd  +
    1,311,297 wr)
19 ==626== LLd misses:    1,314,138 (    2,894 rd  +
    1,311,244 wr)
20 ==626== D1 miss rate:      1.8% (    0.0%  +
    6.2%  )
21 ==626== LLd miss rate:    1.8% (    0.0%  +
    6.2%  )
22 ==626==
23 ==626== LL refs:        1,317,396 (    6,099 rd  +
    1,311,297 wr)
24 ==626== LL misses:      1,316,510 (    5,266 rd  +
    1,311,244 wr)
25 ==626== LL miss rate:      0.6% (    0.0%  +
    6.2%  )
26 -----
27 I1 cache:               32768 B, 32 B, 4-way associative
28 D1 cache:               32768 B, 32 B, 2-way associative
29 LL cache:               524288 B, 32 B, 8-way associative
30 Command:                /tmp/02-mmult
31 Data file:              cachegrind.out.626
32 Events recorded: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:           0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:         /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation:       off
39
40 -----
41 Ir          I1mr I1Lmr Dr          D1mr  DLmr Dw          D1mw
    DLmw
42 -----
43 147,098,977 2,390 2,372 52,506,759 3,709 2,894 21,011,454
    1,311,297 1,311,244 PROGRAM TOTALS
44
45 -----
46 Ir          I1mr I1Lmr Dr          D1mr  DLmr Dw          D1mw
    DLmw          file:function
47 -----
48 146,800,887 29 29 52,428,894 5 5 20,971,551
    1,310,720 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
    matrix_multiply
49
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----

```

```

53 Ir          I1mr ILmr Dr          Dimr DLmr Dw          Dimw
    DLmw
54
55 -- line 18 -----
56 . . . . .
    . matrix_t* create_matrix(size_t rows,
    size_t cols);
57 . . . . .
    .
58 . . . . .
    void destroy_matrix(matrix_t* m);
59 . . . . .
    .
60 . . . . .
    int print_matrix(FILE* fp, matrix_t* m)
    ;
61 . . . . .
    .
62 . . . . .
    int
63 . . . . .
    main(int argc, char** argv)
64 10 2 2 0 0 0 5 0
    {
65 . . . . .
    /* matrixes */
66 . . . . .
    matrix_t *a, *b, *c;
67 . . . . .
    /* n (dimension) and block size */
68 . . . . .
    size_t n, bs;
69 . . . . .
    /* line buffer (init to null to
    simplify freeing on error) */
70 1 0 0 0 0 0 1 0
    char *line = NULL;
71 . . . . .
    /* line parsing auxiliar pointers */
72 . . . . .
    char *nptr, *endptr;
73 . . . . .
    /* auxiliar variables */
74 . . . . .
    long l;
75 . . . . .
    double e;
76 2 1 1 0 0 0 1 0
    size_t lineno = 1;
77 . . . . .
    struct timespec t0;
78 . . . . .
    struct timespec t1;

```



```

79      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .
82      25      4      4      .      9      0      0      1      0
83      10      1      1      .      4      0      0      6      0
84      .      .      .      .      .      .      .      .
85      18      3      3      .      8      1      0      2      0
86      .      .      .      .      .      .      .      .
87      6      0      0      .      2      0      0      0      0
88      9      0      0      .      4      0      0      0      0
89      .      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .      .
91      2      1      1      .      1      0      0      1      1
92      10      1      1      .      3      1      0      1      0
93      8      1      1      .      3      0      0      0      0
94      .      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .      .
97      4      2      2      .      2      0      0      0      0
98      .      .      .      .      .      .      .      .
99      .      .      .      .      .      .      .      .
100      .      .      .      .      .      .      .      .
101      3      2      2      .      1      0      0      0      0
102      .      .      .      .      .      .      .      .
103      .      .      .      .      .      .      .      .
104      .      .      .      .      .      .      .      .

      double dt;

      size_t i;

      for(; !feof(stdin); lineno++) {
          a=b=c=NULL;

          line = read_line(stdin);

          if (!line) goto _exit_main;
          if (line[0] == 0) break;

          /* parse dimension */
          nptr = line;
          l = strtol(nptr, &endptr, 10);
          if (errno) {
              perror("");
              goto _exit_main;
          }
          if (nptr == endptr) {
              fprintf(stderr, "missing
dimension");
              goto _exit_main;
          }
          if (l < 1) {
              fprintf(stderr, "invalid
dimension");
              goto _exit_main;
          }
      }

```

```

105      2      1      1          1      0      0          1      0
      0          n = (size_t) 1;
106      .      .      .      .      .      .      .      .
      .      .      .      #if 0
107      .      .      .      .      .      .      .      .
      .      .      .      /* parse block size */
108      .      .      .      .      .      .      .      .
      .      .      .      nptr = endptr;
109      .      .      .      .      .      .      .      .
      .      .      .      l = strtol(nptr, &endptr, 10);
110      .      .      .      .      .      .      .      .
      .      .      .      if (errno) {
111      .      .      .      .      .      .      .      .
      .      .      .      perror("");
112      .      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
113      .      .      .      .      .      .      .      .
      .      .      .      }
114  -- line 75 -----
115  -- line 83 -----
116      .      .      .      .      .      .      .      .
      .      .      .      }
117      .      .      .      .      .      .      .      .
      .      .      .      bs = (size_t) 1;
118      .      .      .      .      .      .      .      .
      .      .      .      .
119      .      .      .      .
      .      .      .      if (n % bs) {
120      .      .      .      .      .      .      .      .
      .      .      .      fprintf(stderr, "block size doesn
't match");
121      .      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
122      .      .      .      .      .      .      .      .
      .      .      .      }
123      .      .      .      .      .      .      .      .
      .      .      .      #else
124      2      0      0          1      0      0          1      0
      0          bs = n;
125      .      .      .      .      .      .      .      .
      .      .      .      #endif
126      .      .      .      .      .      .      .      .
      .      .      .      .
127      .      .      .      .      .      .      .      .
      .      .      .      /* load matrix a */
128      11      1      1          5      1      0          1      0
      0          if (!(a = create_matrix(n, n)))
129      .      .      .      .      .      .      .      .
      .      .      .      goto _exit_main;
130      .      .      .      .      .      .      .      .
      .      .      .      .
131      20      3      3          7      0      0          2      0
      0          for (i=0; i < n*n; i++) {

```

```

132      2    0    0      1    0    0      1    0
      0      nptr = endptr;
133      9    1    1      3    0    0      1    0
      0      e = strtod(nptr, &endptr);
134      8    1    1      3    0    0      0    0
      0      if (errno) {
135      .    .    .      .    .    .      .    .
      .      perror("");
136      .    .    .      .    .    .      .    .
      .      goto _exit_main;
137      .    .    .      .    .    .      .    .
      .      }
138      4    2    2      2    0    0      0    0
      0      if (nptr == endptr) {
139      .    .    .      .    .    .      .    .
      .      fprintf(stderr, "missing A
matrix element");
140      .    .    .      .    .    .      .    .
      .      goto _exit_main;
141      .    .    .      .    .    .      .    .
      .      }
142      7    1    1      4    0    0      1    0
      0      a->array[i] = e;
143      .    .    .      .    .    .      .    .
      .      }
144      .    .    .      .    .    .      .    .
      .
145      .    .    .      .    .    .      .    .
      .      /* load matrix b */
146      11   1    1      5    0    0      1    0
      0      if (!(b = create_matrix(n, n)))
147      .    .    .      .    .    .      .    .
      .      goto _exit_main;
148      .    .    .      .    .    .      .    .
      .
149      20   2    2      7    0    0      2    0
      0      for (i=0; i < n*n; i++) {
150      2    1    1      1    0    0      1    0
      0      nptr = endptr;
151      9    1    1      3    0    0      1    0
      0      e = strtod(nptr, &endptr);
152      8    1    1      3    0    0      0    0
      0      if (errno) {
153      .    .    .      .    .    .      .    .
      .      perror("");
154      .    .    .      .    .    .      .    .
      .      goto _exit_main;
155      .    .    .      .    .    .      .    .
      .      }
156      4    1    1      2    0    0      0    0
      0      if (nptr == endptr) {
157      .    .    .      .    .    .      .    .
      .      fprintf(stderr, "missing B
matrix element");

```

```

158      .      .      .      .      .      .      .      .
159      .      .      .      .      .      .      .      .
160      7      1      1      .      4      0      0      1      0
161      .      .      .      .      .      .      .      .
162      .      .      .      .      .      .      .      .
163      8      1      1      .      2      0      0      0      0
164      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
166      13     2      2      .      6      1      1      1      0
167      .      .      .      .      .      .      .      .
168      .      .      .      .      .      .      .      .
169      8      1      1      .      2      0      0      0      0
170      .      .      .      .      .      .      .      .
171      7      1      1      .      2      1      1      1      1
172      .      .      .      .      .      .      .      .
173      .      .      .      .      .      .      .      .
174      13     3      2      .      5      2      2      0      0
175      .      .      .      .      .      .      .      .
176      .      .      .      .      .      .      .      .
177      12     1      1      .      7      0      0      0      0
178      .      .      .      .      .      .      .      .
179      6      1      1      .      3      0      0      0      0
180      6      1      1      .      3      0      0      0      0
181      6      1      1      .      3      0      0      0      0
182      6      0      0      .      3      0      0      0      0

```

```

183     . . . . } . . . .
184     . . . . . . . . . .
185     3 1 1 0 0 0 0 0
186         0 return 0;
187     . . . . . . . . . .
188     . . . . _exit_main:
189     . . . . fprintf(stderr, " at line %u\n", (
190         unsigned) lineno);
191     . . . . free(line);
192     . . . . destroy_matrix(a);
193     . . . . destroy_matrix(b);
194     . . . . destroy_matrix(c);
195     . . . . exit(1);
196     6 1 1 2 0 0 0 0
197         0 }
198     . . . . matrix_t*
199     . . . . matrix_multiply(matrix_t* m1, matrix_t*
200         m2, int bs)
201     11 2 2 0 0 0 6 0
202         0 {
203     . . . . size_t n, en, i, j, k, kk, jj;
204     . . . . double sum;
205     . . . . matrix_t* mr;
206     . . . .
207     3 1 1 2 0 0 1 0
208         0 n = m1->rows;
209     . . . .
210     11 1 1 5 0 0 1 0
211         0 if(!(mr = create_matrix(n,n))) return
212         NULL;
213     . . . .
214     9 2 2 3 0 0 1 0
215         0 en = bs*(n/bs);

```

```

208      .      .      .      .      .      .      .      .
209      16      2      2      .      5      0      0      2      0
                0      for(i=0; i<n; i++)
210      16      1      1      .      5      0      0      2      0
                0      for(j=0; j<n; j++)
211      11      1      1      .      5      0      0      2      0
                0      mr->array[i*n+j] = 0.0;
212      .      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .      .
214      .      .      .      .      .      .      .      .
                .      #if 1
215      .      .      .      .      .      .      .      .
                .      if (1) {
216      2      0      0      .      0      0      0      1      0
                0      size_t j;
                size_t dim = 1024*1024*10;
217      9      1      1      .      3      0      0      1      0
                0      int *v = malloc(dim*sizeof(int));
218      83,886,088      2      2      31,457,282      0      0      10,485,761      0
                0      for (j = 0; j < dim; ++j)
219      62,914,560      0      0      20,971,520      0      0      10,485,760      1,310,720
                1,310,720      v[j] = -1;
220      6      1      1      .      3      2      2      0      0
                0      free(v);
221      .      .      .      .      .      .      .      .
                .      }
222      .      .      .      .      .      .      .      .
                .      #endif
223      .      .      .      .      .      .      .      .
224      17      2      2      .      6      0      0      2      0
                0      for(kk=0; kk<en; kk+=bs)
225      17      2      2      .      6      0      0      2      0
                0      for(jj=0; jj<en; jj+=bs)
226      16      1      1      .      5      0      0      2      0
                0      for(i=0; i<n; i++)
227      21      2      2      .      8      0      0      2      0
                0      for(j=jj; j<jj+bs; j++) {
228      11      1      1      .      6      1      1      1      0
                0      sum = mr->array[i*n+j];
229      21      3      3      .      8      0      0      2      0
                0      for(k=kk; k<kk+bs; k++)
230      24      2      2      .      13      2      2      1      0
                0      sum += m1->array[i*n+k] * m2
                ->array[k*n+j];
231      11      1      1      .      6      0      0      1      0
                0      mr->array[i*n+j] = sum;
232      .      .      .      .      .      .      .      .
                .      }
233      1      0      0      .      1      0      0      0      0
                0      return mr;

```

```

234         6   1   1           2   0   0           0   0
           0 }
235     .   .   .           .   .   .           .   .
236     .   .   .           .   .   .           .   .
           . char*
237     .   .   .           .   .   .           .   .
           . read_line(FILE *fp)
238     18   3   3           0   0   0           8   0
           0 {
239     .   .   .           .   .   .           .   .
           . #define DEF_LINE_SZ 1024
240     .   .   .           .   .   .           .   .
           .
241     .   .   .           .   .   .           .   .
           . int c;
242     6   0   0           0   0   0           4   0
           0 size_t len = 0, tam = DEF_LINE_SZ;
243     .   .   .           .   .   .           .   .
           . char* str;
244     .   .   .           .   .   .           .   .
           .
245     14   1   1           6   1   0           2   1
           0 str = malloc(tam);
246     6   0   0           2   0   0           0   0
           0 if (!str) {
247     .   .   .           .   .   .           .   .
           . perror("");
248     .   .   .           .   .   .           .   .
           . return NULL;
249     .   .   .           .   .   .           .   .
           . }
250     .   .   .           .   .   .           .   .
           .
251     101  5   5           34   1   0           7   0
           0 while (EOF != (c=fgetc(fp)) && c != '\n
           ') {
252     40   2   2           15   0   0           10   0
           0 str[len++]=c;
253     25   0   0           10   0   0           0   0
           0 if (len==tam-1) {
254     .   .   .           .   .   .           .   .
           . str = realloc(str, tam *= 2);
255     .   .   .           .   .   .           .   .
           . if (!str) {
256     .   .   .           .   .   .           .   .
           . perror("");
257     .   .   .           .   .   .           .   .
           . return NULL;
258     .   .   .           .   .   .           .   .
           . }
259     .   .   .           .   .   .           .   .
           . }

```

```

260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .
262      8      1      1      2      0      0      0      0
263      7      0      0      2      0      0      2      0
264      .      .      .      .      .      .      .      .
265      12     1      1      4      0      0      4      0
266      2      0      0      2      0      0      0      0
267      12     1      1      4      0      0      0      0
268      .      .      .      .      .      .      .      .
269      .      .      .      .      .      .      .      .
270      .      .      .      .      .      .      .      .
271      .      .      .      .      .      .      .      .
272      30     1      1      0      0      0      15     0
273      .      .      .      .      .      .      .      .
274      .      .      .      .      .      .      .      .
275      30     2      2      9      1      0      3      0
276      .      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .      .
278      .      .      .      .      .      .      .      .
279      .      .      .      .      .      .      .      .
280      9      1      1      6      0      0      3      0
281      9      1      1      6      0      0      3      0
282      51     2      2      21     0      0      3      0
283      .      .      .      .      .      .      .      .
284      .      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .      .

```



```

286         .      .      .      .      .      .      .      .
287         .      .      .      .      .      .      .      .
288         3      1      1      .      3      0      0      0      0
289         .      .      .      .      .      .      .      .
290         .      .      .      .      .      .      .      .
291         .      .      .      .      .      .      .      .
292         .      .      .      .      .      .      .      .
293         27     1      1      .      0      0      0      12      0
294         9      0      0      .      3      0      0      0      0
295         24     1      1      .      12     0      0      0      0
296         24     1      1      .      9      0      0      0      0
297         18     0      0      .      6      0      0      0      0
298         .      .      .      .      .      .      .      .
299         .      .      .      .      .      .      .      .
300         .      .      .      .      .      .      .      .
301         10     2      2      .      0      0      0      5      0
302         .      .      .      .      .      .      .      .
303         .      .      .      .      .      .      .      .
304         3      1      1      .      2      0      0      1      0
305         13     1      1      .      6      1      1      0      0
306         .      .      .      .      .      .      .      .
307         .      .      .      .      .      .      .      .
308         .      .      .      .      .      .      .      .
309         16     2      2      .      5      0      0      2      0
310         16     2      2      .      5      0      0      2      0
311         22     3      3      .      10     0      0      0      0

```

```

312         . . . . . perror("");
313         . . . . . return -1;
314         . . . . . }
315         10 1 1 4 0 0 0 0
316         . . . . . if (fprintf(fp, "\n") < 0) {
317         . . . . . perror("");
318         . . . . . return -1;
319         . . . . . }
320         1 1 1 0 0 0 0 0
321         . . . . . return 0;
322         6 1 1 2 0 0 0 0
323         . . . . . }
324
325 -----
326 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw
327          DLmw
328 -----
329 146,801,827 120 119 52,429,222 18 12 20,971,672 1,310,723
330 1,310,721 events annotated

```

```

1 2 81 112 14 18
2 ind, a cache and branch-prediction profiler
3 ==629== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
4   Nethercote et al.
5 ==629== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
6   copyright info
7 ==629== Command: /tmp/02-mmult
8 ==629== Parent PID: 597
9 ==629==
10 --629-- Warning: Cannot auto-detect cache config, using
11 defaults.
12 --629-- Run with -v to see.
13 ==629==
14 ==629== I   refs:      147,109,392
15 ==629== I1  misses:      2,408
16 ==629== L1i misses:      2,388
17 ==629== I1  miss rate:    0.00%
18 ==629== L1i miss rate:    0.00%
19 ==629==
20 ==629== D   refs:      73,522,048 (52,509,341 rd +
21   21,012,707 wr)
22 ==629== D1  misses:      1,314,921 ( 3,621 rd +
23   1,311,300 wr)
24 ==629== L1d misses:      1,314,145 ( 2,898 rd +
25   1,311,247 wr)
26 ==629== D1  miss rate:    1.8% ( 0.0% +

```

```

21      6.2% )
==629== LLd miss rate:          1.8% (          0.0%  +
      6.2% )
22 ==629==
23 ==629== LL refs:             1,317,329 (          6,029 rd  +
      1,311,300 wr)
24 ==629== LL misses:          1,316,533 (          5,286 rd  +
      1,311,247 wr)
25 ==629== LL miss rate:          0.6% (          0.0%  +
      6.2% )
26 -----
27 I1 cache:                    32768 B, 32 B, 4-way associative
28 D1 cache:                    32768 B, 32 B, 2-way associative
29 LL cache:                    524288 B, 32 B, 8-way associative
30 Command:                     /tmp/02-mmult
31 Data file:                    cachegrind.out.629
32 Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds: 0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
39
40 -----
41 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw
42 -----
43 147,109,392 2,408 2,388 52,509,341 3,621 2,898 21,012,707
      1,311,300 1,311,247 PROGRAM TOTALS
44 -----
45 -----
46 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw          file:function
47 -----
48 146,801,346 29 29 52,429,109 7 7 20,971,590
      1,310,720 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
      matrix_multiply
49 -----
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw
54 -----
55 -- line 18 -----
56 . . . . .

```

```

        .   matrix_t* create_matrix(size_t rows,
        .   size_t cols);
57      .   .   .   .   .   .   .   .
58      .   .   .   .   .   .   .   .
        .   void destroy_matrix(matrix_t* m);
59      .   .   .   .   .   .   .   .
        .   .   .   .   .   .   .   .
60      .   .   .   .   .   .   .   .
        .   int print_matrix(FILE* fp, matrix_t* m)
        .   ;
61      .   .   .   .   .   .   .   .
62      .   .   .   .   .   .   .   .
        .   int
63      .   .   .   .   .   .   .   .
        .   main(int argc, char** argv)
64      10   2   2   0   0   0   5   0
        .   0 {
65      .   .   .   .   .   .   .   .
        .   .   /* matrixes */
66      .   .   .   .   .   .   .   .
        .   .   matrix_t *a, *b, *c;
67      .   .   .   .   .   .   .   .
        .   .   /* n (dimension) and block size */
68      .   .   .   .   .   .   .   .
        .   .   size_t n, bs;
69      .   .   .   .   .   .   .   .
        .   .   /* line buffer (init to null to
        .   .   simplify freeing on error) */
70      1   0   0   0   0   0   1   0
        .   0   char *line = NULL;
71      .   .   .   .   .   .   .   .
        .   .   /* line parsing auxiliar pointers */
72      .   .   .   .   .   .   .   .
        .   .   char *nptr, *endptr;
73      .   .   .   .   .   .   .   .
        .   .   /* auxiliar variables */
74      .   .   .   .   .   .   .   .
        .   .   long l;
75      .   .   .   .   .   .   .   .
        .   .   double e;
76      2   1   1   0   0   0   1   0
        .   0   size_t lineno = 1;
77      .   .   .   .   .   .   .   .
        .   .   struct timespec t0;
78      .   .   .   .   .   .   .   .
        .   .   struct timespec t1;
79      .   .   .   .   .   .   .   .
        .   .   double dt;
80      .   .   .   .   .   .   .   .
        .   .   size_t i;
81      .   .   .   .   .   .   .   .

```

```

82      25      4      4          9      0      0          1          0
      0      for(;; !feof(stdin); lineno++) {
83      10      1      1          4      0      0          6          0
      0      a=b=c=NULL;
84      .      .      .          .      .      .          .          .
      .
85      18      3      3          8      1      0          2          0
      0      line = read_line(stdin);
86      .      .      .          .      .      .          .          .
      .
87      6      0      0          2      0      0          0          0
      0      if (!line) goto _exit_main;
88      9      0      0          4      0      0          0          0
      0      if (line[0] == 0) break;
89      .      .      .          .      .      .          .          .
      .
90      .      .      .          .      .      .          .          .
      .
91      2      1      1          1      0      0          1          1
      0      nptr = line;
92      10      1      1          3      1      0          1          0
      0      l = strtol(nptr, &endptr, 10);
93      8      1      1          3      0      0          0          0
      0      if (errno) {
94      .      .      .          .      .      .          .          .
      .      perror("");
95      .      .      .          .      .      .          .          .
      .      goto _exit_main;
96      .      .      .          .      .      .          .          .
      .      }
97      4      2      2          2      0      0          0          0
      0      if (nptr == endptr) {
98      .      .      .          .      .      .          .          .
      .      fprintf(stderr, "missing
      dimension");
99      .      .      .          .      .      .          .          .
      .      goto _exit_main;
100     .      .      .          .      .      .          .          .
      .      }
101     3      2      2          1      0      0          0          0
      0      if (l < 1) {
102     .      .      .          .      .      .          .          .
      .      fprintf(stderr, "invalid
      dimension");
103     .      .      .          .      .      .          .          .
      .      goto _exit_main;
104     .      .      .          .      .      .          .          .
      .      }
105     2      1      1          1      0      0          1          0
      0      n = (size_t) l;
106     .      .      .          .      .      .          .          .
      .      #if 0
107     .      .      .          .      .      .          .          .
      .      /* parse block size */

```

```

108         .      .      .      .      .      .      .      .      .      .
109         .      .      .      .      .      .      .      .      .      .
110         .      .      .      .      .      .      .      .      .      .
111         .      .      .      .      .      .      .      .      .      .
112         .      .      .      .      .      .      .      .      .      .
113         .      .      .      .      .      .      .      .      .      .
114         .      .      .      .      .      .      .      .      .      .
115         .      .      .      .      .      .      .      .      .      .
116         .      .      .      .      .      .      .      .      .      .
117         .      .      .      .      .      .      .      .      .      .
118         .      .      .      .      .      .      .      .      .      .
119         .      .      .      .      .      .      .      .      .      .
120         .      .      .      .      .      .      .      .      .      .
121         .      .      .      .      .      .      .      .      .      .
122         .      .      .      .      .      .      .      .      .      .
123         .      .      .      .      .      .      .      .      .      .
124         .      .      .      .      .      .      .      .      .      .
125         .      .      .      .      .      .      .      .      .      .
126         .      .      .      .      .      .      .      .      .      .
127         .      .      .      .      .      .      .      .      .      .
128         .      .      .      .      .      .      .      .      .      .
129         .      .      .      .      .      .      .      .      .      .
130         .      .      .      .      .      .      .      .      .      .
131         .      .      .      .      .      .      .      .      .      .
132         .      .      .      .      .      .      .      .      .      .
133         .      .      .      .      .      .      .      .      .      .
134         .      .      .      .      .      .      .      .      .      .
135         .      .      .      .      .      .      .      .      .      .

        nptr = endptr;
        l = strtol(nptr, &endptr, 10);
        if (errno) {
            perror("");
            goto _exit_main;
        }

-- line 75 -----
-- line 83 -----

        }

        bs = (size_t) l;

        if (n % bs) {
            fprintf(stderr, "block size doesn't match");
            goto _exit_main;
        }

        #else
        2    0    0    1    0    0    1    0
        0        bs = n;
        #endif

        /* load matrix a */
11    1    1    5    1    0    1    0
        0        if (!(a = create_matrix(n, n)))
            goto _exit_main;

50    3    3    19    0    0    5    0
        0        for (i=0; i < n*n; i++) {
8    0    0    4    0    0    4    0
        0        nptr = endptr;
36    1    1    12    0    0    4    0
        0        e = strtod(nptr, &endptr);
32    1    1    12    0    0    0    0
        0        if (errno) {

```

```

136         perror("");
137         goto _exit_main;
138     }
139     16    2    2    8    0    0    0    0
140         if (nptr == endptr) {
141             fprintf(stderr, "missing A
matrix element");
142             goto _exit_main;
143         }
144     28    1    1    16    0    0    4    0
145         a->array[i] = e;
146     }
147     /* load matrix b */
148     11    1    1    5    0    0    1    0
149         if (!(b = create_matrix(n, n)))
150             goto _exit_main;
151     50    2    2    19    0    0    5    0
152         for (i=0; i < n*n; i++) {
153             8    1    1    4    0    0    4    0
154             nptr = endptr;
155             36    1    1    12    0    0    4    0
156             e = strtod(nptr, &endptr);
157             32    1    1    12    0    0    0    0
158             if (errno) {
159                 perror("");
160                 goto _exit_main;
161             }
162     16    1    1    8    0    0    0    0
163         if (nptr == endptr) {
164             fprintf(stderr, "missing B
matrix element");
165             goto _exit_main;
166         }
167     28    1    1    16    0    0    4    1
168         b->array[i] = e;
169     }

```

```

162         .      .      .      .      .      .      .      .      .      .
163         8      1      1      .      2      0      0      .      0      0
164         .      .      .      .      .      .      .      .      .      .
165         .      .      .      .      .      .      .      .      .      .
166         13     2      2      .      6      1      1      1      1      0
167         .      .      .      .      .      .      .      .      .      .
168         .      .      .      .      .      .      .      .      .      .
169         8      1      1      .      2      0      0      .      0      0
170         .      .      .      .      .      .      .      .      .      .
171         7      1      1      .      2      1      1      1      1      1
172         .      .      .      .      .      .      .      .      .      .
173         .      .      .      .      .      .      .      .      .      .
174         13     3      2      .      5      2      2      .      0      0
175         .      .      .      .      .      .      .      .      .      .
176         .      .      .      .      .      .      .      .      .      .
177         12     2      1      .      7      0      0      0      0      0
178         .      .      .      .      .      .      .      .      .      .
179         6      1      1      .      3      0      0      .      0      0
180         6      1      1      .      3      0      0      .      0      0
181         6      1      1      .      3      0      0      .      0      0
182         6      0      0      .      3      0      0      .      0      0
183         .      .      .      .      .      .      .      .      .      .
184         .      .      .      .      .      .      .      .      .      .
185         3      1      1      .      0      0      0      .      0      0
186         .      .      .      .      .      .      .      .      .      .

```



```

187 . . . _exit_main: . . .
188 . . . fprintf(stderr, " at line %u\n", (
    unsigned) lineno);
189 . . . free(line);
190 . . . destroy_matrix(a);
191 . . . destroy_matrix(b);
192 . . . destroy_matrix(c);
193 . . . exit(1);
194 6 1 1 2 0 0 0 0
    0 }
195 . . .
196 . . . matrix_t*
197 . . . matrix_multiply(matrix_t* m1, matrix_t*
    m2, int bs)
198 11 2 2 0 0 0 6 0
    0 {
199 . . . size_t n, en, i, j, k, kk, jj;
200 . . . double sum;
201 . . . matrix_t* mr;
202 . . .
203 3 1 1 2 0 0 1 0
    0 n = m1->rows;
204 . . .
205 11 1 1 5 0 0 1 0
    0 if(!(mr = create_matrix(n,n))) return
    NULL;
206 . . .
207 9 2 2 3 0 0 1 0
    0 en = bs*(n/bs);
208 . . .
209 24 2 2 8 0 0 3 0
    0 for(i=0; i<n; i++)
210 48 1 1 16 0 0 6 0
    0 for(j=0; j<n; j++)
211 44 1 1 20 0 0 8 0
    0 mr->array[i*n+j] = 0.0;
212 . . .

```

213
214	.	.	.	#endif	1
215	.	.	.	if (1) {
216	2	0	0	size_t j;
217	9	1	1	0	0	0	0	1	0	0
218	83,886,088	2	2	size_t dim = 1024*1024*10;	0	0	10,485,761	.	.	0
219	62,914,560	0	0	0	20,971,520	0	0	10,485,760	1,310,720	0
220	1,310,720	6	1	int *v = malloc(dim*sizeof(int));	3	0	0	1	0	0
221	.	.	.	for (j = 0; j < dim; ++j)
222	.	.	.	v[j] = -1;
223	.	.	.	0	free(v);
224	17	2	2	}
225	17	2	2	#endif
226	24	1	1
227	62	2	2	for(kk=0; kk<en; kk+=bs)	6	0	0	2	0	0
228	44	1	1	for(jj=0; jj<en; jj+=bs)	8	0	0	3	0	0
229	124	3	3	for(i=0; i<n; i++)	24	0	0	6	0	0
230	192	2	2	0	for(j=jj; j<jj+bs; j++) {	24	2	2	4	0
231	44	1	1	0	sum = mr->array[i*n+j];	48	0	0	12	0
232	.	.	.	0	for(k=kk; k<kk+bs; k++)	104	3	3	8	0
233	1	0	0	sum += m1->array[i*n+k] * m2	->array[k*n+j];	24	0	0	4	0
234	6	1	1	0	mr->array[i*n+j] = sum;
235	.	.	.	}
236	.	.	.	0	return mr;	2	0	0	0	0
237	.	.	.	0 }
238	18	3	3	char*
				read_line(FILE *fp)	0	0	0	8	0	0
				0 {

```

239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .      .
242      .      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      .      .      .      .      .      .      .      .
246      .      .      .      .      .      .      .      .
247      .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
251      .      .      .      .      .      .      .      .
252      .      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .
262      .      .      .      .      .      .      .      .
263      .      .      .      .      .      .      .      .
264      .      .      .      .      .      .      .      .
265      .      .      .      .      .      .      .      .

#define DEF_LINE_SZ 1024

int c;
size_t len = 0, tam = DEF_LINE_SZ;
char* str;

str = malloc(tam);
if (!str) {
    perror("");
    return NULL;
}

while (EOF != (c=fgetc(fp)) && c != '\n') {
    str[len++]=c;
    if (len==tam-1) {
        str = realloc(str, tam *= 2);
        if (!str) {
            perror("");
            return NULL;
        }
    }
}

if (c != EOF)
    str[len++]='\n';

```

```

266         2    0    0        str[len++]='\0';
267         12   1    1        4    0    0        0    0
268         0    }
269         .    .    .
270         .    .    .
271         .    .    .    matrix_t*
272         30   1    1        0    0    0        15    0
273         0    {
274         .    .    .    matrix_t * m;
275         30   2    2        9    1    0        3    0
276         0    if (!(m = malloc(sizeof(matrix_t)))) {
277         .    .    .    perror("");
278         .    .    .    return NULL;
279         .    .    .    }
280         9    1    1        6    0    0        3    0
281         0    m->rows = rows;
282         9    1    1        6    0    0        3    0
283         0    m->cols = cols;
284         51   2    2        21   0    0        3    0
285         0    if (!(m->array = malloc(sizeof(double)
286         *rows*cols))) {
287         .    .    .    free(m);
288         .    .    .    perror("");
289         .    .    .    return NULL;
290         .    .    .    }
291         3    1    1        3    0    0        0    0
292         0    return m;
293         18   1    1        6    0    0        0    0
294         0    }
295         .    .    .
296         .    .    .    void

```

```

292      .      .      .      .      .      .      .      .
293      27      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      {
294      9      0      0      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      if (!m) return;
295      24      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      free(m->array);
296      24      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      free(m);
297      18      0      0      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      }
298      .      .      .      .      .      .      .      .
299      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      int
300      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      print_matrix(FILE* fp, matrix_t* m)
301      10      2      2      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      {
302      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      size_t i, j;
303      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      size_t n;
304      3      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      n = m->rows;
305      13      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      if (fprintf(fp, "%lu", (unsigned long)
      m->rows) < 0) {
306      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      perror("");
307      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      return -1;
308      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      }
309      24      2      2      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      for(i=0; i<n; i++)
310      48      2      2      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      for (j=0; j<n; j++)
311      88      3      3      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      if (fprintf(fp, " %g", m->array[i*
      n+j]) < 0) {
312      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      perror("");
313      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      return -1;
314      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      }
315      10      1      1      .      .      .      .      .
      .      .      .      .      .      .      .      .
      0      if (fprintf(fp, "\n") < 0) {
316      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      perror("");
317      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      return -1;

```

```

318      .      .      .      .      .      .      .      .
319      1      1      1      0      0      0      0      0
320      6      1      1      2      0      0      0      0
321      0      }
322
323  Ir      I1mr ILmr Dr      D1mr DLmr Dw      D1mw
324      DLmw
325  146,802,968  121  119 52,429,703  20  14 20,971,776 1,310,724
      1,310,722  events annotated

```

```

1  3 10 29 43 26 70 95 29 60 76
2  nd branch-prediction profiler
3  ==632== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
      Nethercote et al.
4  ==632== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
      copyright info
5  ==632== Command: /tmp/02-mmult
6  ==632== Parent PID: 597
7  ==632==
8  --632-- Warning: Cannot auto-detect cache config, using
      defaults.
9  --632--      Run with -v to see.
10 ==632==
11 ==632== I   refs:      147,126,985
12 ==632== I1  misses:      2,407
13 ==632== LLi misses:      2,389
14 ==632== I1  miss rate:      0.00%
15 ==632== LLi miss rate:      0.00%
16 ==632==
17 ==632== D   refs:      73,528,542 (52,513,745 rd +
      21,014,797 wr)
18 ==632== D1  misses:      1,314,930 ( 3,625 rd +
      1,311,305 wr)
19 ==632== LLd misses:      1,314,153 ( 2,901 rd +
      1,311,252 wr)
20 ==632== D1  miss rate:      1.8% ( 0.0% +
      6.2% )
21 ==632== LLd miss rate:      1.8% ( 0.0% +
      6.2% )
22 ==632==
23 ==632== LL refs:      1,317,337 ( 6,032 rd +
      1,311,305 wr)
24 ==632== LL misses:      1,316,542 ( 5,290 rd +
      1,311,252 wr)
25 ==632== LL miss rate:      0.6% ( 0.0% +
      6.2% )
26

```

```

27 I1 cache:      32768 B, 32 B, 4-way associative
28 D1 cache:      32768 B, 32 B, 2-way associative
29 LL cache:      524288 B, 32 B, 8-way associative
30 Command:       /tmp/02-mmult
31 Data file:      cachegrind.out.632
32 Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:   Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:     0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
39
40 -----
41 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
42          DLmw
43 147,126,985 2,407 2,389 52,513,745 3,625 2,901 21,014,797
44 1,311,305 1,311,252 PROGRAM TOTALS
45 -----
46 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
47          DLmw          file:function
48 146,802,337 29 29 52,429,584 11 11 20,971,667
49 1,310,721 1,310,721 /root/CARPETA/tp2-2020-2q-src/main.c:
50 matrix_multiply
51 -----
52 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
53 -----
54 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
55          DLmw
56 -- line 18 -----
57          .      .      .      .      .      .      .
58          .      .      .      .      .      .      .
59          .      .      .      .      .      .      .
60          .      .      .      .      .      .      .
61          .      .      .      .      .      .      .

```

```

62      .      .      .      .      .      .      .      .
63      .      .      .      .      .      .      .      .
64      .      .      .      .      .      .      .      .
10      2      2      0      0      0      5      0
0      {
65      .      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .
69      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
70      1      0      0      0      0      0      1      0
      0      char *line = NULL;
71      .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .
76      2      1      1      0      0      0      1      0
      0      size_t lineno = 1;
77      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .
82      25      4      4      9      0      0      1      0
      0      for(; !feof(stdin); lineno++) {
83      10      1      1      4      0      0      6      0
      0      a=b=c=NULL;
84      .      .      .      .      .      .      .      .
85      18      3      3      8      1      0      2      0
      0      line = read_line(stdin);
86      .      .      .      .      .      .      .      .
87      6      0      0      2      0      0      0      0
      0      if (!line) goto _exit_main;

```


88	9	0	0	if (line[0] == 0) break;	0
89
90
91	2	1	1	/* parse dimension */	1
92	10	1	1	nptr = line;	0
93	8	1	1	l = strtol(nptr, &endptr, 10);	0
94	.	.	.	if (errno) {	.
95	.	.	.	perror("");	.
96	.	.	.	goto _exit_main;	.
97	4	2	2	}	0
98	.	.	.	if (nptr == endptr) {	.
99	.	.	.	fprintf(stderr, "missing	.
100	.	.	.	dimension");	.
101	3	2	2	goto _exit_main;	0
102	.	.	.	}	.
103	.	.	.	if (l < 1) {	.
104	.	.	.	fprintf(stderr, "invalid	.
105	2	1	1	dimension");	0
106	.	.	.	goto _exit_main;	.
107	.	.	.	}	.
108	.	.	.	if 0	.
109	.	.	.	/* parse block size */	.
110	.	.	.	npntr = endptr;	.
111	.	.	.	l = strtol(nptr, &endptr, 10);	.
112	.	.	.	if (errno) {	.
113	.	.	.	perror("");	.
	.	.	.	goto _exit_main;	.
	.	.	.	}	.

```

114 -- line 75 -----
115 -- line 83 -----
116 . . . } . . .
117 . . . bs = (size_t) 1; . . .
118 . . . . . . . . . .
119 . . . if (n % bs) { . . .
120 . . . . . fprintf(stderr, "block size doesn
't match");
121 . . . . . goto _exit_main;
122 . . . . . }
123 . . . #else . . .
124 2 0 0 1 0 0 1 0
0 bs = n;
125 . . . . .
126 . . . #endif . . .
127 . . . . .
128 11 1 1 5 1 0 1 0
0 if (!(a = create_matrix(n, n)))
129 . . . . . goto _exit_main;
130 . . . . .
131 100 3 3 39 0 0 10 0
0 for (i=0; i < n*n; i++) {
132 18 0 0 9 0 0 9 0
0 nptr = endptr;
133 81 1 1 27 0 0 9 0
0 e = strtod(nptr, &endptr);
134 72 1 1 27 0 0 0 0
0 if (errno) {
135 . . . . . perror("");
136 . . . . . goto _exit_main;
137 . . . . . }
138 36 2 2 18 0 0 0 0
0 if (nptr == endptr) {
139 . . . . . fprintf(stderr, "missing A
matrix element");
140 . . . . . goto _exit_main;

```

141	}	.	.	.
142	63	1	1	36	0	0	9	1
143	.	.	1	a->array[i] = e;
144	.	.	.	}
145
146	11	1	1	/* load matrix b */	5	0	0	1
147	.	.	0	if (!(b = create_matrix(n, n)))
148	.	.	.	goto _exit_main;
149	100	2	2	39	0	0	10	0
150	18	1	1	for (i=0; i < n*n; i++) {	9	0	0	9
151	81	1	1	nptr = endptr;	27	0	0	9
152	72	1	1	e = strtod(nptr, &endptr);	27	0	0	0
153	.	.	0	if (errno) {
154	.	.	.	perror("");
155	.	.	.	goto _exit_main;
156	36	1	1	}	18	0	0	0
157	.	.	0	if (nptr == endptr) {
158	.	.	.	fprintf(stderr, "missing B
159	.	.	.	matrix element");
160	63	1	1	goto _exit_main;
161	.	.	1	}
162	.	.	.	b->array[i] = e;
163	8	1	1	}
164	.	.	0	clock_gettime(CLOCK_REALTIME, &t0);
165
166	13	2	2	/* multiply matrixes */	6	1	1	1
	.	.	0	if (!(c = matrix_multiply(a, b, bs))
	.	.	.)

```

167      .      .      .      .      .      .      .      .
168      .      .      .      .      .      .      .      .
169      8      1      1      .      2      0      0      0      0
170      .      .      .      .      .      .      .      .
171      7      1      1      .      2      1      1      1      1
172      .      .      .      .      .      .      .      .
173      .      .      .      .      .      .      .      .
174      12     1      1      .      5      2      2      1      0
175      .      .      .      .      .      .      .      .
176      .      .      .      .      .      .      .      .
177      12     1      1      .      7      0      0      0      0
178      .      .      .      .      .      .      .      .
179      6      1      1      .      3      0      0      0      0
180      .      .      .      .      .      .      .      .
181      6      1      1      .      3      0      0      0      0
182      6      0      0      .      3      0      0      0      0
183      .      .      .      .      .      .      .      .
184      .      .      .      .      .      .      .      .
185      3      1      1      .      0      0      0      0      0
186      .      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .      .
192      .      .      .      .      .      .      .      .

goto _exit_main;

clock_gettime(CLOCK_REALTIME, &t1);

dt = (float) (t1.tv_sec - t0.tv_sec);
dt = dt + ((float)(t1.tv_nsec - t0.tv_nsec)) / 1.0e9;

if(print_matrix(stdout, c) == -1)
    goto _exit_main;

fprintf(stderr, "time: %g\n", dt);

free(line);
destroy_matrix(a);
destroy_matrix(b);
destroy_matrix(c);
}

return 0;

_exit_main:

fprintf(stderr, " at line %u\n", (
unsigned) lineno);

free(line);
destroy_matrix(a);
destroy_matrix(b);

```

193	destroy_matrix(c);	.	.	.
194	6	1	1	.	exit(1);	2	0	0
195	.	.	.	0	}	.	.	.
196
197	matrix_t*	.	.	.
198	11	2	2	.	matrix_multiply(matrix_t* m1, matrix_t*	.	.	.
199	m2, int bs)	.	.	.
200
201
202
203	3	1	1
204
205	11	1	1
206
207	9	2	2
208
209	32	2	2
210	96	1	1
211	99	1	1
212
213
214
215
216	2	0	0
217	9	1	1
218	83,886,088	2	2

```

0      for (j = 0; j < dim; ++j)
219 62,914,560 0 0 20,971,520 0 0 10,485,760 1,310,720
      v[j] = -1;
220      6 1 1 3 2 2 0 0
      free(v);
221      . . . . . . . .
      }
222      . . . . . . . .
      #endif
223      . . . . . . . .
      .
224      17 2 2 6 0 0 2 0
      for(kk=0; kk<en; kk+=bs)
225      17 2 2 6 0 0 2 0
      for(jj=0; jj<en; jj+=bs)
226      32 1 1 11 0 0 4 0
      for(i=0; i<n; i++)
227      123 2 2 48 0 0 12 0
      for(j=jj; j<jj+bs; j++) {
228      99 1 1 54 3 3 9 0
      sum = mr->array[i*n+j];
229      369 3 3 144 0 0 36 0
      for(k=kk; k<kk+bs; k++)
230      648 2 2 351 6 6 27 0
      sum += m1->array[i*n+k] * m2
      ->array[k*n+j];
231      99 1 1 54 0 0 9 0
      mr->array[i*n+j] = sum;
232      . . . . . . . .
      }
233      1 0 0 1 0 0 0 0
      return mr;
234      6 1 1 2 0 0 0 0
      0 }
235      . . . . . . . .
      .
236      . . . . . . . .
      char*
237      . . . . . . . .
      read_line(FILE *fp)
238      18 3 3 0 0 0 8 0
      {
239      . . . . . . . .
      #define DEF_LINE_SZ 1024
240      . . . . . . . .
      .
241      . . . . . . . .
      int c;
242      6 0 0 0 0 0 4 0
      size_t len = 0, tam = DEF_LINE_SZ;
243      . . . . . . . .
      char* str;
244      . . . . . . . .
      .

```

```

245      14      1      1          6      1      0          2          1
      0      str = malloc(tam);
246      6      0      0          2      0      0          0          0
      0      if (!str) {
247      .      .      .          .      .      .          .      .
      .          perror("");
248      .      .      .          .      .      .          .      .
      .          return NULL;
249      .      .      .          .      .      .          .      .
      .      .      .      }
250      .      .      .          .      .      .          .      .
      .
251      581     5      5          194     1      0          39          0
      0      while (EOF != (c=fgetc(fp)) && c != '\n
      ') {
252      296     2      2          111     0      0          74          1
      1      str[len++]=c;
253      185     0      0          74      0      0          0          0
      0      if (len==tam-1) {
254      .      .      .          .      .      .          .      .
      .          str = realloc(str, tam *= 2);
255      .      .      .          .      .      .          .      .
      .          if (!str) {
256      .      .      .          .      .      .          .      .
      .          perror("");
257      .      .      .          .      .      .          .      .
      .          return NULL;
258      .      .      .          .      .      .          .      .
      .      .      .      }
259      .      .      .          .      .      .          .      .
      .      .      .      }
260      .      .      .          .      .      .          .      .
      .      .      .      }
261      .      .      .          .      .      .          .      .
      .
262      8      1      1          2      0      0          0          0
      0      if (c != EOF)
263      7      0      0          2      0      0          2          0
      0      str[len++]='\n';
264      .      .      .          .      .      .          .      .
      .
265      12     1      1          4      0      0          4          0
      0      str[len++]='\0';
266      2      0      0          2      0      0          0          0
      0      return str;
267      12     1      1          4      0      0          0          0
      0  }
268      .      .      .          .      .      .          .      .
      .
269      .      .      .          .      .      .          .      .
      .
270      .      .      .          .      .      .          .      .
      .      .      .      matrix_t*
271      .      .      .          .      .      .          .      .

```

```

272         .   create_matrix(size_t rows, size_t cols)
30      1      1      0      0      0      15      0
273         .   .   .   .   .   .   .
274         .   .   .   .   .   .   .
275         .   .   .   .   .   .   .
30      2      2      9      1      0      3      0
276         .   .   .   .   .   .   .
0      if (!(m = malloc(sizeof(matrix_t)))) {
277         .   .   .   .   .   .   .
278         .   .   .   .   .   .   .
279         .   .   .   .   .   .   .
280         .   .   .   .   .   .   .
9      1      1      6      0      0      3      0
281         .   .   .   .   .   .   .
9      1      1      6      0      0      3      0
282         .   .   .   .   .   .   .
51     2      2      21     0      0      3      0
0      if (!(m->array = malloc(sizeof(double)
*rows*cols))) {
283         .   .   .   .   .   .   .
284         .   .   .   .   .   .   .
285         .   .   .   .   .   .   .
286         .   .   .   .   .   .   .
287         .   .   .   .   .   .   .
288         .   .   .   .   .   .   .
3      1      1      3      0      0      0      0
0      return m;
289     18     1      1      6      0      0      0      0
0      }
290         .   .   .   .   .   .   .
291         .   .   .   .   .   .   .
292         .   .   .   .   .   .   .
293         .   .   .   .   .   .   .
27     1      1      0      0      0      12     0
0      {
294     9      0      0      3      0      0      0      0
0      if (!m) return;
295     24     1      1      12     0      0      0      0
0      free(m->array);
296     24     1      1      9      0      0      0      0
0      free(m);
297     18     0      0      6      0      0      0      0
0      }

```



```

298      .      .      .      .      .      .      .      .
299      .      .      .      .      .      .      .      .
300      .      .      .      .      .      .      .      .
301      .      .      .      .      .      .      .      .
302      .      .      .      .      .      .      .      .
303      .      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .      .
305      .      .      .      .      .      .      .      .
306      .      .      .      .      .      .      .      .
307      .      .      .      .      .      .      .      .
308      .      .      .      .      .      .      .      .
309      .      .      .      .      .      .      .      .
310      .      .      .      .      .      .      .      .
311      .      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .      .
314      .      .      .      .      .      .      .      .
315      .      .      .      .      .      .      .      .
316      .      .      .      .      .      .      .      .
317      .      .      .      .      .      .      .      .
318      .      .      .      .      .      .      .      .
319      .      .      .      .      .      .      .      .
320      .      .      .      .      .      .      .      .
321      .      .      .      .      .      .      .      .
322      .      .      .      .      .      .      .      .
323      .      .      .      .      .      .      .      .
324      .      .      .      .      .      .      .      .

    int
    print_matrix(FILE* fp, matrix_t* m)
    {
        size_t i, j;
        size_t n;
        n = m->rows;
        if (fprintf(fp, "%lu", (unsigned long)
            m->rows) < 0) {
            perror("");
            return -1;
        }
        for(i=0; i<n; i++)
            for (j=0; j<n; j++)
                if (fprintf(fp, " %g", m->array[i*n
                    +j]) < 0) {
                    perror("");
                    return -1;
                }
            if (fprintf(fp, "\n") < 0) {
                perror("");
                return -1;
            }
        return 0;
    }

```

	Ir		IImr ILmr Dr		DImr DLmr Dw		DImw
			DLmw				

```
325 146,805,085 120 119 52,430,618 25 18 20,971,960 1,310,727
    1,310,725 events annotated
```

```
1 4 97 46 101 53 76 89 135 88 113 87 126 89 90 88 142 102
2 er
3 ==635== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
    Nethercote et al.
4 ==635== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
5 ==635== Command: /tmp/02-mmult
6 ==635== Parent PID: 597
7 ==635==
8 --635-- Warning: Cannot auto-detect cache config, using
    defaults.
9 --635-- Run with -v to see.
10 ==635==
11 ==635== I refs: 147,153,677
12 ==635== I1 misses: 2,407
13 ==635== LLi misses: 2,387
14 ==635== I1 miss rate: 0.00%
15 ==635== LLi miss rate: 0.00%
16 ==635==
17 ==635== D refs: 73,538,426 (52,520,455 rd +
    21,017,971 wr)
18 ==635== D1 misses: 1,314,942 ( 3,630 rd +
    1,311,312 wr)
19 ==635== LLd misses: 1,314,166 ( 2,907 rd +
    1,311,259 wr)
20 ==635== D1 miss rate: 1.8% ( 0.0% +
    6.2% )
21 ==635== LLd miss rate: 1.8% ( 0.0% +
    6.2% )
22 ==635==
23 ==635== LL refs: 1,317,349 ( 6,037 rd +
    1,311,312 wr)
24 ==635== LL misses: 1,316,553 ( 5,294 rd +
    1,311,259 wr)
25 ==635== LL miss rate: 0.6% ( 0.0% +
    6.2% )
26 -----
27 I1 cache: 32768 B, 32 B, 4-way associative
28 D1 cache: 32768 B, 32 B, 2-way associative
29 LL cache: 524288 B, 32 B, 8-way associative
30 Command: /tmp/02-mmult
31 Data file: cachegrind.out.635
32 Events recorded: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds: 0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
```

39	
40	
41	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw
	DLmw
42	
43	147,153,677 2,407 2,387 52,520,455 3,630 2,907 21,017,971
44	1,311,312 1,311,259 PROGRAM TOTALS
45	
46	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw
	DLmw file:function
47	
48	146,804,064 29 29 52,430,421 16 16 20,971,794
49	1,310,723 1,310,723 /root/CARPETA/tp2-2020-2q-src/main.c:
50	matrix_multiply
51	
52	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
53	
54	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw
55	DLmw
56	
57	-- line 18 -----
58	. . . matrix_t* create_matrix(size_t rows,
59	size_t cols);
60	. . .
61	. void destroy_matrix(matrix_t* m);
62	. . .
63	. int print_matrix(FILE* fp, matrix_t* m)
64	;
65	. . .
66	. int
67	. main(int argc, char** argv)
68	10 2 2 0 0 0 5 0
69	0 {
70	. . .
71	. /* matrixes */
72	. matrix_t *a, *b, *c;
73	. . .

```

68      .      .      .      /* n (dimension) and block size */
69      .      .      .      size_t n, bs;
70      .      .      .      /* line buffer (init to null to
71      .      .      .      simplify freeing on error) */
72      .      .      .      char *line = NULL;
73      .      .      .      /* line parsing auxiliar pointers */
74      .      .      .      char *nptr, *endptr;
75      .      .      .      /* auxiliar variables */
76      .      .      .      long l;
77      .      .      .      double e;
78      .      .      .      size_t lineno = 1;
79      .      .      .      struct timespec t0;
80      .      .      .      struct timespec t1;
81      .      .      .      double dt;
82      .      .      .      size_t i;
83      .      .      .      for(;; !feof(stdin); lineno++) {
84      .      .      .      a=b=c=NULL;
85      .      .      .      line = read_line(stdin);
86      .      .      .      if (!line) goto _exit_main;
87      .      .      .      if (line[0] == 0) break;
88      .      .      .      /* parse dimension */
89      .      .      .      nptr = line;
90      .      .      .      l = strtol(nptr, &endptr, 10);
91      .      .      .      if (errno) {

```

```

94         perror("");
95         goto _exit_main;
96     }
97     4 2 2 0 2 0 0 0 0
98     if (nptr == endptr) {
99         fprintf(stderr, "missing
100         dimension");
101         goto _exit_main;
102     }
103     3 2 2 0 1 0 0 0 0
104     if (l < 1) {
105         fprintf(stderr, "invalid
106         dimension");
107         goto _exit_main;
108     }
109     2 1 1 0 1 0 0 1 0
110     n = (size_t) l;
111     #if 0
112     /* parse block size */
113     nptr = endptr;
114     l = strtol(nptr, &endptr, 10);
115     if (errno) {
116         perror("");
117         goto _exit_main;
118     }
119     -- line 75 -----
120     -- line 83 -----
121     }
122     bs = (size_t) l;
123     if (n % bs) {
124         fprintf(stderr, "block size doesn

```

```

121         't match");
122         .      .      .      .      .      .      .      .
123         .      .      .      .      .      .      .      .
124         .      .      .      .      .      .      .      .
125         .      .      .      .      .      .      .      .
126         .      .      .      .      .      .      .      .
127         .      .      .      .      .      .      .      .
128         .      .      .      .      .      .      .      .
129         .      .      .      .      .      .      .      .
130         .      .      .      .      .      .      .      .
131         .      .      .      .      .      .      .      .
132         .      .      .      .      .      .      .      .
133         .      .      .      .      .      .      .      .
134         .      .      .      .      .      .      .      .
135         .      .      .      .      .      .      .      .
136         .      .      .      .      .      .      .      .
137         .      .      .      .      .      .      .      .
138         .      .      .      .      .      .      .      .
139         .      .      .      .      .      .      .      .
140         .      .      .      .      .      .      .      .
141         .      .      .      .      .      .      .      .
142         .      .      .      .      .      .      .      .
143         .      .      .      .      .      .      .      .
144         .      .      .      .      .      .      .      .
145         .      .      .      .      .      .      .      .
146         .      .      .      .      .      .      .      .

```

```

        goto _exit_main;
    }

    #else
        2    0    0    1    0    0    1    0
        0    bs = n;
    #endif

    /* load matrix a */
    11    1    1    5    1    0    1    0
        0    if (!(a = create_matrix(n, n)))
        goto _exit_main;

    170    3    3    67    0    0    17    0
        0    for (i=0; i < n*n; i++) {
    32    0    0    16    0    0    16    0
        0    nptr = endptr;
    144    1    1    48    0    0    16    0
        0    e = strtod(nptr, &endptr);
    128    1    1    48    0    0    0    0
        0    if (errno) {
        perror("");
        goto _exit_main;
    }
    64    2    2    32    0    0    0    0
        0    if (nptr == endptr) {
        fprintf(stderr, "missing A
matrix element");
        goto _exit_main;
    }
    112    1    1    64    0    0    16    3
        3    a->array[i] = e;
    }

    /* load matrix b */
    11    1    1    5    0    0    1    0
        0    if (!(b = create_matrix(n, n)))

```

```

147      .      .      .      .      .      .      .      .
148      .      .      .      .      .      .      .      .
149      170    2    2      67    0    0      17      0
150      .      .      .      .      .      .      .      .
151      32    1    1      16    0    0      16      0
152      .      .      .      .      .      .      .      .
153      144    1    1      48    0    0      16      0
154      .      .      .      .      .      .      .      .
155      128    1    1      48    0    0      0      0
156      .      .      .      .      .      .      .      .
157      .      .      .      .      .      .      .      .
158      .      .      .      .      .      .      .      .
159      .      .      .      .      .      .      .      .
160      64    1    1      32    0    0      0      0
161      .      .      .      .      .      .      .      .
162      .      .      .      .      .      .      .      .
163      .      .      .      .      .      .      .      .
164      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
166      112    1    1      64    0    0      16      4
167      .      .      .      .      .      .      .      .
168      .      .      .      .      .      .      .      .
169      .      .      .      .      .      .      .      .
170      .      .      .      .      .      .      .      .
171      .      .      .      .      .      .      .      .
172      .      .      .      .      .      .      .      .

```

```

                                0      dt = dt + ((float)(t1.tv_nsec - t0.
                                tv_nsec)) / 1.0e9;
173      .      .      .      .      .      .      .      .
174      13      3      2      .      5      2      2      0      0
                                0      if(print_matrix(stdout, c) == -1)
175      .      .      .      .      .      .      .      .
                                goto _exit_main;
176      .      .      .      .      .      .      .      .
                                .
177      12      2      1      .      7      0      0      0      0
                                0      fprintf(stderr, "time: %g\n", dt);
178      .      .      .      .      .      .      .      .
                                .
179      6      1      1      .      3      0      0      0      0
                                0      free(line);
180      6      1      1      .      3      0      0      0      0
                                0      destroy_matrix(a);
181      6      1      1      .      3      0      0      0      0
                                0      destroy_matrix(b);
182      6      0      0      .      3      0      0      0      0
                                0      destroy_matrix(c);
183      .      .      .      .      .      .      .      .
                                }
184      .      .      .      .      .      .      .      .
                                .
185      3      1      1      .      0      0      0      0      0
                                0      return 0;
186      .      .      .      .      .      .      .      .
                                .
187      .      .      .      .      .      .      .      .
                                .      _exit_main:
188      .      .      .      .      .      .      .      .
                                .      fprintf(stderr, " at line %u\n", (
                                unsigned) lineno);
189      .      .      .      .      .      .      .      .
                                .      free(line);
190      .      .      .      .      .      .      .      .
                                .      destroy_matrix(a);
191      .      .      .      .      .      .      .      .
                                .      destroy_matrix(b);
192      .      .      .      .      .      .      .      .
                                .      destroy_matrix(c);
193      .      .      .      .      .      .      .      .
                                .      exit(1);
194      6      1      1      .      2      0      0      0      0
                                0      }
195      .      .      .      .      .      .      .      .
                                .
196      .      .      .      .      .      .      .      .
                                .      matrix_t*
197      .      .      .      .      .      .      .      .
                                .      matrix_multiply(matrix_t* m1, matrix_t*
                                m2, int bs)

```


198	11	2	2	0	0	0	6	0
199
200
201
202
203	3	1	1	2	0	0	1	0
204
205	11	1	1	5	0	0	1	0
206	NULL;		0	if(!(mr = create_matrix(n,n))) return				
207	9	2	2	3	0	0	1	0
208
209	40	2	2	14	0	0	5	0
210	160	1	1	56	0	0	20	0
211	176	1	1	80	0	0	32	3
212
213
214	.	.	.	if (1) {				
215	.	.	.	size_t j;				
216	2	0	0	0	0	0	1	0
217	9	1	1	3	0	0	1	0
218	83,886,088	2	2	31,457,282	0	0	10,485,761	0
219	62,914,560	0	0	20,971,520	0	0	10,485,760	1,310,720
220	1,310,720	6	1	3	2	2	0	0
221
222
223
224	17	2	2	6	0	0	2	0

```

225         17      2      2      0      for(kk=0; kk<en; kk+=bs)
226         40      1      1      0      for(jj=0; jj<en; jj+=bs)
227         204     2      2      0      for(i=0; i<n; i++)
228         176     1      1      0      for(j=jj; j<jj+bs; j++) {
229         816     3      3      0      sum = mr->array[i*n+j];
230         1,536   2      2      0      for(k=kk; k<kk+bs; k++)
231         176     1      1      0      sum += m1->array[i*n+k] * m2->
232         .      .      .      0      array[k*n+j];
233         1      0      0      0      mr->array[i*n+j] = sum;
234         6      1      1      0      }
235         .      .      .      0      return mr;
236         .      .      .      0      }
237         .      .      .      .      char*
238         18      3      3      0      read_line(FILE *fp)
239         .      .      .      0      {
240         .      .      .      .      #define DEF_LINE_SZ 1024
241         .      .      .      .      int c;
242         6      0      0      0      size_t len = 0, tam = DEF_LINE_SZ;
243         .      .      .      .      char* str;
244         .      .      .      .      str = malloc(tam);
245         14      1      1      0      if (!str) {
246         6      0      0      0      perror("");
247         .      .      .      .      return NULL;
248         .      .      .      .      }
249         .      .      .      .      }
250         .      .      .      .      .

```

```

251      1,001      5      5      334      1      0      67      0
           0      while (EOF != (c=fgetc(fp)) && c != '\n')
           {
252      520      2      2      195      0      0      130      2
           2      str[len++]=c;
253      325      0      0      130      0      0      0      0
           0      if (len==tam-1) {
254      .      .      .      .      .      .      .      .
           .      str = realloc(str, tam *= 2);
255      .      .      .      .      .      .      .      .
           .      if (!str) {
256      .      .      .      .      .      .      .      .
           .      perror("");
257      .      .      .      .      .      .      .      .
           .      return NULL;
258      .      .      .      .      .      .      .      .
           .      }
259      .      .      .      .      .      .      .      .
           .      }
260      .      .      .      .      .      .      .      .
           .      }
261      .      .      .      .      .      .      .      .
           .
262      8      1      1      2      0      0      0      0
           0      if (c != EOF)
263      7      0      0      2      0      0      2      0
           0      str[len++]='\n';
264      .      .      .      .      .      .      .      .
           .
265      12      1      1      4      0      0      4      0
           0      str[len++]='\0';
266      2      0      0      2      0      0      0      0
           0      return str;
267      12      1      1      4      0      0      0      0
           0 }
268      .      .      .      .      .      .      .      .
           .
269      .      .      .      .      .      .      .      .
           .
270      .      .      .      .      .      .      .      .
           .      matrix_t*
271      .      .      .      .      .      .      .      .
           .      create_matrix(size_t rows, size_t cols)
272      30      1      1      0      0      0      15      0
           0 {
273      .      .      .      .      .      .      .      .
           .      matrix_t * m;
274      .      .      .      .      .      .      .      .
           .
275      30      2      2      9      1      0      3      0
           0      if (!(m = malloc(sizeof(matrix_t)))) {
276      .      .      .      .      .      .      .      .
           .      perror("");
277      .      .      .      .      .      .      .      .

```

```

278         .         .         .         .         .         .         .         .
279         .         .         .         .         .         .         .         .
280         9         1         1         .         6         0         0         .         3         0
281         .         .         .         .         .         .         .         .
282         9         1         1         .         6         0         0         .         3         0
283         51        2         2         .         21        0         0         .         3         0
284         .         .         .         .         .         .         .         .
285         .         .         .         .         .         .         .         .
286         .         .         .         .         .         .         .         .
287         .         .         .         .         .         .         .         .
288         3         1         1         .         3         0         0         .         0         0
289         18        1         1         .         6         0         0         .         0         0
290         .         .         .         .         .         .         .         .
291         .         .         .         .         .         .         .         .
292         .         .         .         .         .         .         .         .
293         27        1         1         .         0         0         0         .         12        0
294         9         0         0         .         3         0         0         .         0         0
295         24        1         1         .         12        0         0         .         0         0
296         24        1         1         .         9         0         0         .         0         0
297         18        0         0         .         6         0         0         .         0         0
298         .         .         .         .         .         .         .         .
299         .         .         .         .         .         .         .         .
300         .         .         .         .         .         .         .         .
301         10        2         2         .         0         0         0         .         5         0
302         .         .         .         .         .         .         .         .
303         .         .         .         .         .         .         .         .

```

```

304      3      1      1          2      0      0          1      0
          0      n = m->rows;
305      13      1      1          6      1      1          0      0
          0      if (fprintf(fp, "%lu", (unsigned long)
m->rows) < 0) {
306      .      .      .          .      .      .          .      .
          .      perror("");
307      .      .      .          .      .      .          .      .
          .      return -1;
308      .      .      .          .      .      .          .      .
          .      }
309      40      2      2          14      0      0          5      0
          0      for(i=0; i<n; i++)
310      160      2      2          56      0      0          20      0
          0      for (j=0; j<n; j++)
311      352      3      3          160      0      0          0      0
          0      if (fprintf(fp, " %g", m->array[i*n
+j]) < 0) {
312      .      .      .          .      .      .          .      .
          .      perror("");
313      .      .      .          .      .      .          .      .
          .      return -1;
314      .      .      .          .      .      .          .      .
          .      }
315      10      1      1          4      0      0          0      0
          0      if (fprintf(fp, "\n") < 0) {
316      .      .      .          .      .      .          .      .
          .      perror("");
317      .      .      .          .      .      .          .      .
          .      return -1;
318      .      .      .          .      .      .          .      .
          .      }
319      1      1      1          0      0      0          0      0
          0      return 0;
320      6      1      1          2      0      0          0      0
          0      }

```

```

323 Ir      I1mr ILmr Dr      D1mr DLmr Dw      Dimw
      DLmw
324 -----
325 146,808,382 121 119 52,432,069 29 23 20,972,236 1,310,735
      1,310,733 events annotated

```

```

1 5 125 118 38 87 87 115 168 73 98 118 86 89 43 46 93 156 132 47
   79 148 79 110 62 57 137
2 017, and GNU GPL'd, by Nicholas Nethercote et al.
3 ==638== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4 ==638== Command: /tmp/02-mmult
5 ==638== Parent PID: 597
6 ==638==

```

```

7  --638-- Warning: Cannot auto-detect cache config, using
    defaults.
8  --638--          Run with -v to see.
9  ==638==
10 ==638== I   refs:      147,186,902
11 ==638== I1  misses:      2,400
12 ==638== L1i misses:      2,382
13 ==638== I1  miss rate:      0.00%
14 ==638== L1i miss rate:      0.00%
15 ==638==
16 ==638== D   refs:      73,550,864 (52,528,959 rd  +
    21,021,905 wr)
17 ==638== D1  misses:      1,315,099 (    3,754 rd  +
    1,311,345 wr)
18 ==638== L1d misses:      1,314,183 (    2,915 rd  +
    1,311,268 wr)
19 ==638== D1  miss rate:      1.8% (    0.0%  +
    6.2%  )
20 ==638== L1d miss rate:      1.8% (    0.0%  +
    6.2%  )
21 ==638==
22 ==638== LL refs:      1,317,499 (    6,154 rd  +
    1,311,345 wr)
23 ==638== LL misses:      1,316,565 (    5,297 rd  +
    1,311,268 wr)
24 ==638== LL miss rate:      0.6% (    0.0%  +
    6.2%  )
25 -----
26 I1 cache:      32768 B, 32 B, 4-way associative
27 D1 cache:      32768 B, 32 B, 2-way associative
28 LL cache:      524288 B, 32 B, 8-way associative
29 Command:      /tmp/02-mmult
30 Data file:      cachegrind.out.638
31 Events recorded: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
32 Events shown:   Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Event sort order: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Thresholds:     0.1 100 100 100 100 100 100 100 100
35 Include dirs:
36 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
37 Auto-annotation: off
38
39 -----
40 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw
    DLmw
41 -----
42 147,186,902 2,400 2,382 52,528,959 3,754 2,915 21,021,905
    1,311,345 1,311,268 PROGRAM TOTALS
43
44 -----
45 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw

```

```

46      DLmw      file:function
47 146,806,731   29   29 52,431,722   23   23 20,971,983
    1,310,725 1,310,725 /root/CARPETA/tp2-2020-2q-src/main.c:
    matrix_multiply
48
49 -----
50 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
51 -----
52 Ir          I1mr ILmr Dr          Dimr DLmr Dw          Dimw
    DLmw
53
54 -- line 18 -----
55      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
56      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
57      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
58      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
59      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
60      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
61      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
62      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
63      10      2      2      0      0      0      5      0
    0 {
64      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
65      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
69      1      0      0      0      0      0      1      0
    0      char *line = NULL;
70      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
71      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .
    .      .      .      .      .      .      .      .

```

```

73      .      .      .      /* auxiliar variables */
74      .      .      .      long l;
75      .      .      .      double e;
76      2      1      1      0      0      0      1      0
77      .      .      .      size_t lineno = 1;
78      .      .      .      struct timespec t0;
79      .      .      .      struct timespec t1;
80      .      .      .      double dt;
81      .      .      .      size_t i;
82      25      4      4      9      0      0      1      0
83      10      1      1      4      0      0      6      0
84      .      .      .      a=b=c=NULL;
85      18      3      3      8      1      0      2      0
86      .      .      .      line = read_line(stdin);
87      6      0      0      2      0      0      0      0
88      9      0      0      4      0      0      0      0
89      .      .      .      if (!line) goto _exit_main;
90      .      .      .      if (line[0] == 0) break;
91      .      .      .      /* parse dimension */
92      2      1      1      1      0      0      1      1
93      10      1      1      3      1      0      1      0
94      8      1      1      3      0      0      0      0
95      .      .      .      l = strtol(npstr, &endptr, 10);
96      .      .      .      if (errno) {
97      4      2      2      2      0      0      0      0
98      .      .      .      perror("");
99      .      .      .      goto _exit_main;
100     .      .      .      }
101     .      .      .      if (npstr == endptr) {
102     .      .      .      fprintf(stderr, "missing
dimension");
103     .      .      .      goto _exit_main;

```



```

99      .      .      .      .      .      .      .      .
100      3      2      2      .      1      0      0      .      0
101      .      .      .      .      if (l < 1) {
102      .      .      .      .      fprintf(stderr, "invalid
dimension");
103      .      .      .      .      goto _exit_main;
104      .      .      .      .      }
105      2      1      1      .      1      0      0      .      0
106      .      .      .      .      n = (size_t) l;
107      .      .      .      .      #if 0
108      .      .      .      .      /* parse block size */
109      .      .      .      .      nptr = endptr;
110      .      .      .      .      l = strtol(nptr, &endptr, 10);
111      .      .      .      .      if (errno) {
112      .      .      .      .      perror("");
113      .      .      .      .      goto _exit_main;
114      .      .      .      .      }
115      -- line 75 -----
116      .      .      .      .      }
117      .      .      .      .      bs = (size_t) l;
118      .      .      .      .      if (n % bs) {
119      .      .      .      .      fprintf(stderr, "block size doesn
't match");
120      .      .      .      .      goto _exit_main;
121      .      .      .      .      }
122      .      .      .      .      #else
123      2      0      0      .      1      0      0      .      0
124      .      .      .      .      bs = n;
125      .      .      .      .      #endif

```

126	.	.	.	/* load matrix a */	.
127	11	1	1	5 1 0 1	0
128	.	.	0	if (!(a = create_matrix(n, n)))	.
129	.	.	.	goto _exit_main;	.
130	260	3	3	103 0 0 26	0
131	50	0	0	for (i=0; i < n*n; i++) {	0
132	225	1	1	25 0 0 25	0
133	200	1	1	nptr = endptr;	0
134	.	.	0	e = strtod(nptr, &endptr);	0
135	.	.	0	75 0 0 0	0
136	.	.	0	if (errno) {	0
137	.	.	.	perror("");	.
138	.	.	.	goto _exit_main;	.
139	.	.	.	}	.
140	100	2	2	50 0 0 0	0
141	.	.	0	if (nptr == endptr) {	.
142	.	.	.	fprintf(stderr, "missing A	.
143	.	.	.	matrix element");	.
144	.	.	.	goto _exit_main;	.
145	.	.	.	}	.
146	175	1	1	100 0 0 25	5
147	.	.	5	a->array[i] = e;	.
148	.	.	.	}	.
149
150	.	.	.	/* load matrix b */	.
151	11	1	1	5 0 0 1	0
152	.	.	0	if (!(b = create_matrix(n, n)))	.
153	.	.	.	goto _exit_main;	.
154
155	260	2	2	103 0 0 26	0
156	50	1	1	for (i=0; i < n*n; i++) {	0
157	225	1	1	25 0 0 25	0
158	200	1	1	nptr = endptr;	0
159	.	.	0	e = strtod(nptr, &endptr);	0
160	.	.	0	75 0 0 0	0
161	.	.	0	if (errno) {	.
162

```

153         perror("");
154         goto _exit_main;
155     }
156     if (nptr == endptr) {
157         fprintf(stderr, "missing B
matrix element");
158         goto _exit_main;
159     }
160     b->array[i] = e;
161 }
162     clock_gettime(CLOCK_REALTIME, &t0);
163
164     /* multiply matrixes */
165     if (!(c = matrix_multiply(a, b, bs))
    )
166         goto _exit_main;
167
168     clock_gettime(CLOCK_REALTIME, &t1);
169
170     dt = (float) (t1.tv_sec - t0.tv_sec
    );
171     dt = dt + ((float)(t1.tv_nsec - t0.
tv_nsec)) / 1.0e9;
172
173     if(print_matrix(stdout, c) == -1)
174         goto _exit_main;
175
176     fprintf(stderr, "time: %g\n", dt);
177

```

```

178         6 1 1 . 3 0 0 0 0
179         0 free(line);
179         6 1 1 3 0 0 0 0
180         0 destroy_matrix(a);
180         6 1 1 3 0 0 0 0
181         0 destroy_matrix(b);
181         6 0 0 3 0 0 0 0
182         0 destroy_matrix(c);
182         . . . . .
183         . . . }
183         . . . . .
184         3 1 1 0 0 0 0 0
185         0 return 0;
185         . . . . .
186         . . . . .
187         . _exit_main:
187         . . . . .
188         . fprintf(stderr, " at line %u\n", (
188         unsigned) lineno);
188         . . . . .
189         . free(line);
189         . . . . .
190         . destroy_matrix(a);
190         . . . . .
191         . destroy_matrix(b);
191         . . . . .
192         . destroy_matrix(c);
192         . . . . .
193         . exit(1);
193         6 1 1 2 0 0 0 0
194         0 }
194         . . . . .
195         . . . . .
196         . matrix_t*
196         . matrix_multiply(matrix_t* m1, matrix_t*
197         m2, int bs)
197         11 2 2 0 0 0 6 0
198         0 {
198         . . . . .
199         . size_t n, en, i, j, k, kk, jj;
199         . . . . .
200         . double sum;
200         . . . . .
201         . matrix_t* mr;
201         . . . . .
202         3 1 1 2 0 0 1 0
203         0 n = m1->rows;
203         . . . . .

```

```

204      11      1      1      .      5      0      0      1      0
      0      if(!(mr = create_matrix(n,n))) return
      NULL;
205      .      .      .      .      .      .      .      .
      .
206      9      2      2      .      3      0      0      1      0
      0      en = bs*(n/bs);
207      .      .      .      .      .      .      .      .
      .
208      48      2      2      .      17      0      0      6      0
      0      for(i=0; i<n; i++)
209      240      1      1      .      85      0      0      30      0
      0      for(j=0; j<n; j++)
210      275      1      1      .      125      0      0      50      5
      5      mr->array[i*n+j] = 0.0;
211      .      .      .      .      .      .      .      .
      .
212      .      .      .      .      .      .      .      .
      .      #if 1
213      .      .      .      .      .      .      .      .
      .      if (1) {
214      .      .      .      .      .      .      .      .
      .      size_t j;
215      2      0      0      .      0      0      0      1      0
      0      size_t dim = 1024*1024*10;
216      9      1      1      .      3      0      0      1      0
      0      int *v = malloc(dim*sizeof(int));
217      83,886,088      2      2      31,457,282      0      0      10,485,761      0
      0      for (j = 0; j < dim; ++j)
218      62,914,560      0      0      20,971,520      0      0      10,485,760      1,310,720
      1,310,720      v[j] = -1;
219      6      1      1      .      3      2      2      0      0
      0      free(v);
220      .      .      .      .      .      .      .      .
      .      }
221      .      .      .      .      .      .      .      .
      .      #endif
222      .      .      .      .      .      .      .      .
      .
223      17      2      2      .      6      0      0      2      0
      0      for(kk=0; kk<en; kk+=bs)
224      17      2      2      .      6      0      0      2      0
      0      for(jj=0; jj<en; jj+=bs)
225      48      1      1      .      17      0      0      6      0
      0      for(i=0; i<n; i++)
226      305      2      2      .      120      0      0      30      0
      0      for(j=jj; j<jj+bs; j++) {
227      275      1      1      .      150      7      7      25      0
      0      sum = mr->array[i*n+j];
228      1,525      3      3      .      600      0      0      150      0
      0      for(k=kk; k<kk+bs; k++)
229      3,000      2      2      .      1,625      14      14      125      0
      0      sum += m1->array[i*n+k] * m2->

```

```

230     array[k*n+j];
275     1    1    150    0    0    25    0
231         0    mr->array[i*n+j] = sum;
.    .    .    .    .    .    .
232     1    0    0    1    0    0    0    0
        0    return mr;
233     6    1    1    2    0    0    0    0
        0 }
234     .    .    .    .    .    .    .    .
235     .    .    .    .    .    .    .    .
        .    char*
236     .    .    .    .    .    .    .    .
        .    read_line(FILE *fp)
237     18   3    3    0    0    0    8    0
        0 {
238     .    .    .    .    .    .    .    .
        .    #define DEF_LINE_SZ 1024
239     .    .    .    .    .    .    .    .
        .    .
240     .    .    .    .    .    .    .    .
        .    int c;
241     6    0    0    0    0    0    4    0
        0    size_t len = 0, tam = DEF_LINE_SZ;
242     .    .    .    .    .    .    .    .
        .    char* str;
243     .    .    .    .    .    .    .    .
        .    .
244     14   1    1    6    1    0    2    1
        0    str = malloc(tam);
245     6    0    0    2    0    0    0    0
        0    if (!str) {
246     .    .    .    .    .    .    .    .
        .    perror("");
247     .    .    .    .    .    .    .    .
        .    return NULL;
248     .    .    .    .    .    .    .    .
        .    }
249     .    .    .    .    .    .    .    .
        .    .
250     1,541  5    5    514    1    0    103    0
        0    while (EOF != (c=fgetc(fp)) && c != '\n')
        {
251     808    2    2    303    0    0    202    3
        3    str[len++]=c;
252     505    0    0    202    0    0    0    0
        0    if (len==tam-1) {
253     .    .    .    .    .    .    .    .
        .    str = realloc(str, tam *= 2);
254     .    .    .    .    .    .    .    .
        .    if (!str) {
255     .    .    .    .    .    .    .    .
        .    perror("");

```

```

256         .         .         .         .         .         .         .         .
257         .         .         .         .         .         .         .         .
258         .         .         .         .         .         .         .         .
259         .         .         .         .         .         .         .         .
260         .         .         .         .         .         .         .         .
261         8         1         1         .         .         .         .         .
262         .         .         .         .         .         .         .         .
263         .         .         .         .         .         .         .         .
264         12        1         1         .         .         .         .         .
265         .         .         .         .         .         .         .         .
266         12        1         1         .         .         .         .         .
267         .         .         .         .         .         .         .         .
268         .         .         .         .         .         .         .         .
269         .         .         .         .         .         .         .         .
270         .         .         .         .         .         .         .         .
271         30        1         1         .         .         .         .         .
272         .         .         .         .         .         .         .         .
273         .         .         .         .         .         .         .         .
274         30        2         2         .         .         .         .         .
275         .         .         .         .         .         .         .         .
276         .         .         .         .         .         .         .         .
277         .         .         .         .         .         .         .         .
278         .         .         .         .         .         .         .         .
279         9         1         1         .         .         .         .         .
280         9         1         1         .         .         .         .         .
281         51        2         2         .         .         .         .         .
282         .         .         .         .         .         .         .         .

```

```

283         .         .         .         free(m);
284         .         .         .         perror("");
285         .         .         .         return NULL;
286         .         .         .         }
287         .         .         .         .         .         .         .         .         .
288         3         1         1         .         3         0         0         0         0
289         18        1         1         .         6         0         0         0         0
290         .         .         .         .         .         .         .         .         .
291         .         .         .         void
292         .         .         .         .         .         .         .         .         .
293         27        1         1         .         0         0         0         12         0
294         9         0         0         .         3         0         0         0         0
295         24        1         1         .         12        0         0         0         0
296         24        1         1         .         9         0         0         0         0
297         18        0         0         .         6         0         0         0         0
298         .         .         .         .         .         .         .         .         .
299         .         .         .         int
300         10        2         2         .         0         0         0         5         0
301         .         .         .         .         .         .         .         .         .
302         .         .         .         size_t i, j;
303         3         1         1         .         2         0         0         1         0
304         13        1         1         .         6         1         1         0         0
305         .         .         .         .         .         .         .         .         .
306         .         .         .         perror("");
307         .         .         .         .         .         .         .         .         .
308         48        2         2         .         17        0         0         6         0
309         .         .         .         .         .         .         .         .         .
310         .         .         .         return -1;
311         .         .         .         .         .         .         .         .         .
312         .         .         .         }
313         48        2         2         .         17        0         0         6         0
314         .         .         .         .         .         .         .         .         .
315         .         .         .         for(i=0; i<n; i++)

```



```

309      240      2      2      85      0      0      30      0
310      550      3      3      250      0      0      0      0
      0      if (fprintf(fp, " %g", m->array[i*n
      +j]) < 0) {
311      .      .      .      .      .      .      .      .
312      .      .      .      .      .      .      .      .
313      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
314      10      1      1      4      0      0      0      0
      0      if (fprintf(fp, "\n") < 0) {
315      .      .      .      .      .      .      .      .
316      .      .      .      .      .      .      .      .
317      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
318      1      1      1      0      0      0      0      0
      0      return 0;
319      6      1      1      2      0      0      0      0
      0      }

```

```

322 Ir      I1mr ILmr Dr      D1mr DLmr Dw      D1mw
      DLmw
323 -----
324 146,813,063 120 119 52,434,158 36 30 20,972,616 1,310,741
      1,310,739 events annotated

```

```

1 6 68 18 73 48 69 59 126 52 95 128 111 73 82 68 104 68 105 100
   134 76 81 150 129 82 150 77 118 115 171 110 145 53 74 146
   123 104
2 e et al.
3 ==641== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4 ==641== Command: /tmp/02-mmult
5 ==641== Parent PID: 597
6 ==641==
7 --641-- Warning: Cannot auto-detect cache config, using
   defaults.
8 --641-- Run with -v to see.
9 ==641==
10 ==641== I refs: 147,225,983
11 ==641== I1 misses: 2,407
12 ==641== LLi misses: 2,389
13 ==641== I1 miss rate: 0.00%
14 ==641== LLi miss rate: 0.00%
15 ==641==
16 ==641== D refs: 73,565,863 (52,539,343 rd +
   21,026,520 wr)

```

17	==641== D1 misses: 1,315,141 (3,775 rd + 1,311,366 wr)										
18	==641== LLd misses: 1,314,203 (2,925 rd + 1,311,278 wr)										
19	==641== D1 miss rate: 1.8% (0.0% + 6.2%)										
20	==641== LLd miss rate: 1.8% (0.0% + 6.2%)										
21	==641==										
22	==641== LL refs: 1,317,548 (6,182 rd + 1,311,366 wr)										
23	==641== LL misses: 1,316,592 (5,314 rd + 1,311,278 wr)										
24	==641== LL miss rate: 0.6% (0.0% + 6.2%)										
25	-----										
26	I1 cache: 32768 B, 32 B, 4-way associative										
27	D1 cache: 32768 B, 32 B, 2-way associative										
28	LL cache: 524288 B, 32 B, 8-way associative										
29	Command: /tmp/02-mmult										
30	Data file: cachegrind.out.641										
31	Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw										
32	Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw										
33	Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw										
34	Thresholds: 0.1 100 100 100 100 100 100 100 100										
35	Include dirs:										
36	User annotated: /root/CARPETA/tp2-2020-2q-src/main.c										
37	Auto-annotation: off										
38											
39	-----										
40	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw			
41		DLmw									
42	147,225,983 2,407 2,389 52,539,343 3,775 2,925 21,026,520										
43	1,311,366 1,311,278 PROGRAM TOTALS										
44	-----										
45	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw			
46		DLmw		file:function							
47	146,810,542 29 29 52,433,589 31 31 20,972,246										
48	1,310,728 1,310,728 /root/CARPETA/tp2-2020-2q-src/main.c:										
49	matrix_multiply										
50	-----										
50	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c										
51	-----										

```

52 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
    DLmw
53
54 -- line 18 -----
55 .          .          .          .          .          .          .
    .          matrix_t* create_matrix(size_t rows,
    size_t cols);
56 .          .          .          .          .          .          .
    .          .
57 .          .          .          .          .          .          .
    .          void destroy_matrix(matrix_t* m);
58 .          .          .          .          .          .          .
    .          .
59 .          .          .          .          .          .          .
    .          int print_matrix(FILE* fp, matrix_t* m)
    ;
60 .          .          .          .          .          .          .
    .          .
61 .          .          .          .          .          .          .
    .          int
62 .          .          .          .          .          .          .
    .          main(int argc, char** argv)
63 10      2      2          0      0      0          5          0
    0 {
64 .          .          .          .          .          .          .
    .          /* matrixes */
65 .          .          .          .          .          .          .
    .          matrix_t *a, *b, *c;
66 .          .          .          .          .          .          .
    .          /* n (dimension) and block size */
67 .          .          .          .          .          .          .
    .          size_t n, bs;
68 .          .          .          .          .          .          .
    .          /* line buffer (init to null to
simplify freeing on error) */
69 1      0      0          0      0      0          1          0
    0      char *line = NULL;
70 .          .          .          .          .          .          .
    .          /* line parsing auxiliar pointers */
71 .          .          .          .          .          .          .
    .          char *nptr, *endptr;
72 .          .          .          .          .          .          .
    .          /* auxiliar variables */
73 .          .          .          .          .          .          .
    .          long l;
74 .          .          .          .          .          .          .
    .          double e;
75 2      1      1          0      0      0          1          0
    0      size_t lineno = 1;
76 .          .          .          .          .          .          .
    .          struct timespec t0;
77 .          .          .          .          .          .          .
    .          struct timespec t1;
78 .          .          .          .          .          .          .

```

```

79         .         .         .         double dt;
80         .         .         .         size_t i;
81         25        4        4          9        0        0          1        0
82         10        1        1         for(; !feof(stdin); lineno++) {
83         .         .         .         a=b=c=NULL;
84         18        3        3          8        1        0          2        0
85         .         .         .         line = read_line(stdin);
86         6         0        0          2        0        0          0        0
87         9         0        0          4        0        0          0        0
88         .         .         .         if (!line) goto _exit_main;
89         .         .         .         if (line[0] == 0) break;
90         2         1        1          1        0        0          1        1
91         10        1        1          3        1        0          1        0
92         8         1        1          3        0        0          0        0
93         .         .         .         if (errno) {
94         .         .         .         perror("");
95         .         .         .         goto _exit_main;
96         4         2        2          2        0        0          0        0
97         .         .         .         if (nptr == endptr) {
98         .         .         .         fprintf(stderr, "missing
99         .         .         .         goto _exit_main;
100        3         2        2          1        0        0          0        0
101        .         .         .         if (1 < 1) {
102        .         .         .         fprintf(stderr, "invalid
103        .         .         .         goto _exit_main;
104        2         1        1          1        0        0          1        0

```

```

105         0      n = (size_t) 1;
106         .      .      .      .      .      .      .      .
107         .      .      .      .      .      .      .      .
108         .      .      .      .      .      .      .      .
109         .      .      .      .      .      .      .      .
110         .      .      .      .      .      .      .      .
111         .      .      .      .      .      .      .      .
112         .      .      .      .      .      .      .      .
113         .      .      .      .      .      .      .      .
114         .      .      .      .      .      .      .      .
115         .      .      .      .      .      .      .      .
116         .      .      .      .      .      .      .      .
117         .      .      .      .      .      .      .      .
118         .      .      .      .      .      .      .      .
119         .      .      .      .      .      .      .      .
120         .      .      .      .      .      .      .      .
121         .      .      .      .      .      .      .      .
122         .      .      .      .      .      .      .      .
123         .      .      .      .      .      .      .      .
124         .      .      .      .      .      .      .      .
125         .      .      .      .      .      .      .      .
126         .      .      .      .      .      .      .      .
127         .      .      .      .      .      .      .      .
128         .      .      .      .      .      .      .      .
129         .      .      .      .      .      .      .      .
130         .      .      .      .      .      .      .      .
131         .      .      .      .      .      .      .      .

        0      n = (size_t) 1;

        #if 0

        /* parse block size */

        nptr = endptr;

        l = strtol(nptr, &endptr, 10);

        if (errno) {

            perror("");

            goto _exit_main;

        }

-- line 75 -----
-- line 83 -----

        }

        bs = (size_t) 1;

        if (n % bs) {

            fprintf(stderr, "block size doesn
't match");

            goto _exit_main;

        }

        #else

2      0      0      1      0      0      1      0
0      bs = n;

        #endif

        /* load matrix a */
11      1      1      5      1      0      1      0
0      if (!(a = create_matrix(n, n)))
        goto _exit_main;

370     3      3      147     0      0      37      0
0      for (i=0; i < n*n; i++) {
131     72     0      0      36      0      0      36      0
0      nptr = endptr;

```

132	324	1	1	108	0	0	36	0
				e = strtod(np	tr, &endptr);			
133	288	1	1	108	0	0	0	0
			0	if (errno) {				
134
			.	perror("");				
135
			.	goto _exit_main;				
136
			.	}				
137	144	2	2	72	0	0	0	0
			0	if (np	tr == endptr) {			
138
			.	fprintf(stderr, "missing A				
139
			.	goto _exit_main;				
140
			.	}				
141	252	1	1	144	0	0	36	8
			8	a->array[i] = e;				
142
			.	}				
143
		
144
			.	/* load matrix b */				
145	11	1	1	5	0	0	1	0
			0	if (!(b = create_matrix(n, n))				
146
			.	goto _exit_main;				
147
		
148	370	2	2	147	0	0	37	0
			0	for (i=0; i < n*n; i++) {				
149	72	1	1	36	0	0	36	0
			0	np	tr = endptr;			
150	324	1	1	108	0	0	36	0
			0	e = strtod(np	tr, &endptr);			
151	288	1	1	108	0	0	0	0
			0	if (errno) {				
152
			.	perror("");				
153
			.	goto _exit_main;				
154
			.	}				
155	144	1	1	72	0	0	0	0
			0	if (np	tr == endptr) {			
156
			.	fprintf(stderr, "missing B				
157
			.	goto _exit_main;				

```

158         .      .      .      .      .      .      .      .
159         252     1      1      .      144     0      0      36      9
160             .      .      .      .      .      .      .      .
161             .      .      .      .      .      .      .      .
162             8      1      1      .      2      0      0      0      0
163             .      .      .      .      .      .      .      .
164             .      .      .      .      .      .      .      .
165             13     2      2      .      6      1      1      1      0
166             .      .      .      .      .      .      .      .
167             .      .      .      .      .      .      .      .
168             8      1      1      .      2      0      0      0      0
169             .      .      .      .      .      .      .      .
170             7      1      1      .      2      1      1      1      1
171             .      .      .      .      .      .      .      .
172             12     1      1      .      5      2      2      1      0
173             .      .      .      .      .      .      .      .
174             13     3      2      .      5      2      2      0      0
175             .      .      .      .      .      .      .      .
176             12     1      1      .      7      0      0      0      0
177             .      .      .      .      .      .      .      .
178             6      1      1      .      3      0      0      0      0
179             .      .      .      .      .      .      .      .
180             6      1      1      .      3      0      0      0      0
181             .      .      .      .      .      .      .      .
182             6      0      0      .      3      0      0      0      0
183             .      .      .      .      .      .      .      .

```

```

184      3      1      1      .      0      0      0      .      0      0
185      .      .      .      .      .      .      .      .      .      .
186      .      .      .      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .      .      .      .
192      .      .      .      .      .      .      .      .      .      .
193      6      1      1      .      2      0      0      .      0      0
194      .      .      .      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .      .      .      .
196      .      .      .      .      .      .      .      .      .      .
197      11     2      2      .      0      0      0      .      6      0
198      .      .      .      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .      .      .      .
202      3      1      1      .      2      0      0      .      1      0
203      .      .      .      .      .      .      .      .      .      .
204      11     1      1      .      5      0      0      .      1      0
205      .      .      .      .      .      .      .      .      .      .
206      9      2      2      .      3      0      0      .      1      0
207      .      .      .      .      .      .      .      .      .      .
208      56     2      2      .      20     0      0      .      7      0
      .      .      .      .      .      .      .      .      .      .

```



```

209      336      1      1      120      0      0      42      0
      0      for(j=0; j<n; j++)
210      396      1      1      180      0      0      72      8
      8      mr->array[i*n+j] = 0.0;
211      .      .      .      .      .      .      .      .
      .
212      .      .      .      .      .      .      .      .
      .      #if 1
213      .      .      .      .      .      .      .      .
      .      if (1) {
214      .      .      .      .      .      .      .      .
      .      size_t j;
215      2      0      0      0      0      0      1      0
      0      size_t dim = 1024*1024*10;
216      9      1      1      3      0      0      1      0
      0      int *v = malloc(dim*sizeof(int));
217 83,886,088      2      2 31,457,282      0      0 10,485,761      0
      0      for (j = 0; j < dim; ++j)
218 62,914,560      0      0 20,971,520      0      0 10,485,760 1,310,720
      1,310,720      v[j] = -1;
219      6      1      1      3      2      2      0      0
      0      free(v);
220      .      .      .      .      .      .      .      .
      .      }
221      .      .      .      .      .      .      .      .
      .      #endif
222      .      .      .      .      .      .      .      .
      .
223      17      2      2      6      0      0      2      0
      0      for(kk=0; kk<en; kk+=bs)
224      17      2      2      6      0      0      2      0
      0      for(jj=0; jj<en; jj+=bs)
225      56      1      1      20      0      0      7      0
      0      for(i=0; i<n; i++)
226      426      2      2      168      0      0      42      0
      0      for(j=jj; j<jj+bs; j++) {
227      396      1      1      216      10      10      36      0
      0      sum = mr->array[i*n+j];
228      2,556      3      3      1,008      0      0      252      0
      0      for(k=kk; k<kk+bs; k++)
229      5,184      2      2      2,808      19      19      216      0
      0      sum += m1->array[i*n+k] * m2->
      array[k*n+j];
230      396      1      1      216      0      0      36      0
      0      mr->array[i*n+j] = sum;
231      .      .      .      .      .      .      .      .
      .      }
232      1      0      0      1      0      0      0      0
      0      return mr;
233      6      1      1      2      0      0      0      0
      0 }
234      .      .      .      .      .      .      .      .
      .
235      .      .      .      .      .      .      .      .

```

```

236         . char*
237         . read_line(FILE *fp)
238         18 3 3 0 0 0 8 0
239         0 {
240         . #define DEF_LINE_SZ 1024
241         .
242         . int c;
243         6 0 0 0 0 0 4 0
244         0 size_t len = 0, tam = DEF_LINE_SZ;
245         . char* str;
246         .
247         14 1 1 6 1 0 2 1
248         0 str = malloc(tam);
249         6 0 0 2 0 0 0 0
250         0 if (!str) {
251         . perror("");
252         . return NULL;
253         . }
254         .
255         2,201 5 5 734 1 0 147 0
256         0 while (EOF != (c=fgetc(fp)) && c != '\n')
257         {
258         1,160 2 2 435 0 0 290 4
259         4 str[len++]=c;
260         725 0 0 290 0 0 0 0
261         0 if (len==tam-1) {
262         . str = realloc(str, tam *= 2);
263         . if (!str) {
264         . perror("");
265         . return NULL;
266         . }
267         . }
268         . }
269         .
270         8 1 1 2 0 0 0 0
271         0 if (c != EOF)

```

```

262         7    0    0         2    0    0         2    0
           0         str[len++]='\n';
263     .    .    .         .    .    .         .    .
           .
264     12    1    1         4    0    0         4    0
           0         str[len++]='\0';
265     2    0    0         2    0    0         0    0
           0         return str;
266     12    1    1         4    0    0         0    0
           0    }
267     .    .    .         .    .    .         .    .
268     .    .    .         .    .    .         .    .
           .
269     .    .    .         .    .    .         .    .
           .    matrix_t*
270     .    .    .         .    .    .         .    .
           .    create_matrix(size_t rows, size_t cols)
271     30    1    1         0    0    0         15    0
           0    {
272     .    .    .         .    .    .         .    .
           .    matrix_t * m;
273     .    .    .         .    .    .         .    .
           .
274     30    2    2         9    1    0         3    0
           0         if (!(m = malloc(sizeof(matrix_t)))) {
275     .    .    .         .    .    .         .    .
           .    perror("");
276     .    .    .         .    .    .         .    .
           .    return NULL;
277     .    .    .         .    .    .         .    .
           .    }
278     .    .    .         .    .    .         .    .
           .
279     9    1    1         6    0    0         3    0
           0         m->rows = rows;
280     9    1    1         6    0    0         3    0
           0         m->cols = cols;
281     51    2    2         21   0    0         3    0
           0         if (!(m->array = malloc(sizeof(double)
           *rows*cols))) {
282     .    .    .         .    .    .         .    .
           .    free(m);
283     .    .    .         .    .    .         .    .
           .    perror("");
284     .    .    .         .    .    .         .    .
           .    return NULL;
285     .    .    .         .    .    .         .    .
           .    }
286     .    .    .         .    .    .         .    .
           .
287     3    1    1         3    0    0         0    0
           0         return m;
288     18    1    1         6    0    0         0    0

```

```

289         0 }
290         . . . . .
291         . . . . .
292         . destroy_matrix(matrix_t* m)
27 1 1 0 0 0 12 0
293     9 0 0 { 3 0 0 0 0
294     24 1 1 12 0 0 0 0
295     24 1 1 9 0 0 0 0
296     18 0 0 6 0 0 0 0
297         0 }
298         . . . . .
299         . int
300         . print_matrix(FILE* fp, matrix_t* m)
301     10 2 2 0 0 0 5 0
302     0 {
303         . size_t i, j;
304         . size_t n;
305     3 1 1 2 0 0 1 0
306     13 1 1 6 1 1 0 0
307         0 if (fprintf(fp, "%lu", (unsigned long)
308         m->rows) < 0) {
309         . perror("");
310         . return -1;
311         . }
312     56 2 2 20 0 0 7 0
313     336 2 2 120 0 0 42 0
314     792 3 3 360 0 0 0 0
315         0 if (fprintf(fp, " %g", m->array[i*n
316         +j]) < 0) {
317         . perror("");
318         . return -1;
319         . }
320     10 1 1 4 0 0 0 0

```

315
316
317
318	1	1	1	0	0	0	0	0	0	0
319	6	1	1	2	0	0	0	0	0	0
320										
321	-----									
322	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw		D1mw	
323		DLmw								
324	146,819,332	120	119	52,436,987	44	38	20,973,112	1,310,752		
	1,310,750		events	annotated						

```

1  7 56 84 81 58 109 114 54 63 131 99 85 146 137 153 103 107 142
   50 174 152 80 61 102 87 44 122 120 57 110 143 111 93 153
   139 116 90 106 116 61 141 128 92 55 30 64 51 74 71 74
2  bVEX; rerun with -h for copyright info
3  ==644== Command: /tmp/02-mmult
4  ==644== Parent PID: 597
5  ==644==
6  --644-- Warning: Cannot auto-detect cache config, using
   defaults.
7  --644--          Run with -v to see.
8  ==644==
9  ==644== I      refs:          147,273,033
10 ==644== I1  misses:           2,400
11 ==644== LLi misses:           2,382
12 ==644== I1  miss rate:         0.00%
13 ==644== LLi miss rate:         0.00%
14 ==644==
15 ==644== D      refs:          73,583,930 (52,551,899 rd  +
   21,032,031 wr)
16 ==644== D1  misses:           1,315,005 (    3,660 rd  +
   1,311,345 wr)
17 ==644== LLd misses:           1,314,228 (    2,936 rd  +
   1,311,292 wr)
18 ==644== D1  miss rate:         1.8% (    0.0%      +
   6.2%   )
19 ==644== LLd miss rate:         1.8% (    0.0%      +
   6.2%   )
20 ==644==
21 ==644== LL refs:           1,317,405 (    6,060 rd  +
   1,311,345 wr)
22 ==644== LL misses:           1,316,610 (    5,318 rd  +
   1,311,292 wr)
23 ==644== LL miss rate:         0.6% (    0.0%      +

```

24	6.2%)	
25	I1 cache: 32768 B, 32 B, 4-way associative	
26	D1 cache: 32768 B, 32 B, 2-way associative	
27	LL cache: 524288 B, 32 B, 8-way associative	
28	Command: /tmp/02-mmult	
29	Data file: cachegrind.out.644	
30	Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw	
31	Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw	
32	Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw	
33	Thresholds: 0.1 100 100 100 100 100 100 100	
34	Include dirs:	
35	User annotated: /root/CARPETA/tp2-2020-2q-src/main.c	
36	Auto-annotation: off	
37		
38		
39	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
40	DLmw	
41	147,273,033 2,400 2,382 52,551,899 3,660 2,936 21,032,031	
42	1,311,345 1,311,292 PROGRAM TOTALS	
43		
44	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
45	DLmw file:function	
46	146,815,701 29 29 52,436,124 41 41 20,972,595	
47	1,310,731 1,310,731 /root/CARPETA/tp2-2020-2q-src/main.c:	
48	matrix_multiply	
49	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c	
50		
51	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
52	DLmw	
53	-- line 18 -----	
54	
55 matrix_t* create_matrix(size_t rows,	
56 size_t cols);	
57 void destroy_matrix(matrix_t* m);	
58	

```

.   int print_matrix(FILE* fp, matrix_t* m)
.   ;
59 .   .   .   .   .   .   .   .
60 .   .   .   .   .   .   .   .
.   int
61 .   .   .   .   .   .   .   .
.   main(int argc, char** argv)
62 10   2   2   0   0   0   5   0
.   {
63 .   .   .   .   .   .   .   .
.   /* matrixes */
64 .   .   .   .   .   .   .   .
.   matrix_t *a, *b, *c;
65 .   .   .   .   .   .   .   .
.   /* n (dimension) and block size */
66 .   .   .   .   .   .   .   .
.   size_t n, bs;
67 .   .   .   .   .   .   .   .
.   /* line buffer (init to null to
simplify freeing on error) */
68 1   0   0   0   0   0   1   0
.   char *line = NULL;
69 .   .   .   .   .   .   .   .
.   /* line parsing auxiliar pointers */
70 .   .   .   .   .   .   .   .
.   char *nptr, *endptr;
71 .   .   .   .   .   .   .   .
.   /* auxiliar variables */
72 .   .   .   .   .   .   .   .
.   long l;
73 .   .   .   .   .   .   .   .
.   double e;
74 2   1   1   0   0   0   1   0
.   size_t lineno = 1;
75 .   .   .   .   .   .   .   .
.   struct timespec t0;
76 .   .   .   .   .   .   .   .
.   struct timespec t1;
77 .   .   .   .   .   .   .   .
.   double dt;
78 .   .   .   .   .   .   .   .
.   size_t i;
79 .   .   .   .   .   .   .   .
.   .
80 25   4   4   9   0   0   1   0
.   for(;; !feof(stdin); lineno++) {
81 10   1   1   4   0   0   6   0
.   a=b=c=NULL;
82 .   .   .   .   .   .   .   .
.   .
83 18   3   3   8   1   0   2   0
.   line = read_line(stdin);
84 .   .   .   .   .   .   .   .

```

```

85      6      0      0      .      2      0      0      0      0
      0      if (!line) goto _exit_main;
86      9      0      0      4      0      0      0      0
      0      if (line[0] == 0) break;
87      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
88      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .
89      2      1      1      .      /* parse dimension */
      0      1      0      0      1      1
90      10     1      1      0      nptr = line;
      0      3      1      0      1      0
91      8      1      1      0      l = strtol(nptr, &endptr, 10);
      0      3      0      0      0      0
92      .      .      .      .      if (errno) {
      .      .      .      .      perror("");
93      .      .      .      .      goto _exit_main;
94      .      .      .      .      }
95      4      2      2      2      0      0      0      0
      0      if (nptr == endptr) {
96      .      .      .      .      fprintf(stderr, "missing
dimension");
97      .      .      .      .      goto _exit_main;
98      .      .      .      .      }
99      3      2      2      1      0      0      0      0
      0      if (l < 1) {
100     .      .      .      .      fprintf(stderr, "invalid
dimension");
101     .      .      .      .      goto _exit_main;
102     .      .      .      .      }
103     2      1      1      1      0      0      1      0
      0      n = (size_t) l;
104     .      .      .      .      .
105     .      .      .      .      #if 0
106     .      .      .      .      /* parse block size */
107     .      .      .      .      nptr = endptr;
108     .      .      .      .      l = strtol(nptr, &endptr, 10);
109     .      .      .      .      if (errno) {
110     .      .      .      .      perror("");

```



```

111         .      .      .      goto _exit_main;
112         .      .      }
113 -- line 75 -----
114         .      .      .      }
115         .      .      .      bs = (size_t) l;
116         .      .      .
117         .      .      .      if (n % bs) {
118         .      .      .          fprintf(stderr, "block size doesn
't match");
119         .      .      .          goto _exit_main;
120         .      .      .      }
121         .      .      .      #else
122         2      0      0          1      0      0          1      0
0          bs = n;
123         .      .      .
124         .      .      .      #endif
125         .      .      .
126         11     1      1          /* load matrix a */
0          5      1      0          1      0
if (!(a = create_matrix(n, n)))
127         .      .      .      goto _exit_main;
128         .      .      .
129         500    3      3          199    0      0          50      0
0          for (i=0; i < n*n; i++) {
130         98     0      0          49     0      0          49      0
0          nptr = endptr;
131         441    1      1          147    0      0          49      0
0          e = strtod(nptr, &endptr);
132         392    1      1          147    0      0          0        0
0          if (errno) {
133         .      .      .          perror("");
134         .      .      .          goto _exit_main;
135         .      .      .      }
136         196    2      2          98     0      0          0        0
0          if (nptr == endptr) {
137         .      .      .          fprintf(stderr, "missing A

```

```

138         .         .         .         .         .         .         .         .         .         .
139         .         .         .         .         .         .         .         .         .         .
140         343      1      1         196      0      0         49         11
141         .         .         .         .         .         .         .         .         .         .
142         .         .         .         .         .         .         .         .         .         .
143         .         .         .         .         .         .         .         .         .         .
144         11      1      1         5      0      0         1         0
145         .         .         .         .         .         .         .         .         .         .
146         .         .         .         .         .         .         .         .         .         .
147         500     2      2         199     0      0         50         0
148         98      1      1         49     0      0         49         0
149         441     1      1         147     0      0         49         0
150         392     1      1         147     0      0         0         0
151         .         .         .         .         .         .         .         .         .         .
152         .         .         .         .         .         .         .         .         .         .
153         .         .         .         .         .         .         .         .         .         .
154         196     1      1         98     0      0         0         0
155         .         .         .         .         .         .         .         .         .         .
156         .         .         .         .         .         .         .         .         .         .
157         .         .         .         .         .         .         .         .         .         .
158         343     1      1         196     0      0         49         11
159         .         .         .         .         .         .         .         .         .         .
160         .         .         .         .         .         .         .         .         .         .
161         8       1      1         2      0      0         0         0
162         .         .         .         .         .         .         .         .         .         .
163         .         .         .         .         .         .         .         .         .         .

```

```

164      13      2      2      6      1      1      1      0
        0      if (!(c = matrix_multiply(a, b, bs))
    )
165      .      .      .      .      .      .      .      .
        .      goto _exit_main;
166      .      .      .      .      .      .      .      .
        .      .
167      8      1      1      2      0      0      0      0
        0      clock_gettime(CLOCK_REALTIME, &t1);
168      .      .      .      .      .      .      .      .
        .      .
169      7      1      1      2      1      1      1      1
        1      dt = (float) (t1.tv_sec - t0.tv_sec
    );
170      12      1      1      5      2      2      1      0
        0      dt = dt + ((float)(t1.tv_nsec - t0.
tv_nsec)) / 1.0e9;
171      .      .      .      .      .      .      .      .
        .      .
172      13      3      2      5      2      2      0      0
        0      if(print_matrix(stdout, c) == -1)
173      .      .      .      .      .      .      .      .
        .      goto _exit_main;
174      .      .      .      .      .      .      .      .
        .      .
175      12      1      1      7      0      0      0      0
        0      fprintf(stderr, "time: %g\n", dt);
176      .      .      .      .      .      .      .      .
        .      .
177      6      1      1      3      0      0      0      0
        0      free(line);
178      6      1      1      3      0      0      0      0
        0      destroy_matrix(a);
179      6      1      1      3      0      0      0      0
        0      destroy_matrix(b);
180      6      0      0      3      0      0      0      0
        0      destroy_matrix(c);
181      .      .      .      .      .      .      .      .
        .      }
182      .      .      .      .      .      .      .      .
        .      .
183      3      1      1      0      0      0      0      0
        0      return 0;
184      .      .      .      .      .      .      .      .
        .      .
185      .      .      .      .      .      .      .      .
        .      _exit_main:
186      .      .      .      .      .      .      .      .
        .      fprintf(stderr, " at line %u\n", (
unsigned) lineno);
187      .      .      .      .      .      .      .      .
        .      free(line);
188      .      .      .      .      .      .      .      .
        .      destroy_matrix(a);

```

```

189         .      .      .      .      .      .      .      .      .
190         .      .      .      .      .      .      .      .      .
191         .      .      .      .      .      .      .      .      .
192         6      1      1      .      .      .      .      .      .
193         .      .      .      .      .      .      .      .      .
194         .      .      .      .      .      .      .      .      .
195         .      .      .      .      .      .      .      .      .
196         11     2      2      .      .      .      .      .      .
197         .      .      .      .      .      .      .      .      .
198         .      .      .      .      .      .      .      .      .
199         .      .      .      .      .      .      .      .      .
200         .      .      .      .      .      .      .      .      .
201         3      1      1      .      .      .      .      .      .
202         .      .      .      .      .      .      .      .      .
203         11     1      1      .      .      .      .      .      .
204         .      .      .      .      .      .      .      .      .
205         9      2      2      .      .      .      .      .      .
206         .      .      .      .      .      .      .      .      .
207         64     2      2      .      .      .      .      .      .
208         448    1      1      .      .      .      .      .      .
209         539    1      1      .      .      .      .      .      .
210         .      .      .      .      .      .      .      .      .
211         .      .      .      .      .      .      .      .      .
212         .      .      .      .      .      .      .      .      .
213         .      .      .      .      .      .      .      .      .
214         2      0      0      .      .      .      .      .      .

```

```

215      9      1      1          3      0      0          1      0
           0      int *v = malloc(dim*sizeof(int));
216 83,886,088      2      2 31,457,282      0      0 10,485,761      0
           0      for (j = 0; j < dim; ++j)
217 62,914,560      0      0 20,971,520      0      0 10,485,760 1,310,720
           1,310,720      v[j] = -1;
218      6      1      1          3      2      2          0      0
           0      free(v);
219      .      .      .      .      .      .      .      .
           .      }
220      .      .      .      .      .      .      .      .
           .      #endif
221      .      .      .      .      .      .      .      .
           .
222      17      2      2          6      0      0          2      0
           0      for(kk=0; kk<en; kk+=bs)
223      17      2      2          6      0      0          2      0
           0      for(jj=0; jj<en; jj+=bs)
224      64      1      1          23      0      0          8      0
           0      for(i=0; i<n; i++)
225      567      2      2          224      0      0          56      0
           0      for(j=jj; j<jj+bs; j++) {
226      539      1      1          294      13      13          49      0
           0      sum = mr->array[i*n+j];
227      3,969      3      3          1,568      0      0          392      0
           0      for(k=kk; k<kk+bs; k++)
228      8,232      2      2          4,459      26      26          343      0
           0      sum += m1->array[i*n+k] * m2->
           array[k*n+j];
229      539      1      1          294      0      0          49      0
           0      mr->array[i*n+j] = sum;
230      .      .      .      .      .      .      .      .
           .      }
231      1      0      0          1      0      0          0      0
           0      return mr;
232      6      1      1          2      0      0          0      0
           0 }
233      .      .      .      .      .      .      .      .
           .
234      .      .      .      .      .      .      .      .
           .      char*
235      .      .      .      .      .      .      .      .
           .      read_line(FILE *fp)
236      18      3      3          0      0      0          8      0
           0 {
237      .      .      .      .      .      .      .      .
           .      #define DEF_LINE_SZ 1024
238      .      .      .      .      .      .      .      .
           .
239      .      .      .      .      .      .      .      .
           .      int c;
240      6      0      0          0      0      0          4      0
           0      size_t len = 0, tam = DEF_LINE_SZ;
241      .      .      .      .      .      .      .      .

```

```

242         char* str;
243         14 1 1 6 1 0 2 1
244         0 str = malloc(tam);
245         6 0 0 2 0 0 0 0
246         0 if (!str) {
247         . . . . .
248         . . . . .
249         . . . . .
250         . . . . .
251         . . . . .
252         . . . . .
253         . . . . .
254         . . . . .
255         . . . . .
256         . . . . .
257         . . . . .
258         . . . . .
259         . . . . .
260         8 1 1 2 0 0 0 0
261         0 if (c != EOF)
262         7 0 0 2 0 0 2 0
263         0 str[len++] = '\n';
264         . . . . .
265         12 1 1 4 0 0 4 0
266         0 str[len++] = '\0';
267         2 0 0 2 0 0 0 0
268         0 return str;
269         12 1 1 4 0 0 0 0
270         0 }
271         . . . . .
272         . . . . .

```

```

268      .      .      .      .      .      .      .      .
269      .      .      .      .      .      .      .      .
270      30      1      1      0      0      0      15      0
271      .      .      .      .      .      .      .      .
272      .      .      .      .      .      .      .      .
273      30      2      2      9      1      0      3      0
274      .      .      .      .      .      .      .      .
275      .      .      .      .      .      .      .      .
276      .      .      .      .      .      .      .      .
277      .      .      .      .      .      .      .      .
278      9      1      1      6      0      0      3      0
279      9      1      1      6      0      0      3      0
280      51      2      2      21     0      0      3      0
281      .      .      .      .      .      .      .      .
282      .      .      .      .      .      .      .      .
283      .      .      .      .      .      .      .      .
284      .      .      .      .      .      .      .      .
285      .      .      .      .      .      .      .      .
286      3      1      1      3      0      0      0      0
287      18      1      1      6      0      0      0      0
288      .      .      .      .      .      .      .      .
289      .      .      .      .      .      .      .      .
290      .      .      .      .      .      .      .      .
291      27      1      1      0      0      0      12      0
292      9      0      0      3      0      0      0      0
293      24      1      1      12     0      0      0      0
294      24      1      1      9      0      0      0      0

```

```

295         0      free(m);
18      0      0      6      0      0      0      0
296         0  }
.      .      .      .      .      .      .      .
297         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
298         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
299         10     2     2     0     0     0     5     0
300         0  {
.      .      .      .      .      .      .      .
301         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
302         3     1     1     2     0     0     1     0
0     n = m->rows;
303         13     1     1     6     1     1     0     0
0     if (fprintf(fp, "%lu", (unsigned long)
m->rows) < 0) {
304         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
305         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
306         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
307         64     2     2     23     0     0     8     0
0     for(i=0; i<n; i++)
308         448     2     2     161     0     0     56     0
0     for (j=0; j<n; j++)
309         1,078     3     3     490     0     0     0     0
0     if (fprintf(fp, " %g", m->array[i*n+j
]) < 0) {
310         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
311         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
312         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
313         10     1     1     4     0     0     0     0
0     if (fprintf(fp, "\n") < 0) {
314         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
315         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
316         .      .      .      .      .      .      .      .
.      .      .      .      .      .      .      .
317         1     1     1     0     0     0     0     0
0     return 0;
318         6     1     1     2     0     0     0     0
0  }
319
320

```


321	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw
		DLmw						
322	-----							
323	146,827,393	120	119	52,440,658	54	48	20,973,736	1,310,762
	1,310,760 events annotated							
1	8 240 184 111 110 129 175 191 230 105 183 118 82 140 181 172 148 179 136 110 107 151 113 116 165 185 178 143 170 219 133 177 202 240 232 166 119 161 187 203 268 181 158 128 109 199 101 128 205 229 254 165 163 208 223 208 251 178 181 135 176 205 159 197 186							
2	PID: 597							
3	==647==							
4	--647-- Warning: Cannot auto-detect cache config, using defaults.							
5	--647-- Run with -v to see.							
6	==647==							
7	==647== I refs: 147,337,400							
8	==647== I1 misses: 2,405							
9	==647== LLi misses: 2,387							
10	==647== I1 miss rate: 0.00%							
11	==647== LLi miss rate: 0.00%							
12	==647==							
13	==647== D refs: 73,608,465 (52,568,673 rd + 21,039,792 wr)							
14	==647== D1 misses: 1,315,252 (3,828 rd + 1,311,424 wr)							
15	==647== LLd misses: 1,314,257 (2,949 rd + 1,311,308 wr)							
16	==647== D1 miss rate: 1.8% (0.0% + 6.2%)							
17	==647== LLd miss rate: 1.8% (0.0% + 6.2%)							
18	==647==							
19	==647== LL refs: 1,317,657 (6,233 rd + 1,311,424 wr)							
20	==647== LL misses: 1,316,644 (5,336 rd + 1,311,308 wr)							
21	==647== LL miss rate: 0.6% (0.0% + 6.2%)							
22	-----							
23	I1 cache: 32768 B, 32 B, 4-way associative							
24	D1 cache: 32768 B, 32 B, 2-way associative							
25	LL cache: 524288 B, 32 B, 8-way associative							
26	Command: /tmp/02-mmult							
27	Data file: cachegrind.out.647							
28	Events recorded: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw							
29	Events shown: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw							
30	Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw							
31	Thresholds: 0.1 100 100 100 100 100 100 100 100							
32	Include dirs:							
33	User annotated: /root/CARPETA/tp2-2020-2q-src/main.c							

34	Auto-annotation: off	
35		
36	-----	-----
37	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
	DLmw	
38	-----	-----
39	147,337,400 2,405 2,387 52,568,673 3,828 2,949 21,039,792	
40	1,311,424 1,311,308 PROGRAM TOTALS	
41	-----	-----
42	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
	DLmw file:function	
43	-----	-----
44	146,822,412 29 29 52,439,429 52 52 20,973,042	
	1,310,735 1,310,735 /root/CARPETA/tp2-2020-2q-src/main.c:	
	matrix_multiply	
45	-----	-----
46		
47	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c	
48	-----	-----
49	Ir I1mr ILmr Dr D1mr DLmr Dw D1mw	
	DLmw	
50		
51	-- line 18 -----	
52	
	. matrix_t* create_matrix(size_t rows,	
	size_t cols);	
53	
54	
	. void destroy_matrix(matrix_t* m);	
55	
56	
	. int print_matrix(FILE* fp, matrix_t* m)	
	;	
57	
58	
	. int	
59	
	. main(int argc, char** argv)	
60	10 2 2 0 0 0 5 0	
	0 {	
61	
	. /* matrixes */	
62	
	. matrix_t *a, *b, *c;	

```

63      .      .      .      .      .      .      .      .      .      .
64      .      .      .      .      .      .      .      .      .      .
65      .      .      .      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .      .      .
69      .      .      .      .      .      .      .      .      .      .
70      .      .      .      .      .      .      .      .      .      .
71      .      .      .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .      .      .
76      .      .      .      .      .      .      .      .      .      .
77      .      .      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .      .      .
82      .      .      .      .      .      .      .      .      .      .
83      .      .      .      .      .      .      .      .      .      .
84      .      .      .      .      .      .      .      .      .      .
85      .      .      .      .      .      .      .      .      .      .
86      .      .      .      .      .      .      .      .      .      .
87      .      .      .      .      .      .      .      .      .      .
88      .      .      .      .      .      .      .      .      .      .
89      .      .      .      .      .      .      .      .      .      .

/* n (dimension) and block size */
size_t n, bs;

/* line buffer (init to null to
simplify freeing on error) */
char *line = NULL;

/* line parsing auxiliar pointers */
char *nptr, *endptr;

/* auxiliar variables */
long l;
double e;
size_t lineno = 1;
struct timespec t0;
struct timespec t1;
double dt;
size_t i;

for(; !feof(stdin); lineno++) {
    a=b=c=NULL;

    line = read_line(stdin);

    if (!line) goto _exit_main;
    if (line[0] == 0) break;

    /* parse dimension */
    nptr = line;
    l = strtol(nptr, &endptr, 10);

```

```

90         0      if (errno) {
91             .      perror("");
92             .      goto _exit_main;
93         4      2      2      }
94         0      if (nptr == endptr) {
95             .      fprintf(stderr, "missing
dimension");
96             .      goto _exit_main;
97         3      2      2      }
98         0      if (l < 1) {
99             .      fprintf(stderr, "invalid
dimension");
100            .      goto _exit_main;
101        2      1      1      }
102            0      n = (size_t) l;
103            .      #if 0
104            .      /* parse block size */
105            .      nptr = endptr;
106            .      l = strtoul(nptr, &endptr, 10);
107            .      if (errno) {
108            .          perror("");
109            .          goto _exit_main;
110        -- line 75 -----
111        -- line 83 -----
112            .      }
113            .      bs = (size_t) l;
114            .
115            .      if (n % bs) {
116            .

```

```

        fprintf(stderr, "block size doesn
        't match");
117         .         .         .         .         .         .         .
        .         .         .         goto _exit_main;         .         .
118         .         .         .         .         .         .         .
        .         .         .         }         .         .         .
119         .         .         .         .         .         .         .
        .         .         .         #else         .         .         .
120         2         0         0         1         0         0         1         0
        .         .         .         .         .         .         .
121         .         .         .         .         .         .         .
        .         .         .         #endif         .         .         .
122         .         .         .         .         .         .         .
        .         .         .         .         .         .         .
123         .         .         .         .         .         .         .
        .         .         .         /* load matrix a */         .         .
124         11         1         1         5         1         0         1         0
        .         .         .         0         if (!(a = create_matrix(n, n)))
125         .         .         .         .         .         .         .
        .         .         .         goto _exit_main;         .         .
126         .         .         .         .         .         .         .
        .         .         .         .         .         .         .
127         650        3         3         259        0         0         65         0
        .         .         .         for (i=0; i < n*n; i++) {
128         128        0         0         64         0         0         64         0
        .         .         .         nptr = endptr;
129         576        1         1         192        0         0         64         0
        .         .         .         0         e = strtod(nptr, &endptr);
130         512        1         1         192        0         0         0         0
        .         .         .         0         if (errno) {
131         .         .         .         .         .         .         .
        .         .         .         perror("");         .         .
132         .         .         .         .         .         .         .
        .         .         .         goto _exit_main;         .         .
133         .         .         .         .         .         .         .
        .         .         .         }         .         .         .
134         256        2         2         128        0         0         0         0
        .         .         .         0         if (nptr == endptr) {
135         .         .         .         .         .         .         .
        .         .         .         fprintf(stderr, "missing A
        matrix element");
136         .         .         .         .         .         .         .
        .         .         .         goto _exit_main;         .         .
137         .         .         .         .         .         .         .
        .         .         .         }         .         .         .
138         448        1         1         256        0         0         64         15
        .         .         .         15         a->array[i] = e;
139         .         .         .         .         .         .         .
        .         .         .         }         .         .         .
140         .         .         .         .         .         .         .
        .         .         .         .         .         .         .
141         .         .         .         .         .         .         .
        .         .         .         /* load matrix b */         .         .
142         11         1         1         5         0         0         1         0

```

```

143         0      if (!(b = create_matrix(n, n)))
144             .      goto _exit_main;
145         650    2    2      259    0    0      65    0
146             0      for (i=0; i < n*n; i++) {
147         128    1    1      64    0    0      64    0
148             0      nptr = endptr;
149         576    1    1      192    0    0      64    0
150             0      e = strtod(nptr, &endptr);
151         512    1    1      192    0    0      0    0
152             0      if (errno) {
153             .      perror("");
154             .      goto _exit_main;
155             .      }
156         256    1    1      128    0    0      0    0
157             0      if (nptr == endptr) {
158             .      fprintf(stderr, "missing B
159             .      matrix element");
160             .      goto _exit_main;
161             .      }
162         448    1    1      256    0    0      64    16
163             16      b->array[i] = e;
164             .      }
165             .      }
166         8    1    1      2    0    0      0    0
167             0      clock_gettime(CLOCK_REALTIME, &t0);
168             .      .
169             .      .
170             .      .
171             .      .
172             .      .
173             .      .
174             .      .
175             .      .
176             .      .
177             .      .
178             .      .
179             .      .
180             .      .
181             .      .
182             .      .
183             .      .
184             .      .
185             .      .
186             .      .
187             .      .
188             .      .
189             .      .
190             .      .
191             .      .
192             .      .
193             .      .
194             .      .
195             .      .
196             .      .
197             .      .
198             .      .
199             .      .
200             .      .
201             .      .
202             .      .
203             .      .
204             .      .
205             .      .
206             .      .
207             .      .
208             .      .
209             .      .
210             .      .
211             .      .
212             .      .
213             .      .
214             .      .
215             .      .
216             .      .
217             .      .
218             .      .
219             .      .
220             .      .
221             .      .
222             .      .
223             .      .
224             .      .
225             .      .
226             .      .
227             .      .
228             .      .
229             .      .
230             .      .
231             .      .
232             .      .
233             .      .
234             .      .
235             .      .
236             .      .
237             .      .
238             .      .
239             .      .
240             .      .
241             .      .
242             .      .
243             .      .
244             .      .
245             .      .
246             .      .
247             .      .
248             .      .
249             .      .
250             .      .
251             .      .
252             .      .
253             .      .
254             .      .
255             .      .
256             .      .
257             .      .
258             .      .
259             .      .
260             .      .
261             .      .
262             .      .
263             .      .
264             .      .
265             .      .
266             .      .
267             .      .
268             .      .
269             .      .
270             .      .
271             .      .
272             .      .
273             .      .
274             .      .
275             .      .
276             .      .
277             .      .
278             .      .
279             .      .
280             .      .
281             .      .
282             .      .
283             .      .
284             .      .
285             .      .
286             .      .
287             .      .
288             .      .
289             .      .
290             .      .
291             .      .
292             .      .
293             .      .
294             .      .
295             .      .
296             .      .
297             .      .
298             .      .
299             .      .
300             .      .
301             .      .
302             .      .
303             .      .
304             .      .
305             .      .
306             .      .
307             .      .
308             .      .
309             .      .
310             .      .
311             .      .
312             .      .
313             .      .
314             .      .
315             .      .
316             .      .
317             .      .
318             .      .
319             .      .
320             .      .
321             .      .
322             .      .
323             .      .
324             .      .
325             .      .
326             .      .
327             .      .
328             .      .
329             .      .
330             .      .
331             .      .
332             .      .
333             .      .
334             .      .
335             .      .
336             .      .
337             .      .
338             .      .
339             .      .
340             .      .
341             .      .
342             .      .
343             .      .
344             .      .
345             .      .
346             .      .
347             .      .
348             .      .
349             .      .
350             .      .
351             .      .
352             .      .
353             .      .
354             .      .
355             .      .
356             .      .
357             .      .
358             .      .
359             .      .
360             .      .
361             .      .
362             .      .
363             .      .
364             .      .
365             .      .
366             .      .
367             .      .
368             .      .
369             .      .
370             .      .
371             .      .
372             .      .
373             .      .
374             .      .
375             .      .
376             .      .
377             .      .
378             .      .
379             .      .
380             .      .
381             .      .
382             .      .
383             .      .
384             .      .
385             .      .
386             .      .
387             .      .
388             .      .
389             .      .
390             .      .
391             .      .
392             .      .
393             .      .
394             .      .
395             .      .
396             .      .
397             .      .
398             .      .
399             .      .
400             .      .
401             .      .
402             .      .
403             .      .
404             .      .
405             .      .
406             .      .
407             .      .
408             .      .
409             .      .
410             .      .
411             .      .
412             .      .
413             .      .
414             .      .
415             .      .
416             .      .
417             .      .
418             .      .
419             .      .
420             .      .
421             .      .
422             .      .
423             .      .
424             .      .
425             .      .
426             .      .
427             .      .
428             .      .
429             .      .
430             .      .
431             .      .
432             .      .
433             .      .
434             .      .
435             .      .
436             .      .
437             .      .
438             .      .
439             .      .
440             .      .
441             .      .
442             .      .
443             .      .
444             .      .
445             .      .
446             .      .
447             .      .
448             .      .
449             .      .
450             .      .
451             .      .
452             .      .
453             .      .
454             .      .
455             .      .
456             .      .
457             .      .
458             .      .
459             .      .
460             .      .
461             .      .
462             .      .
463             .      .
464             .      .
465             .      .
466             .      .
467             .      .
468             .      .
469             .      .
470             .      .
471             .      .
472             .      .
473             .      .
474             .      .
475             .      .
476             .      .
477             .      .
478             .      .
479             .      .
480             .      .
481             .      .
482             .      .
483             .      .
484             .      .
485             .      .
486             .      .
487             .      .
488             .      .
489             .      .
490             .      .
491             .      .
492             .      .
493             .      .
494             .      .
495             .      .
496             .      .
497             .      .
498             .      .
499             .      .
500             .      .
501             .      .
502             .      .
503             .      .
504             .      .
505             .      .
506             .      .
507             .      .
508             .      .
509             .      .
510             .      .
511             .      .
512             .      .
513             .      .
514             .      .
515             .      .
516             .      .
517             .      .
518             .      .
519             .      .
520             .      .
521             .      .
522             .      .
523             .      .
524             .      .
525             .      .
526             .      .
527             .      .
528             .      .
529             .      .
530             .      .
531             .      .
532             .      .
533             .      .
534             .      .
535             .      .
536             .      .
537             .      .
538             .      .
539             .      .
540             .      .
541             .      .
542             .      .
543             .      .
544             .      .
545             .      .
546             .      .
547             .      .
548             .      .
549             .      .
550             .      .
551             .      .
552             .      .
553             .      .
554             .      .
555             .      .
556             .      .
557             .      .
558             .      .
559             .      .
560             .      .
561             .      .
562             .      .
563             .      .
564             .      .
565             .      .
566             .      .
567             .      .
568             .      .
569             .      .
570             .      .
571             .      .
572             .      .
573             .      .
574             .      .
575             .      .
576             .      .
577             .      .
578             .      .
579             .      .
580             .      .
581             .      .
582             .      .
583             .      .
584             .      .
585             .      .
586             .      .
587             .      .
588             .      .
589             .      .
590             .      .
591             .      .
592             .      .
593             .      .
594             .      .
595             .      .
596             .      .
597             .      .
598             .      .
599             .      .
600             .      .
601             .      .
602             .      .
603             .      .
604             .      .
605             .      .
606             .      .
607             .      .
608             .      .
609             .      .
610             .      .
611             .      .
612             .      .
613             .      .
614             .      .
615             .      .
616             .      .
617             .      .
618             .      .
619             .      .
620             .      .
621             .      .
622             .      .
623             .      .
624             .      .
625             .      .
626             .      .
627             .      .
628             .      .
629             .      .
630             .      .
631             .      .
632             .      .
633             .      .
634             .      .
635             .      .
636             .      .
637             .      .
638             .      .
639             .      .
640             .      .
641             .      .
642             .      .
643             .      .
644             .      .
645             .      .
646             .      .
647             .      .
648             .      .
649             .      .
650             .      .
651             .      .
652             .      .
653             .      .
654             .      .
655             .      .
656             .      .
657             .      .
658             .      .
659             .      .
660             .      .
661             .      .
662             .      .
663             .      .
664             .      .
665             .      .
666             .      .
667             .      .
668             .      .
669             .      .
670             .      .
671             .      .
672             .      .
673             .      .
674             .      .
675             .      .
676             .      .
677             .      .
678             .      .
679             .      .
680             .      .
681             .      .
682             .      .
683             .      .
684             .      .
685             .      .
686             .      .
687             .      .
688             .      .
689             .      .
690             .      .
691             .      .
692             .      .
693             .      .
694             .      .
695             .      .
696             .      .
697             .      .
698             .      .
699             .      .
700             .      .
701             .      .
702             .      .
703             .      .
704             .      .
705             .      .
706             .      .
707             .      .
708             .      .
709             .      .
710             .      .
711             .      .
712             .      .
713             .      .
714             .      .
715             .      .
716             .      .
717             .      .
718             .      .
719             .      .
720             .      .
721             .      .
722             .      .
723             .      .
724             .      .
725             .      .
726             .      .
727             .      .
728             .      .
729             .      .
730             .      .
731             .      .
732             .      .
733             .      .
734             .      .
735             .      .
736             .      .
737             .      .
738             .      .
739             .      .
740             .      .
741             .      .
742             .      .
743             .      .
744             .      .
745             .      .
746             .      .
747             .      .
748             .      .
749             .      .
750             .      .
751             .      .
752             .      .
753             .      .
754             .      .
755             .      .
756             .      .
757             .      .
758             .      .
759             .      .
760             .      .
761             .      .
762             .      .
763             .      .
764             .      .
765             .      .
766             .      .
767             .      .
768             .      .
769             .      .
770             .      .
771             .      .
772             .      .
773             .      .
774             .      .
775             .      .
776             .      .
777             .      .
778             .      .
779             .      .
780             .      .
781             .      .
782             .      .
783             .      .
784             .      .
785             .      .
786             .      .
787             .      .
788             .      .
789             .      .
790             .      .
791             .      .
792             .      .
793             .      .
794             .      .
795             .      .
796             .      .
797             .      .
798             .      .
799             .      .
800             .      .
801             .      .
802             .      .
803             .      .
804             .      .
805             .      .
806             .      .
807             .      .
808             .      .
809             .      .
810             .      .
811             .      .
812             .      .
813             .      .
814             .      .
815             .      .
816             .      .
817             .      .
818             .      .
819             .      .
820             .      .
821             .      .
822             .      .
823             .      .
824             .      .
825             .      .
826             .      .
827             .      .
828             .      .
829             .      .
830             .      .
831             .      .
832             .      .
833             .      .
834             .      .
835             .      .
836             .      .
837             .      .
838             .      .
839             .      .
840             .      .
841             .      .
842             .      .
843             .      .
844             .      .
845             .      .
846             .      .
847             .      .
848             .      .
849             .      .
850             .      .
851             .      .
852             .      .
853             .      .
854             .      .
855             .      .
856             .      .
857             .      .
858             .      .
859             .      .
860             .      .
861             .      .
862             .      .
863             .      .
864             .      .
865             .      .
866             .      .
867             .      .
868             .      .
869             .      .
870             .      .
871             .      .
872             .      .
873             .      .
874             .      .
875             .      .
876             .      .
877             .      .
878             .      .
879             .      .
880             .      .
881             .      .
882             .      .
883             .      .
884             .      .
885             .      .
886             .      .
887             .      .
888             .      .
889             .      .
890             .      .
891             .      .
892             .      .
893             .      .
894             .      .
895             .      .
896             .      .
897             .      .
898             .      .
899             .      .
900             .      .
901             .      .
902             .      .
903             .      .
904             .      .
905             .      .
906             .      .
907             .      .
908             .      .
909             .      .
910             .      .
911             .      .
912             .      .
913             .      .
914             .      .
915             .      .
916             .      .
917             .      .
918             .      .
919             .      .
920             .      .
921             .      .
922             .      .
923             .      .
924             .      .
925             .      .
926             .      .
927             .      .
928             .      .
929             .      .
930             .      .
931             .      .
932             .      .
933             .      .
934             .      .
935             .      .
936             .      .
937             .      .
938             .      .
939             .      .
940             .      .
941             .      .
942             .      .
943             .      .
944             .      .
945             .      .
946             .      .
947             .      .
948             .      .
949             .      .
950             .      .
951             .      .
952             .      .
953             .      .
954             .      .
955             .      .
956             .      .
957             .      .
958             .      .
959             .      .
960             .      .
961             .      .
962             .      .
963             .      .
964             .      .
965             .      .
966             .      .
967             .      .
968             .      .
969             .      .
970             .      .
971             .      .
972             .      .
973             .      .
974             .      .
975             .      .
976             .      .
977             .      .
978             .      .
979             .      .
980             .      .
981             .      .
982             .      .
983             .      .
984             .      .
985             .      .
986             .      .
987             .      .
988             .      .
989             .      .
990             .      .
991             .      .
992             .      .
993             .      .
994             .      .
995             .      .
996             .      .
997             .      .
998             .      .
999             .      .
1000            .      .

```

```

168      12      1      1      5      2      2      1      0
          0      dt = dt + ((float)(t1.tv_nsec - t0.
          tv_nsec)) / 1.0e9;
169      .      .      .      .      .      .      .      .
170      13      3      2      5      2      2      0      0
          0      if(print_matrix(stdout, c) == -1)
171      .      .      .      .      .      .      .      .
          .      goto _exit_main;
172      .      .      .      .      .      .      .      .
173      12      1      1      7      0      0      0      0
          0      fprintf(stderr, "time: %g\n", dt);
174      .      .      .      .      .      .      .      .
175      6      1      1      3      0      0      0      0
          0      free(line);
176      6      1      1      3      0      0      0      0
          0      destroy_matrix(a);
177      6      1      1      3      0      0      0      0
          0      destroy_matrix(b);
178      6      0      0      3      0      0      0      0
          0      destroy_matrix(c);
179      .      .      .      .      .      .      .      .
          .      }
180      .      .      .      .      .      .      .      .
181      3      1      1      0      0      0      0      0
          0      return 0;
182      .      .      .      .      .      .      .      .
183      .      .      .      .      .      .      .      .
          .      _exit_main:
184      .      .      .      .      .      .      .      .
          .      fprintf(stderr, " at line %u\n", (
          unsigned) lineno);
185      .      .      .      .      .      .      .      .
          .      free(line);
186      .      .      .      .      .      .      .      .
          .      destroy_matrix(a);
187      .      .      .      .      .      .      .      .
          .      destroy_matrix(b);
188      .      .      .      .      .      .      .      .
          .      destroy_matrix(c);
189      .      .      .      .      .      .      .      .
          .      exit(1);
190      6      1      1      2      0      0      0      0
          0      }
191      .      .      .      .      .      .      .      .
192      .      .      .      .      .      .      .      .
          .      matrix_t*
193      .      .      .      .      .      .      .      .
          .      matrix_multiply(matrix_t* m1, matrix_t*

```

```

194         m2, int bs)
195         11 2 2 0 0 0 6 0
196         0 {
197         . . . size_t n, en, i, j, k, kk, jj;
198         . . . double sum;
199         . . . matrix_t* mr;
200         . . .
201         3 1 1 2 0 0 1 0
202         0 n = m1->rows;
203         11 1 1 5 0 0 1 0
204         0 if(!(mr = create_matrix(n,n))) return
205         NULL;
206         . . .
207         9 2 2 3 0 0 1 0
208         0 en = bs*(n/bs);
209         . . .
210         72 2 2 26 0 0 9 0
211         0 for(i=0; i<n; i++)
212         576 1 1 208 0 0 72 0
213         0 for(j=0; j<n; j++)
214         704 1 1 320 0 0 128 15
215         15 mr->array[i*n+j] = 0.0;
216         . . .
217         . . .
218         . . . #if 1
219         . . . if (1) {
220         . . . size_t j;
221         2 0 0 0 0 0 1 0
222         0 size_t dim = 1024*1024*10;
223         9 1 1 3 0 0 1 0
224         0 int *v = malloc(dim*sizeof(int));
225         83,886,088 2 2 31,457,282 0 0 10,485,761 0
226         0 for (j = 0; j < dim; ++j)
227         62,914,560 0 0 20,971,520 0 0 10,485,760 1,310,720
228         1,310,720 v[j] = -1;
229         6 1 1 3 2 2 0 0
230         0 free(v);
231         . . .
232         . . . }
233         . . . #endif
234         . . .
235         . . .

```



```

220      17      2      2      6      0      0      2      0
      0      for(kk=0; kk<en; kk+=bs)
221      17      2      2      6      0      0      2      0
      0      for(jj=0; jj<en; jj+=bs)
222      72      1      1      26      0      0      9      0
      0      for(i=0; i<n; i++)
223      728      2      2      288      0      0      72      0
      0      for(j=jj; j<jj+bs; j++) {
224      704      1      1      384      17      17      64      0
      0      sum = mr->array[i*n+j];
225      5,824      3      3      2,304      0      0      576      0
      0      for(k=kk; k<kk+bs; k++)
226      12,288      2      2      6,656      33      33      512      0
      0      sum += m1->array[i*n+k] * m2->
      array[k*n+j];
227      704      1      1      384      0      0      64      0
      0      mr->array[i*n+j] = sum;
228      .      .      .      .      .      .      .      .
      .      }
229      1      0      0      1      0      0      0      0
      0      return mr;
230      6      1      1      2      0      0      0      0
      0 }
231      .      .      .      .      .      .      .      .
      .
232      .      .      .      .      .      .      .      .
      .      char*
233      .      .      .      .      .      .      .      .
      .      read_line(FILE *fp)
234      18      3      3      0      0      0      8      0
      0 {
235      .      .      .      .      .      .      .      .
      .      #define DEF_LINE_SZ 1024
236      .      .      .      .      .      .      .      .
      .
237      .      .      .      .      .      .      .      .
      .      int c;
238      6      0      0      0      0      0      4      0
      0      size_t len = 0, tam = DEF_LINE_SZ;
239      .      .      .      .      .      .      .      .
      .      char* str;
240      .      .      .      .      .      .      .      .
      .
241      14      1      1      6      1      0      2      1
      0      str = malloc(tam);
242      6      0      0      2      0      0      0      0
      0      if (!str) {
243      .      .      .      .      .      .      .      .
      .      perror("");
244      .      .      .      .      .      .      .      .
      .      return NULL;
245      .      .      .      .      .      .      .      .
      .      }
246      .      .      .      .      .      .      .      .

```

```

247      3,881      5      5      1,294      1      0      259      0
           0      while (EOF != (c=fgetc(fp)) && c != '\n')
           {
248      2,056      2      2      771      0      0      514      8
           8      str[len++]=c;
249      1,285      0      0      514      0      0      0      0
           0      if (len==tam-1) {
250      .      .      .      .      .      .      .      .
           .      str = realloc(str, tam *= 2);
251      .      .      .      .      .      .      .      .
           .      if (!str) {
252      .      .      .      .      .      .      .      .
           .      perror("");
253      .      .      .      .      .      .      .      .
           .      return NULL;
254      .      .      .      .      .      .      .      .
           .      }
255      .      .      .      .      .      .      .      .
           .      }
256      .      .      .      .      .      .      .      .
           .      }
257      .      .      .      .      .      .      .      .
           .      .
258      8      1      1      2      0      0      0      0
           0      if (c != EOF)
259      7      0      0      2      0      0      2      0
           0      str[len++]='\n';
260      .      .      .      .      .      .      .      .
           .      .
261      12      1      1      4      0      0      4      0
           0      str[len++]='\0';
262      2      0      0      2      0      0      0      0
           0      return str;
263      12      1      1      4      0      0      0      0
           0      }
264      .      .      .      .      .      .      .      .
           .      .
265      .      .      .      .      .      .      .      .
           .      .
266      .      .      .      .      .      .      .      .
           .      matrix_t*
267      .      .      .      .      .      .      .      .
           .      create_matrix(size_t rows, size_t cols)
268      30      1      1      0      0      0      15      0
           0      {
269      .      .      .      .      .      .      .      .
           .      matrix_t * m;
270      .      .      .      .      .      .      .      .
           .      .
271      30      2      2      9      1      0      3      0
           0      if (!(m = malloc(sizeof(matrix_t)))) {
272      .      .      .      .      .      .      .      .
           .      perror("");

```

```

273         .      .      .      .      .      .      .      .
274         .      .      .      .      .      .      .      .
275         .      .      .      .      .      .      .      .
276         9      1      1      .      6      0      0      3      0
277         .      .      .      .      .      .      .      .
278         9      1      1      .      6      0      0      3      0
279         51     2      2      .      21     0      0      3      0
280         .      .      .      .      .      .      .      .
281         .      .      .      .      .      .      .      .
282         .      .      .      .      .      .      .      .
283         .      .      .      .      .      .      .      .
284         3      1      1      .      3      0      0      0      0
285         18     1      1      .      6      0      0      0      0
286         .      .      .      .      .      .      .      .
287         .      .      .      .      .      .      .      .
288         .      .      .      .      .      .      .      .
289         27     1      1      .      0      0      0      12     0
290         9      0      0      .      3      0      0      0      0
291         24     1      1      .      12     0      0      0      0
292         24     1      1      .      9      0      0      0      0
293         18     0      0      .      6      0      0      0      0
294         .      .      .      .      .      .      .      .
295         .      .      .      .      .      .      .      .
296         .      .      .      .      .      .      .      .
297         10     2      2      .      0      0      0      5      0
298         .      .      .      .      .      .      .      .
299         .      .      .      .      .      .      .      .

```

300	3	1	1	.	size_t n;	2	0	0	1	0
				0	n = m->rows;					
301	13	1	1	.	6	1	1		0	0
				0	if (fprintf(fp, "%lu", (unsigned long)					
					m->rows) < 0) {					
302	perror("");
303	return -1;
304	}
305	72	2	2	.	26	0	0		9	0
				0	for(i=0; i<n; i++)					
306	576	2	2	.	208	0	0		72	0
				0	for (j=0; j<n; j++)					
307	1,408	3	3	.	640	0	0		0	0
				0	if (fprintf(fp, " %g", m->array[i*n+j					
) < 0) {						
308	perror("");
309	return -1;
310	}
311	10	1	1	.	4	0	0		0	0
				0	if (fprintf(fp, "\n") < 0) {					
312	perror("");
313	return -1;
314	}
315	1	1	1	.	0	0	0		0	0
				0	return 0;					
316	6	1	1	.	2	0	0		0	0
				0	}					
317										
318										
319	Ir	I1mr	ILmr	Dr		D1mr	DLmr	Dw		D1mw
		DLmw								
320										
321	146,837,450	120	119	52,445,273	65	59	20,974,500	1,310,777		
	1,310,775		events	annotated						

6.3. 4WSA:

```
1 30
2 == Cachegrind, a cache and branch-prediction profiler
```

```

3 ==651== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
   Nethercote et al.
4 ==651== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
5 ==651== Command: /tmp/02-mmult
6 ==651== Parent PID: 597
7 ==651==
8 --651-- Warning: Cannot auto-detect cache config, using
   defaults.
9 --651-- Run with -v to see.
10 ==651==
11 ==651== I    refs:      147,097,895
12 ==651== I1  misses:      2,354
13 ==651== LLi misses:      2,339
14 ==651== I1  miss rate:      0.00%
15 ==651== LLi miss rate:      0.00%
16 ==651==
17 ==651== D    refs:      73,517,815 (52,506,511 rd  +
   21,011,304 wr)
18 ==651== D1  misses:      1,314,783 (    3,494 rd  +
   1,311,289 wr)
19 ==651== LLd misses:      1,314,133 (    2,894 rd  +
   1,311,239 wr)
20 ==651== D1  miss rate:      1.8% (    0.0%  +
   6.2%  )
21 ==651== LLd miss rate:      1.8% (    0.0%  +
   6.2%  )
22 ==651==
23 ==651== LL refs:      1,317,137 (    5,848 rd  +
   1,311,289 wr)
24 ==651== LL misses:      1,316,472 (    5,233 rd  +
   1,311,239 wr)
25 ==651== LL miss rate:      0.6% (    0.0%  +
   6.2%  )
26 -----
27 I1 cache:      32768 B, 32 B, 4-way associative
28 D1 cache:      32768 B, 32 B, 4-way associative
29 LL cache:      524288 B, 32 B, 8-way associative
30 Command:      /tmp/02-mmult
31 Data file:      cachegrind.out.651
32 Events recorded: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:   Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:     0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:  /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation: off
39
40 -----
41 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw
   DLmw

```

```

42
43 147,097,895 2,354 2,339 52,506,511 3,494 2,894 21,011,304
    1,311,289 1,311,239 PROGRAM TOTALS
44 -----
45
46 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
         DLmw             file:function
47 -----
48 146,800,887   29   29 52,428,894       5     5 20,971,551
        1,310,720 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
            matrix_multiply
49 -----
50
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
         DLmw
54
55 -- line 18 -----
56      .      .      .      .      .      .      .
           .      matrix_t* create_matrix(size_t rows,
               size_t cols);
57      .      .      .      .      .      .      .
           .
58      .      .      .      .      .      .      .
           . void destroy_matrix(matrix_t* m);
59      .      .      .      .      .      .      .
           .
60      .      .      .      .      .      .      .
           . int print_matrix(FILE* fp, matrix_t* m)
              ;
61      .      .      .      .      .      .      .
           .
62      .      .      .      .      .      .      .
           . int
63      .      .      .      .      .      .      .
           . main(int argc, char** argv)
64      10      2      2      0      0      0      5      0
           0 {
65      .      .      .      .      .      .      .
           . /* matrixes */
66      .      .      .      .      .      .      .
           . matrix_t *a, *b, *c;
67      .      .      .      .      .      .      .
           . /* n (dimension) and block size */
68      .      .      .      .      .      .      .
           . size_t n, bs;
69      .      .      .      .      .      .      .
           . /* line buffer (init to null to

```

```

70         simplify freeing on error) */
71         1 0 0 0 0 0 0 1 0
72         0 char *line = NULL;
73         . . . . .
74         /* line parsing auxiliar pointers */
75         . . . . .
76         char *nptr, *endptr;
77         . . . . .
78         /* auxiliar variables */
79         . . . . .
80         long l;
81         . . . . .
82         double e;
83         2 1 1 0 0 0 1 0
84         0 size_t lineno = 1;
85         . . . . .
86         struct timespec t0;
87         . . . . .
88         struct timespec t1;
89         . . . . .
90         double dt;
91         . . . . .
92         size_t i;
93         . . . . .
94         25 3 3 9 0 0 1 0
95         0 for(;; !feof(stdin); lineno++) {
96         10 1 1 4 0 0 6 0
97         0 a=b=c=NULL;
98         . . . . .
99         18 3 3 8 1 0 2 0
100        0 line = read_line(stdin);
101        . . . . .
102        6 0 0 2 0 0 0 0
103        0 if (!line) goto _exit_main;
104        9 0 0 4 0 0 0 0
105        0 if (line[0] == 0) break;
106        . . . . .
107        . . . . .
108        /* parse dimension */
109        2 1 1 1 0 0 1 1
110        0 nptr = line;
111        10 1 1 3 1 0 1 0
112        0 l = strtol(nptr, &endptr, 10);
113        8 1 1 3 0 0 0 0
114        0 if (errno) {
115        . . . . .
116        perror("");
117        . . . . .
118        goto _exit_main;

```

```

96         . . . . } . . . . .
97         4 2 2 . 2 0 0 0 0
98         . . . . if (nptr == endptr) {
99         . . . .     fprintf(stderr, "missing
dimension");
100        . . . .     goto _exit_main;
101        . . . . }
102        3 2 2 . 1 0 0 0 0
103        . . . . if (l < 1) {
104        . . . .     fprintf(stderr, "invalid
dimension");
105        . . . .     goto _exit_main;
106        . . . . }
107        2 1 1 . 1 0 0 1 0
108        . . . . 0 n = (size_t) l;
109        . . . . #if 0
110        . . . .     /* parse block size */
111        . . . .     nptr = endptr;
112        . . . .     l = strtol(nptr, &endptr, 10);
113        . . . .     if (errno) {
114        . . . .         perror("");
115        . . . .         goto _exit_main;
116        . . . .     }
117        -- line 75 -----
118        -- line 83 -----
119        . . . . }
120        . . . . bs = (size_t) l;
121        . . . . if (n % bs) {
122        . . . .     fprintf(stderr, "block size doesn
't match");
123        . . . .     goto _exit_main;

```



```

122     . . . } . . .
123     . . . #else . . .
124     2 0 0 1 0 0 1 0
125         0 bs = n;
126     . . . #endif . . .
127     . . .
128     /* load matrix a */
129     11 1 1 5 1 0 1 0
130         0 if (!(a = create_matrix(n, n)))
131         goto _exit_main;
132     . . .
133     20 3 3 7 0 0 2 0
134         0 for (i=0; i < n*n; i++) {
135     2 0 0 1 0 0 1 0
136         nptr = endptr;
137     9 1 1 3 0 0 1 0
138         e = strtod(nptr, &endptr);
139     8 1 1 3 0 0 0 0
140         if (errno) {
141         . . .
142         perror("");
143         . . .
144         goto _exit_main;
145     . . .
146     }
147     4 2 2 2 0 0 0 0
148         if (nptr == endptr) {
149         . . .
150         fprintf(stderr, "missing A
matrix element");
151         . . .
152         goto _exit_main;
153     . . .
154     }
155     7 1 1 4 0 0 1 0
156         a->array[i] = e;
157     . . .
158     }
159     . . .
160     . . .
161     /* load matrix b */
162     11 1 1 5 0 0 1 0
163         0 if (!(b = create_matrix(n, n)))
164     . . .
165         goto _exit_main;

```

```

148      .      .      .      .      .      .      .
149      20      2      2      .      7      0      0      2      0
150      .      .      .      .      for (i=0; i < n*n; i++) {
151      2      1      1      .      1      0      0      1      0
152      .      .      .      .      nptr = endptr;
153      9      1      1      .      3      0      0      1      0
154      .      .      .      .      e = strtod(nptr, &endptr);
155      8      1      1      .      3      0      0      0      0
156      .      .      .      .      if (errno) {
157      .      .      .      .      perror("");
158      .      .      .      .      goto _exit_main;
159      .      .      .      .      }
160      4      1      1      .      2      0      0      0      0
161      .      .      .      .      if (nptr == endptr) {
162      .      .      .      .      fprintf(stderr, "missing B
163      matrix element");
164      .      .      .      .      goto _exit_main;
165      .      .      .      .      }
166      7      1      1      .      4      0      0      1      0
167      .      .      .      .      b->array[i] = e;
168      .      .      .      .      }
169      .      .      .      .
170      8      1      1      .      2      0      0      0      0
171      .      .      .      .      clock_gettime(CLOCK_REALTIME, &t0);
172      .      .      .      .
173      .      .      .      .      /* multiply matrixes */
174      13      2      2      .      6      1      1      1      0
175      .      .      .      .      if (!(c = matrix_multiply(a, b, bs))
176      )
177      .      .      .      .      goto _exit_main;
178      .      .      .      .
179      .      .      .      .
180      8      1      1      .      2      0      0      0      0
181      .      .      .      .      clock_gettime(CLOCK_REALTIME, &t1);
182      .      .      .      .
183      .      .      .      .
184      7      1      1      .      2      1      1      1      1
185      .      .      .      .      dt = (float) (t1.tv_sec - t0.tv_sec
186      );
187      12      1      1      .      5      2      2      1      0
188      .      .      .      .      dt = dt + ((float)(t1.tv_nsec - t0.
189      tv_nsec)) / 1.0e9;

```

```

173      .      .      .      .      .      .      .      .
174      13      3      2      .      5      2      2      .      0      0
175      .      .      .      .      if(print_matrix(stdout, c) == -1)
176      .      .      .      .      goto _exit_main;
177      12      1      1      .      7      0      0      .      0      0
178      .      .      .      .      fprintf(stderr, "time: %g\n", dt);
179      6      1      1      .      3      0      0      .      0      0
180      .      .      .      .      free(line);
181      6      1      1      .      3      0      0      .      0      0
182      .      .      .      .      destroy_matrix(a);
183      6      1      1      .      3      0      0      .      0      0
184      .      .      .      .      destroy_matrix(b);
185      6      0      0      .      3      0      0      .      0      0
186      .      .      .      .      destroy_matrix(c);
187      .      .      .      .      }
188      .      .      .      .      .      .      .      .      .      .
189      3      1      1      .      0      0      0      .      0      0
190      .      .      .      .      return 0;
191      .      .      .      .      .      .      .      .      .      .
192      .      .      .      .      _exit_main:
193      .      .      .      .      fprintf(stderr, " at line %u\n", (
194      .      .      .      .      unsigned) lineno);
195      .      .      .      .      free(line);
196      .      .      .      .      destroy_matrix(a);
197      .      .      .      .      destroy_matrix(b);
198      .      .      .      .      destroy_matrix(c);
199      .      .      .      .      exit(1);
200      6      1      1      .      2      0      0      .      0      0
201      .      .      .      .      0 }
202      .      .      .      .      .      .      .      .      .      .
203      .      .      .      .      matrix_t*
204      .      .      .      .      matrix_multiply(matrix_t* m1, matrix_t*
205      .      .      .      .      m2, int bs)
206      11      2      2      .      0      0      0      .      6      0
207      .      .      .      .      0 {

```

```

199         .      .      .      .      .      .      .      .
200         .      .      .      .      .      .      .      .
201         .      .      .      .      .      .      .      .
202         .      .      .      .      .      .      .      .
203         3      1      1      .      2      0      0      1      0
204         .      .      .      .      .      .      .      .
205         11     1      1      .      5      0      0      1      0
206         .      .      .      .      .      .      .      .
207         9      2      2      .      3      0      0      1      0
208         .      .      .      .      .      .      .      .
209         16     2      2      .      5      0      0      2      0
210         16     1      1      .      5      0      0      2      0
211         11     1      1      .      5      0      0      2      0
212         .      .      .      .      .      .      .      .
213         .      .      .      .      .      .      .      .
214         .      .      .      .      .      .      .      .
215         .      .      .      .      .      .      .      .
216         2      0      0      .      0      0      0      1      0
217         9      1      1      .      3      0      0      1      0
218         83,886,088 2      2 31,457,282 0      0 10,485,761 0
219         62,914,560 0      0 20,971,520 0      0 10,485,760 1,310,720
220         1,310,720 6      1      1      3      2      2      0      0
221         .      .      .      .      .      .      .      .
222         .      .      .      .      .      .      .      .
223         .      .      .      .      .      .      .      .
224         17     2      2      .      6      0      0      2      0

```

```

size_t n, en, i, j, k, kk, jj;

double sum;

matrix_t* mr;

n = m1->rows;

if(!(mr = create_matrix(n,n))) return
    NULL;

en = bs*(n/bs);

for(i=0; i<n; i++)
    for(j=0; j<n; j++)
        mr->array[i*n+j] = 0.0;

#ifdef 1
    if (1) {
        size_t j;
        size_t dim = 1024*1024*10;
        int *v = malloc(dim*sizeof(int));
        for (j = 0; j < dim; ++j)
            v[j] = -1;
        free(v);
    }
#endif
    for(kk=0; kk<en; kk+=bs)

```

```

225      17      2      2          6      0      0          2          0
                                for(jj=0; jj<en; jj+=bs)
226      16      1      1          5      0      0          2          0
                                for(i=0; i<n; i++)
227      21      2      2          8      0      0          2          0
                                for(j=jj; j<jj+bs; j++) {
228      11      1      1          6      1      1          1          0
                                sum = mr->array[i*n+j];
229      21      3      3          8      0      0          2          0
                                for(k=kk; k<kk+bs; k++)
230      24      2      2          13     2      2          1          0
                                sum += m1->array[i*n+k] * m2
                                ->array[k*n+j];
231      11      1      1          6      0      0          1          0
                                mr->array[i*n+j] = sum;
232      .      .      .          .      .      .          .          .
                                }
233      1      0      0          1      0      0          0          0
                                return mr;
234      6      1      1          2      0      0          0          0
                                0 }
235      .      .      .          .      .      .          .          .
236      .      .      .          .      .      .          .          .
                                char*
237      .      .      .          .      .      .          .          .
                                read_line(FILE *fp)
238      18      2      2          0      0      0          8          0
                                0 {
239      .      .      .          .      .      .          .          .
                                #define DEF_LINE_SZ 1024
240      .      .      .          .      .      .          .          .
241      .      .      .          .      .      .          .          .
                                int c;
242      6      0      0          0      0      0          4          0
                                size_t len = 0, tam = DEF_LINE_SZ;
243      .      .      .          .      .      .          .          .
                                char* str;
244      .      .      .          .      .      .          .          .
245      14      1      1          6      1      0          2          1
                                0 str = malloc(tam);
246      6      0      0          2      0      0          0          0
                                0 if (!str) {
247      .      .      .          .      .      .          .          .
                                perror("");
248      .      .      .          .      .      .          .          .
                                return NULL;
249      .      .      .          .      .      .          .          .
                                }
250      .      .      .          .      .      .          .          .

```

```

251      101      5      5      34      1      0      7      0
           0      while (EOF != (c=fgetc(fp)) && c != '\n
           ') {
252      40      2      2      15      0      0      10      0
           0      str[len++]=c;
253      25      0      0      10      0      0      0      0
           0      if (len==tam-1) {
254      .      .      .      .      .      .      .      .
           .      str = realloc(str, tam *= 2);
255      .      .      .      .      .      .      .      .
           .      if (!str) {
256      .      .      .      .      .      .      .      .
           .      perror("");
257      .      .      .      .      .      .      .      .
           .      return NULL;
258      .      .      .      .      .      .      .      .
           .      }
259      .      .      .      .      .      .      .      .
           .      }
260      .      .      .      .      .      .      .      .
           .      }
261      .      .      .      .      .      .      .      .
           .
262      8      1      1      2      0      0      0      0
           0      if (c != EOF)
263      7      0      0      2      0      0      2      0
           0      str[len++]='\n';
264      .      .      .      .      .      .      .      .
           .
265      12      1      1      4      0      0      4      0
           0      str[len++]='\0';
266      2      0      0      2      0      0      0      0
           0      return str;
267      12      1      1      4      0      0      0      0
           0 }
268      .      .      .      .      .      .      .      .
           .
269      .      .      .      .      .      .      .      .
           .
270      .      .      .      .      .      .      .      .
           .      matrix_t*
271      .      .      .      .      .      .      .      .
           .      create_matrix(size_t rows, size_t cols)
272      30      1      1      0      0      0      15      0
           0 {
273      .      .      .      .      .      .      .      .
           .      matrix_t * m;
274      .      .      .      .      .      .      .      .
           .
275      30      2      2      9      1      0      3      0
           0      if (!(m = malloc(sizeof(matrix_t)))) {
276      .      .      .      .      .      .      .      .
           .      perror("");

```

277	return NULL;	.	.	.
278	}	.	.	.
279
280	9	1	1	0	6 0 0	3	0	0
281	9	1	1	0	m->rows = rows;	6 0 0	3	0
282	51	2	2	0	m->cols = cols;	21 0 0	3	0
283	if (!(m->array = malloc(sizeof(double)	.	.	.
284	*rows*cols))) {	.	.	.
285	free(m);	.	.	.
286	perror("");	.	.	.
287	return NULL;	.	.	.
288	3	1	1	0	}	.	.	.
289	18	1	1	0	return m;	3 0 0	0	0
290	6 0 0	0	0	0
291	0 }	.	.	.
292
293	27	1	1	0	void	.	.	.
294	9	0	0	0	destroy_matrix(matrix_t* m)	.	.	.
295	24	1	1	0	0 {	0 0 0	12	0
296	24	1	1	0	3 0 0	0	0	0
297	18	0	0	0	if (!m) return;	12 0 0	0	0
298	free(m->array);	9 0 0	0	0
299	free(m);	6 0 0	0	0
300	0 }	.	.	.
301	10	2	2	0	int	.	.	.
302	print_matrix(FILE* fp, matrix_t* m)	.	.	.
	0 {	0 0 0	5	0
	size_t i, j;	.	.	.

```

303     .   .   .   .   .   .   .   .
304     3   1   1       size_t n;           2   0   0           1   0
305     13  1   1       0   n = m->rows;     6   1   1           0   0
306     0   if (fprintf(fp, "%lu", (unsigned long)
307     m->rows) < 0) {
308     .   .   .   .   .   .   .   .
309     .   .   .   .   .   .   .   .
310     .   .   .   .   .   .   .   .
311     16  2   2       5   0   0           2   0   0
312     0   for(i=0; i<n; i++)
313     16  2   2       5   0   0           2   0   0
314     0   for (j=0; j<n; j++)
315     22  3   3       10  0   0           0   0   0
316     0   if (fprintf(fp, " %g", m->array[i*
317     n+j]) < 0) {
318     .   .   .   .   .   .   .   .
319     .   .   .   .   .   .   .   .
320     .   .   .   .   .   .   .   .
321     .   .   .   .   .   .   .   .
322     .   .   .   .   .   .   .   .
323     10  1   1       4   0   0           0   0   0
324     0   if (fprintf(fp, "\n") < 0) {
325     .   .   .   .   .   .   .   .
326     .   .   .   .   .   .   .   .
327     .   .   .   .   .   .   .   .
328     .   .   .   .   .   .   .   .
329     .   .   .   .   .   .   .   .
330     1   1   1       0   0   0           0   0   0
331     0   return 0;
332     6   1   1       2   0   0           0   0   0
333     0   }

```

```

323 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
324 DLmw
325 146,801,827 118 117 52,429,222 18 12 20,971,672 1,310,723
1,310,721 events annotated

```

```

1 2 81 112 14 18
2 ind, a cache and branch-prediction profiler
3 ==654== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
  Nethercote et al.
4 ==654== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
  copyright info

```



```

5 ==654== Command: /tmp/02-mmult
6 ==654== Parent PID: 597
7 ==654==
8 --654-- Warning: Cannot auto-detect cache config, using
    defaults.
9 --654--          Run with -v to see.
10 ==654==
11 ==654== I   refs:          147,109,372
12 ==654== I1  misses:          2,401
13 ==654== L1i misses:          2,383
14 ==654== I1  miss rate:          0.00%
15 ==654== L1i miss rate:          0.00%
16 ==654==
17 ==654== D   refs:          73,522,043 (52,509,336 rd  +
    21,012,707 wr)
18 ==654== D1  misses:          1,314,795 (    3,498 rd  +
    1,311,297 wr)
19 ==654== L1d misses:          1,314,145 (    2,898 rd  +
    1,311,247 wr)
20 ==654== D1  miss rate:          1.8% (    0.0%  +
    6.2%  )
21 ==654== L1d miss rate:          1.8% (    0.0%  +
    6.2%  )
22 ==654==
23 ==654== LL refs:          1,317,196 (    5,899 rd  +
    1,311,297 wr)
24 ==654== LL misses:          1,316,528 (    5,281 rd  +
    1,311,247 wr)
25 ==654== LL miss rate:          0.6% (    0.0%  +
    6.2%  )
26 -----
27 I1 cache:          32768 B, 32 B, 4-way associative
28 D1 cache:          32768 B, 32 B, 4-way associative
29 LL cache:          524288 B, 32 B, 8-way associative
30 Command:          /tmp/02-mmult
31 Data file:          cachegrind.out.654
32 Events recorded:   Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:      Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order:  Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:        0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:    /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation:   off
39
40 -----
41 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw
    DLmw
42 -----
43 147,109,372 2,401 2,383 52,509,336 3,498 2,898 21,012,707
    1,311,297 1,311,247 PROGRAM TOTALS
44

```

```

45 -----
46 Ir          I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
         DLmw          file:function
47 -----
48 146,801,346  29  29 52,429,109      7    7 20,971,590
         1,310,720 1,310,720 /root/CARPETA/tp2-2020-2q-src/main.c:
         matrix_multiply
49 -----
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
         DLmw
54
55 -- line 18 -----
56 . . . . .
57 . . . . . matrix_t* create_matrix(size_t rows,
58 . . . . . size_t cols);
59 . . . . .
60 . . . . . void destroy_matrix(matrix_t* m);
61 . . . . .
62 . . . . . int print_matrix(FILE* fp, matrix_t* m)
63 . . . . . ;
64 . . . . .
65 . . . . . int
66 . . . . . main(int argc, char** argv)
67 10  2  2  0  0  0  5  0
68 0 {
69 . . . . . /* matrixes */
70 . . . . . matrix_t *a, *b, *c;
71 . . . . . /* n (dimension) and block size */
72 . . . . . size_t n, bs;
73 . . . . . /* line buffer (init to null to
74 . . . . . simplify freeing on error) */
75 1  0  0  0  0  0  1  0
76 0 char *line = NULL;
77 . . . . . /* line parsing auxiliar pointers */

```

```

72      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .
76      2      1      1      .      .      .      .      .
77      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .
82      25     4      4      .      .      .      .      .
83      10     1      1      .      .      .      .      .
84      .      .      .      .      .      .      .      .
85      18     3      3      .      .      .      .      .
86      .      .      .      .      .      .      .      .
87      6      0      0      .      .      .      .      .
88      9      0      0      .      .      .      .      .
89      .      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .      .
91      2      1      1      .      .      .      .      .
92      10     1      1      .      .      .      .      .
93      8      1      1      .      .      .      .      .
94      .      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .      .
97      4      2      2      .      .      .      .      .
98      .      .      .      .      .      .      .      .

      char *nptr, *endptr;

      /* auxiliar variables */

      long l;

      double e;

      size_t lineno = 1;

      struct timespec t0;

      struct timespec t1;

      double dt;

      size_t i;

      for(; !feof(stdin); lineno++) {

          a=b=c=NULL;

          line = read_line(stdin);

          if (!line) goto _exit_main;

          if (line[0] == 0) break;

          /* parse dimension */

          nptr = line;

          l = strtol(nptr, &endptr, 10);

          if (errno) {

              perror("");

              goto _exit_main;

          }

          if (nptr == endptr) {

              fprintf(stderr, "missing

```

```

99         dimension");
100         goto _exit_main;
101     }
102     if (l < 1) {
103         fprintf(stderr, "invalid
104         dimension");
105         goto _exit_main;
106     }
107     n = (size_t) l;
108     #if 0
109     /* parse block size */
110     nptr = endptr;
111     l = strtoul(nptr, &endptr, 10);
112     if (errno) {
113         perror("");
114         goto _exit_main;
115     }
116     }
117     bs = (size_t) l;
118     if (n % bs) {
119         fprintf(stderr, "block size doesn
120         't match");
121         goto _exit_main;
122     }
123     #else
124     bs = n;
125 
```

126	#endif
127
128	11	1	1	.	/* load matrix a */	5	1	0	1	0
129	if (!(a = create_matrix(n, n)))
130	goto _exit_main;
131	50	3	3	.		19	0	0	5	0
132	8	0	0	0	for (i=0; i < n*n; i++) {	4	0	0	4	0
133	36	1	1	0	nptr = endptr;	12	0	0	4	0
134	32	1	1	0	e = strtod(nptr, &endptr);	12	0	0	0	0
135	if (errno) {
136	perror("");
137	goto _exit_main;
138	16	2	2	0	}	8	0	0	0	0
139	if (nptr == endptr) {
140	fprintf(stderr, "missing A
141	matrix element");
142	28	1	1	0	goto _exit_main;
143	}
144	a->array[i] = e;
145	}
146	11	1	1	.	/* load matrix b */	5	0	0	1	0
147	if (!(b = create_matrix(n, n)))
148	goto _exit_main;
149	50	2	2	0		19	0	0	5	0
150	8	1	1	0	for (i=0; i < n*n; i++) {	4	0	0	4	0
151	36	1	1	0	nptr = endptr;	12	0	0	4	0
					e = strtod(nptr, &endptr);					

```

152      32      1      1      12      0      0      0      0
153      .      .      .      if (errno) {
154      .      .      .      perror("");
155      .      .      .      goto _exit_main;
156      16      1      1      8      0      0      0      0
157      .      .      .      if (nptr == endptr) {
158      .      .      .      fprintf(stderr, "missing B
matrix element");
159      .      .      .      goto _exit_main;
160      28      1      1      16      0      0      4      1
161      .      .      .      b->array[i] = e;
162      .      .      .      }
163      8      1      1      2      0      0      0      0
164      .      .      .      clock_gettime(CLOCK_REALTIME, &t0);
165      .      .      .      .
166      13      2      2      6      1      1      1      0
167      .      .      .      /* multiply matrixes */
168      .      .      .      if (!(c = matrix_multiply(a, b, bs))
)
169      8      1      1      2      0      0      0      0
170      .      .      .      clock_gettime(CLOCK_REALTIME, &t1);
171      7      1      1      2      1      1      1      1
172      12      1      1      5      2      2      1      0
173      .      .      .      dt = (float) (t1.tv_sec - t0.tv_sec
);
174      13      3      2      5      2      2      0      0
175      .      .      .      dt = dt + ((float)(t1.tv_nsec - t0.
tv_nsec)) / 1.0e9;
176      .      .      .      if(print_matrix(stdout, c) == -1)
goto _exit_main;

```

```

177         12      1      1          7      0      0          0      0
178             0      fprintf(stderr, "time: %g\n", dt);
179         .      .      .          .      .      .          .      .
179         6      1      1          3      0      0          0      0
180             0      free(line);
180         6      1      1          3      0      0          0      0
181             0      destroy_matrix(a);
181         6      1      1          3      0      0          0      0
182             0      destroy_matrix(b);
182         6      0      0          3      0      0          0      0
183             0      destroy_matrix(c);
183         .      .      .          .      .      .          .      .
184         .      .      .          .      .      .          .      .
185         3      1      1          0      0      0          0      0
186             0      return 0;
186         .      .      .          .      .      .          .      .
187         .      .      .          .      .      .          .      .
187         .      .      .      _exit_main:
188         .      .      .          .      .      .          .      .
188         .      .      .      fprintf(stderr, " at line %u\n", (
189         .      .      .      unsigned) lineno);
189         .      .      .          .      .      .          .      .
190         .      .      .      free(line);
190         .      .      .          .      .      .          .      .
191         .      .      .      destroy_matrix(a);
191         .      .      .          .      .      .          .      .
192         .      .      .      destroy_matrix(b);
192         .      .      .          .      .      .          .      .
193         .      .      .      destroy_matrix(c);
193         .      .      .          .      .      .          .      .
194         6      1      1          2      0      0          0      0
195             0      }
195         .      .      .          .      .      .          .      .
196         .      .      .          .      .      .          .      .
197         .      .      .      matrix_t*
197         .      .      .      matrix_multiply(matrix_t* m1, matrix_t*
198         11      2      2      m2, int bs)
198         .      .      .          0      0      0          6      0
199         .      .      .      0 {
199         .      .      .          .      .      .          .      .
200         .      .      .          size_t n, en, i, j, k, kk, jj;
200         .      .      .          .      .      .          .      .
201         .      .      .          double sum;
201         .      .      .          .      .      .          .      .
202         .      .      .          matrix_t* mr;
202         .      .      .          .      .      .          .      .

```

```

203      3      1      1      2      0      0      1      0
      0      n = m1->rows;
204      .      .      .      .      .      .      .      .
      .
205      11      1      1      5      0      0      1      0
      0      if(!(mr = create_matrix(n,n))) return
      NULL;
206      .      .      .      .      .      .      .      .
      .
207      9      2      2      3      0      0      1      0
      0      en = bs*(n/bs);
208      .      .      .      .      .      .      .      .
      .
209      24      2      2      8      0      0      3      0
      0      for(i=0; i<n; i++)
210      48      1      1      16      0      0      6      0
      0      for(j=0; j<n; j++)
211      44      1      1      20      0      0      8      0
      0      mr->array[i*n+j] = 0.0;
212      .      .      .      .      .      .      .      .
      .
213      .      .      .      .      .      .      .      .
      .      #if 1
214      .      .      .      .      .      .      .      .
      .      if (1) {
215      .      .      .      .      .      .      .      .
      .      size_t j;
216      2      0      0      0      0      0      1      0
      0      size_t dim = 1024*1024*10;
217      9      1      1      3      0      0      1      0
      0      int *v = malloc(dim*sizeof(int));
218      83,886,088      2      2 31,457,282      0      0 10,485,761      0
      0      for (j = 0; j < dim; ++j)
219      62,914,560      0      0 20,971,520      0      0 10,485,760 1,310,720
      1,310,720      v[j] = -1;
220      6      1      1      3      2      2      0      0
      0      free(v);
221      .      .      .      .      .      .      .      .
      .      }
222      .      .      .      .      .      .      .      .
      .      #endif
223      .      .      .      .      .      .      .      .
      .
224      17      2      2      6      0      0      2      0
      0      for(kk=0; kk<en; kk+=bs)
225      17      2      2      6      0      0      2      0
      0      for(jj=0; jj<en; jj+=bs)
226      24      1      1      8      0      0      3      0
      0      for(i=0; i<n; i++)
227      62      2      2      24      0      0      6      0
      0      for(j=jj; j<jj+bs; j++) {
228      44      1      1      24      2      2      4      0
      0      sum = mr->array[i*n+j];
229      124      3      3      48      0      0      12      0

```



```

230         0          for(k=kk; k<kk+bs; k++)
192      2      2      104      3      3      8      0
        0          sum += m1->array[i*n+k] * m2
        ->array[k*n+j];
231      44      1      1      24      0      0      4      0
        0          mr->array[i*n+j] = sum;
232      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
233      1      0      0      1      0      0      0      0
        0          return mr;
234      6      1      1      2      0      0      0      0
        0      }
235      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
238      18      3      3      0      0      0      8      0
        0      {
239      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
242      6      0      0      0      0      0      4      0
        0      size_t len = 0, tam = DEF_LINE_SZ;
243      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
245      14      1      1      6      1      0      2      1
        0      str = malloc(tam);
246      6      0      0      2      0      0      0      0
        0      if (!str) {
247      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
251      281      5      5      94      1      0      19      0
        0      while (EOF != (c=fgetc(fp)) && c != '\n
        ') {
252      136      2      2      51      0      0      34      0
        0      str[len++]=c;
253      85      0      0      34      0      0      0      0
        0      if (len==tam-1) {
254      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
        .      .      .      .      .      .      .      .
        str = realloc(str, tam *= 2);

```

```

256         .         .         .         .         .         .         .         .
257         .         .         .         .         .         .         .         .
258         .         .         .         .         .         .         .         .
259         .         .         .         .         .         .         .         .
260         .         .         .         .         .         .         .         .
261         .         .         .         .         .         .         .         .
262         8         1         1         .         2         0         0         .         0         0
263         .         .         .         .         .         .         .         .
264         .         .         .         .         .         .         .         .
265         12        1         1         .         4         0         0         .         4         0
266         .         .         .         .         .         .         .         .
267         2         0         0         .         2         0         0         .         0         0
268         .         .         .         .         .         .         .         .
269         .         .         .         .         .         .         .         .
270         .         .         .         .         .         .         .         .
271         .         .         .         .         .         .         .         .
272         30        1         1         .         0         0         0         .         15        0
273         .         .         .         .         .         .         .         .
274         .         .         .         .         .         .         .         .
275         30        2         2         .         9         1         0         .         3         0
276         .         .         .         .         .         .         .         .
277         .         .         .         .         .         .         .         .
278         .         .         .         .         .         .         .         .
279         .         .         .         .         .         .         .         .
280         9         1         1         .         6         0         0         .         3         0
281         9         1         1         .         6         0         0         .         3         0
282         51        2         2         .         21        0         0         .         3         0

```

```

0      if (!(m->array = malloc(sizeof(double)
283      *rows*cols))) {
284          free(m);
285          perror("");
286          return NULL;
287      }
288      3  1  1      3  0  0      0  0
0      return m;
289      18  1  1      6  0  0      0  0
0  }
290
291      void
292      destroy_matrix(matrix_t* m)
293      27  1  1      0  0  0      12  0
0  {
294      9  0  0      3  0  0      0  0
0      if (!m) return;
295      24  1  1      12  0  0      0  0
0      free(m->array);
296      24  1  1      9  0  0      0  0
0      free(m);
297      18  0  0      6  0  0      0  0
0  }
298
299      int
300      print_matrix(FILE* fp, matrix_t* m)
301      10  2  2      0  0  0      5  0
0  {
302          size_t i, j;
303          size_t n;
304      3  1  1      2  0  0      1  0
0      n = m->rows;
305      13  1  1      6  1  1      0  0
0      if (fprintf(fp, "%lu", (unsigned long)
m->rows) < 0) {
306          perror("");
307          return -1;
308

```

```

309         24    2    2        8    0    0        3    0
310         48    2    2        16   0    0        6    0
311         88    3    3        40   0    0        0    0
312         0      0      0      if (fprintf(fp, " %g", m->array[i*
313         n+j]) < 0) {
314         .      .      .          perror("");
315         .      .      .          return -1;
316         .      .      .          }
317         10    1    1        4    0    0        0    0
318         0      0      0      if (fprintf(fp, "\n") < 0) {
319         .      .      .          perror("");
320         .      .      .          return -1;
321         .      .      .          }
322         1    1    1        0    0    0        0    0
323         0      0      0      return 0;
324         6    1    1        2    0    0        0    0
325         0      0      0      }

```

```

323 Ir          I1mr ILmr Dr          Dimr DLmr Dw          Dimw
324      DLmw
325 146,802,968 120 119 52,429,703 20 14 20,971,776 1,310,724
      1,310,722 events annotated

```

```

1 3 10 29 43 26 70 95 29 60 76
2 nd branch-prediction profiler
3 ==657== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
4   Nethercote et al.
5 ==657== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
6   copyright info
7 ==657== Command: /tmp/02-mmult
8 ==657== Parent PID: 597
9 ==657==
10 --657-- Warning: Cannot auto-detect cache config, using
11 defaults.
12 --657-- Run with -v to see.
13 ==657==
14 ==657== I   refs:      147,126,801
15 ==657== I1  misses:      2,402
16 ==657== LLi misses:      2,384
17 ==657== I1  miss rate:      0.00%
18 ==657== LLi miss rate:      0.00%

```

```

16 ==657==
17 ==657== D   refs:          73,528,505  (52,513,724 rd   +
    21,014,781 wr)
18 ==657== D1  misses:       1,314,800  (    3,503 rd   +
    1,311,297 wr)
19 ==657== LLd misses:       1,314,153  (    2,901 rd   +
    1,311,252 wr)
20 ==657== D1  miss rate:      1.8% (    0.0%   +
    6.2%  )
21 ==657== LLd miss rate:      1.8% (    0.0%   +
    6.2%  )
22 ==657==
23 ==657== LL refs:          1,317,202  (    5,905 rd   +
    1,311,297 wr)
24 ==657== LL misses:       1,316,537  (    5,285 rd   +
    1,311,252 wr)
25 ==657== LL miss rate:      0.6% (    0.0%   +
    6.2%  )
26 -----
27 I1 cache:          32768 B, 32 B, 4-way associative
28 D1 cache:          32768 B, 32 B, 4-way associative
29 LL cache:          524288 B, 32 B, 8-way associative
30 Command:           /tmp/02-mmult
31 Data file:          cachegrind.out.657
32 Events recorded:    Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:       Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order:   Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:         0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:     /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation:    off
39
40 -----
41 Ir           I1mr ILmr Dr           D1mr DLmr Dw           D1mw
    DLmw
42 -----
43 147,126,801 2,402 2,384 52,513,724 3,503 2,901 21,014,781
    1,311,297 1,311,252 PROGRAM TOTALS
44 -----
45 -----
46 Ir           I1mr ILmr Dr           D1mr DLmr Dw           D1mw
    DLmw           file:function
47 -----
48 146,802,337 29 29 52,429,584 11 11 20,971,667
    1,310,721 1,310,721 /root/CARPETA/tp2-2020-2q-src/main.c:
    matrix_multiply
49 -----
50 -----

```

```

51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr I1Lmr Dr          D1mr DLmr Dw          D1mw
54          DLmw
55 -- line 18 -----
56          .          .          .          .          .          .
57          .          .          .          .          .          .
58          .          .          .          .          .          .
59          .          .          .          .          .          .
60          .          .          .          .          .          .
61          .          .          .          .          .          .
62          .          .          .          .          .          .
63          .          .          .          .          .          .
64          .          .          .          .          .          .
65          .          .          .          .          .          .
66          .          .          .          .          .          .
67          .          .          .          .          .          .
68          .          .          .          .          .          .
69          .          .          .          .          .          .
70          .          .          .          .          .          .
71          .          .          .          .          .          .
72          .          .          .          .          .          .
73          .          .          .          .          .          .
74          .          .          .          .          .          .
75          .          .          .          .          .          .
76          .          .          .          .          .          .
77          .          .          .          .          .          .

```

```

78      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .      .
82      25      4      4      .      .      .      .      .      0
83      10      1      1      0      for(; !feof(stdin); lineno++) {
84      .      .      .      .      .      .      .      .
85      18      3      3      .      .      .      .      .      0
86      .      .      .      .      .      .      .      .
87      6      0      0      .      .      .      .      .      0
88      9      0      0      0      if (!line) goto _exit_main;
89      .      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .      .
91      2      1      1      .      .      .      .      .      1
92      10      1      1      0      nptr = line;
93      8      1      1      0      l = strtol(nptr, &endptr, 10);
94      .      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .      .
97      4      2      2      .      .      .      .      .      0
98      .      .      .      .      .      .      .      .
99      .      .      .      .      .      .      .      .
100     .      .      .      .      .      .      .      .
101     3      2      2      .      .      .      .      .      0
102     .      .      .      .      .      .      .      .
103     .      .      .      .      .      .      .      .

```

```

104         . . . . } . . . . .
105         2 1 1 1 0 0 1 0
106         . . . . n = (size_t) 1;
107         . . . . #if 0
108         . . . . /* parse block size */
109         . . . . nptr = endptr;
110         . . . . l = strtol(nptr, &endptr, 10);
111         . . . . if (errno) {
112         . . . .     perror("");
113         . . . .     goto _exit_main;
114         . . . . }
115 -- line 75 -----
116 -- line 83 -----
117         . . . . }
118         . . . . bs = (size_t) 1;
119         . . . .
120         . . . . if (n % bs) {
121         . . . .     fprintf(stderr, "block size doesn
't match");
122         . . . .     goto _exit_main;
123         . . . . }
124         . . . . #else
125         2 0 0 1 0 0 1 0
126         . . . . bs = n;
127         . . . . #endif
128         . . . .
129         . . . . /* load matrix a */
130         11 1 1 5 1 0 1 0
131         . . . . if (!(a = create_matrix(n, n)))
        . . . .     goto _exit_main;
132         . . . .
133         100 3 3 39 0 0 10 0

```



```

132      18  0  0      0      for (i=0; i < n*n; i++) {
133      81  1  1      nptr = endptr;
134      72  1  1      e = strtod(nptr, &endptr);
135      .   .   .      if (errno) {
136      .   .   .          perror("");
137      .   .   .          goto _exit_main;
138      36  2  2      }
139      .   .   .      if (nptr == endptr) {
140      .   .   .          fprintf(stderr, "missing A
matrix element");
141      .   .   .          goto _exit_main;
142      63  1  1      }
143      .   .   .      a->array[i] = e;
144      .   .   .      }
145      .   .   .      /* load matrix b */
146      11  1  1      5  0  0      1      0
147      .   .   .      if (!(b = create_matrix(n, n)))
148      .   .   .          goto _exit_main;
149      100 2  2      39  0  0      10      0
150      18  1  1      0      for (i=0; i < n*n; i++) {
151      81  1  1      9  0  0      9      0
152      72  1  1      0      nptr = endptr;
153      .   .   .      e = strtod(nptr, &endptr);
154      .   .   .      if (errno) {
155      .   .   .          perror("");
156      36  1  1      .   .   .          goto _exit_main;
157      .   .   .      }
158      .   .   .      if (nptr == endptr) {
159      .   .   .          fprintf(stderr, "missing B

```

```

matrix element");
158      .      .      .      .      .      .      .      .
159      .      .      .      .      .      .      .      .
160      63      1      1      36      0      0      9      1
161      .      .      .      .      .      .      .      .
162      .      .      .      .      .      .      .      .
163      8      1      1      2      0      0      0      0
164      .      .      .      .      .      .      .      .
165      .      .      .      .      .      .      .      .
166      13      2      2      6      1      1      1      0
167      .      .      .      .      .      .      .      .
168      .      .      .      .      .      .      .      .
169      8      1      1      2      0      0      0      0
170      .      .      .      .      .      .      .      .
171      7      1      1      2      1      1      1      1
172      12      1      1      5      2      2      1      0
173      .      .      .      .      .      .      .      .
174      13      3      2      5      2      2      0      0
175      .      .      .      .      .      .      .      .
176      .      .      .      .      .      .      .      .
177      12      1      1      7      0      0      0      0
178      .      .      .      .      .      .      .      .
179      6      1      1      3      0      0      0      0
180      6      1      1      3      0      0      0      0
181      6      1      1      3      0      0      0      0
182      6      0      0      3      0      0      0      0

```

```

183     . . . . } . . . .
184     . . . . . . . . . .
185     3 1 1 0 0 0 0 0
186         0 return 0;
187     . . . . . . . . . .
188     . . . . _exit_main:
189     . . . . fprintf(stderr, " at line %u\n", (
190         unsigned) lineno);
191     . . . . free(line);
192     . . . . destroy_matrix(a);
193     . . . . destroy_matrix(b);
194     . . . . destroy_matrix(c);
195     . . . . exit(1);
196     6 1 1 2 0 0 0 0
197         0 }
198     . . . . . . . . . .
199     . . . . matrix_t*
200     . . . . matrix_multiply(matrix_t* m1, matrix_t*
201         m2, int bs)
202     11 2 2 0 0 0 6 0
203         0 {
204     . . . . size_t n, en, i, j, k, kk, jj;
205     . . . . double sum;
206     . . . . matrix_t* mr;
207     . . . .
208     3 1 1 2 0 0 1 0
209         0 n = m1->rows;
210     . . . .
211     11 1 1 5 0 0 1 0
212         0 if(!(mr = create_matrix(n,n))) return
213         NULL;
214     . . . .
215     9 2 2 3 0 0 1 0
216         0 en = bs*(n/bs);
217     . . . .
218     . . . .

```

```

209      32      2      2      .      11      0      0      4      0
      0      for(i=0; i<n; i++)
210      96      1      1      33      0      0      12      0
      0      for(j=0; j<n; j++)
211      99      1      1      45      0      0      18      1
      1      mr->array[i*n+j] = 0.0;
212      .      .      .      .      .      .      .      .
      .
213      .      .      .      .      .      .      .      .
      .      #if 1
214      .      .      .      .      .      .      .      .
      .      if (1) {
215      .      .      .      .      .      .      .      .
      .      size_t j;
216      2      0      0      0      0      0      1      0
      0      size_t dim = 1024*1024*10;
217      9      1      1      3      0      0      1      0
      0      int *v = malloc(dim*sizeof(int));
218 83,886,088      2      2 31,457,282      0      0 10,485,761      0
      0      for (j = 0; j < dim; ++j)
219 62,914,560      0      0 20,971,520      0      0 10,485,760 1,310,720
      1,310,720      v[j] = -1;
220      6      1      1      3      2      2      0      0
      0      free(v);
221      .      .      .      .      .      .      .      .
      .      }
222      .      .      .      .      .      .      .      .
      .      #endif
223      .      .      .      .      .      .      .      .
      .
224      17      2      2      6      0      0      2      0
      0      for(kk=0; kk<en; kk+=bs)
225      17      2      2      6      0      0      2      0
      0      for(jj=0; jj<en; jj+=bs)
226      32      1      1      11      0      0      4      0
      0      for(i=0; i<n; i++)
227      123      2      2      48      0      0      12      0
      0      for(j=jj; j<jj+bs; j++) {
228      99      1      1      54      3      3      9      0
      0      sum = mr->array[i*n+j];
229      369      3      3      144      0      0      36      0
      0      for(k=kk; k<kk+bs; k++)
230      648      2      2      351      6      6      27      0
      0      sum += m1->array[i*n+k] * m2
      ->array[k*n+j];
231      99      1      1      54      0      0      9      0
      0      mr->array[i*n+j] = sum;
232      .      .      .      .      .      .      .      .
      .      }
233      1      0      0      1      0      0      0      0
      0      return mr;
234      6      1      1      2      0      0      0      0
      0      }

```

```

235      .      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .      .
238      .      .      .      .      .      .      .      .
18      3      3      0      0      0      8      0
0      {
239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      .      .      .      .      .      .      .      .
242      .      .      .      .      .      .      .      .
6      0      0      0      0      0      4      0
0      size_t len = 0, tam = DEF_LINE_SZ;
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      14      1      1      6      1      0      2      1
0      str = malloc(tam);
246      6      0      0      2      0      0      0      0
0      if (!str) {
247      .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
251      581      5      5      194      1      0      39      0
0      while (EOF != (c=fgetc(fp)) && c != '\n
') {
252      296      2      2      111      0      0      74      1
1      str[len++]=c;
253      185      0      0      74      0      0      0      0
0      if (len==tam-1) {
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .

```

262	8	1	1	.	2	0	0	0	0
			0		if (c != EOF)				
263	7	0	0	.	2	0	0	2	0
			0		str[len++]='\n';				
264
			.						
265	12	1	1	.	4	0	0	4	0
			0		str[len++]='\0';				
266	2	0	0	.	2	0	0	0	0
			0		return str;				
267	12	1	1	.	4	0	0	0	0
			0	}					
268
			.						
269
			.						
270
			.		matrix_t*				
271
			.		create_matrix(size_t rows, size_t cols)				
272	30	1	1	.	0	0	0	15	0
			0	{					
273
			.		matrix_t * m;				
274
			.						
275	30	2	2	.	9	1	0	3	0
			0		if (!(m = malloc(sizeof(matrix_t)))) {				
276
			.		perror("");				
277
			.		return NULL;				
278
			.		}				
279
			.						
280	9	1	1	.	6	0	0	3	0
			0		m->rows = rows;				
281	9	1	1	.	6	0	0	3	0
			0		m->cols = cols;				
282	51	2	2	.	21	0	0	3	0
			0		if (!(m->array = malloc(sizeof(double)				
					*rows*cols))) {				
283
			.		free(m);				
284
			.		perror("");				
285
			.		return NULL;				
286
			.		}				
287
			.						

```

288      3      1      1      3      0      0      0      0
      0      return m;
289      18      1      1      6      0      0      0      0
      0      }
290      .      .      .      .      .      .      .      .
      .
291      .      .      .      .      .      .      .      .
      .      void
292      .      .      .      .      .      .      .      .
      .      destroy_matrix(matrix_t* m)
293      27      1      1      0      0      0      12      0
      0      {
294      9      0      0      3      0      0      0      0
      0      if (!m) return;
295      24      1      1      12      0      0      0      0
      0      free(m->array);
296      24      1      1      9      0      0      0      0
      0      free(m);
297      18      0      0      6      0      0      0      0
      0      }
298      .      .      .      .      .      .      .      .
      .
299      .      .      .      .      .      .      .      .
      .      int
300      .      .      .      .      .      .      .      .
      .      print_matrix(FILE* fp, matrix_t* m)
301      10      2      2      0      0      0      5      0
      0      {
302      .      .      .      .      .      .      .      .
      .      size_t i, j;
303      .      .      .      .      .      .      .      .
      .      size_t n;
304      3      1      1      2      0      0      1      0
      0      n = m->rows;
305      13      1      1      6      1      1      0      0
      0      if (fprintf(fp, "%lu", (unsigned long)
m->rows) < 0) {
306      .      .      .      .      .      .      .      .
      .      perror("");
307      .      .      .      .      .      .      .      .
      .      return -1;
308      .      .      .      .      .      .      .      .
      .      }
309      32      2      2      11      0      0      4      0
      0      for(i=0; i<n; i++)
310      96      2      2      33      0      0      12      0
      0      for (j=0; j<n; j++)
311      198      3      3      90      0      0      0      0
      0      if (fprintf(fp, " %g", m->array[i*n
+j]) < 0) {
312      .      .      .      .      .      .      .      .
      .      perror("");
313      .      .      .      .      .      .      .      .
      .      return -1;

```

```

314      .      .      .      .      .      .      .      .
315      10     1     1         4     0     0         0         0
316              0      if (fprintf(fp, "\n") < 0) {
317              .      .      .      .      .      .      .      .
318              .      .      .      .      .      .      .      .
319              .      .      .      .      .      .      .      .
320              .      .      .      .      .      .      .      .
321              .      .      .      .      .      .      .      .
322              .      .      .      .      .      .      .      .
323      Ir      I1mr ILmr Dr      Dimr DLmr Dw      Dimw
324              DLmw
325      146,805,085 120 119 52,430,618 24 18 20,971,960 1,310,727
          1,310,725 events annotated

```

```

1 4 97 46 101 53 76 89 135 88 113 87 126 89 90 88 142 102
2 er
3 ==660== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
4 Nethercote et al.
5 ==660== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
6 copyright info
7 ==660== Command: /tmp/02-mmult
8 ==660== Parent PID: 597
9 ==660==
10 --660-- Warning: Cannot auto-detect cache config, using
11 defaults.
12 --660-- Run with -v to see.
13 ==660==
14 ==660== I refs: 147,153,509
15 ==660== I1 misses: 2,400
16 ==660== LLi misses: 2,382
17 ==660== I1 miss rate: 0.00%
18 ==660== LLi miss rate: 0.00%
19 ==660==
20 ==660== D refs: 73,538,399 (52,520,440 rd +
21 21,017,959 wr)
22 ==660== D1 misses: 1,314,813 ( 3,509 rd +
23 1,311,304 wr)
24 ==660== LLd misses: 1,314,166 ( 2,907 rd +
25 1,311,259 wr)
26 ==660== D1 miss rate: 1.8% ( 0.0% +
27 6.2% )
28 ==660== LLd miss rate: 1.8% ( 0.0% +
29 6.2% )
30 ==660==

```



```

23 ==660== LL refs:          1,317,213  (    5,909 rd  +
      1,311,304 wr)
24 ==660== LL misses:      1,316,548  (    5,289 rd  +
      1,311,259 wr)
25 ==660== LL miss rate:          0.6% (    0.0%  +
      6.2%  )
26 -----
27 I1 cache:          32768 B, 32 B, 4-way associative
28 D1 cache:          32768 B, 32 B, 4-way associative
29 LL cache:          524288 B, 32 B, 8-way associative
30 Command:           /tmp/O2-mmult
31 Data file:          cachegrind.out.660
32 Events recorded:   Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
33 Events shown:      Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
34 Event sort order: Ir I1mr I1mr Dr D1mr DLmr Dw D1mw DLmw
35 Thresholds:        0.1 100 100 100 100 100 100 100 100
36 Include dirs:
37 User annotated:     /root/CARPETA/tp2-2020-2q-src/main.c
38 Auto-annotation:    off
39
40 -----
41 Ir          I1mr I1mr Dr          D1mr DLmr Dw          D1mw
      DLmw
42 -----
43 147,153,509 2,400 2,382 52,520,440 3,509 2,907 21,017,959
      1,311,304 1,311,259 PROGRAM TOTALS
44 -----
45 -----
46 Ir          I1mr I1mr Dr          D1mr DLmr Dw          D1mw
      DLmw          file:function
47 -----
48 146,804,064 29 29 52,430,421 16 16 20,971,794
      1,310,723 1,310,723 /root/CARPETA/tp2-2020-2q-src/main.c:
      matrix_multiply
49 -----
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr I1mr Dr          D1mr DLmr Dw          D1mw
      DLmw
54
55 -- line 18 -----
56 . . . . .
      . matrix_t* create_matrix(size_t rows,
      size_t cols);
57 . . . . .

```

```

58         .      .      .      void destroy_matrix(matrix_t* m);      .
59         .      .      .      .      .      .      .      .      .
60         .      .      .      .      .      .      .      .      .
        .      .      .      int print_matrix(FILE* fp, matrix_t* m)
        ;
61         .      .      .      .      .      .      .      .      .
62         .      .      .      .      .      .      .      .      .
63         .      .      .      int
64         .      .      .      main(int argc, char** argv)
10      2      2      0      0      0      5      0
        0      {
65         .      .      .      .      .      .      .      .      .
        .      .      .      /* matrixes */
66         .      .      .      .      .      .      .      .      .
        .      .      .      matrix_t *a, *b, *c;
67         .      .      .      .      .      .      .      .      .
        .      .      .      /* n (dimension) and block size */
68         .      .      .      .      .      .      .      .      .
        .      .      .      size_t n, bs;
69         .      .      .      .      .      .      .      .      .
        .      .      .      /* line buffer (init to null to
70      1      0      0      0      0      0      1      0
        0      simplify freeing on error) */
        .      .      .      char *line = NULL;
71         .      .      .      .      .      .      .      .      .
        .      .      .      /* line parsing auxiliar pointers */
72         .      .      .      .      .      .      .      .      .
        .      .      .      char *nptr, *endptr;
73         .      .      .      .      .      .      .      .      .
        .      .      .      /* auxiliar variables */
74         .      .      .      .      .      .      .      .      .
        .      .      .      long l;
75         .      .      .      .      .      .      .      .      .
        .      .      .      double e;
76      2      1      1      0      0      0      1      0
        0      size_t lineno = 1;
77         .      .      .      .      .      .      .      .      .
        .      .      .      struct timespec t0;
78         .      .      .      .      .      .      .      .      .
        .      .      .      struct timespec t1;
79         .      .      .      .      .      .      .      .      .
        .      .      .      double dt;
80         .      .      .      .      .      .      .      .      .
        .      .      .      size_t i;
81         .      .      .      .      .      .      .      .      .
        .      .      .      .
82      25      4      4      9      0      0      1      0
        0      for(; !feof(stdin); lineno++) {
83      10      1      1      4      0      0      6      0
        0      a=b=c=NULL;

```

```

84      .      .      .      .      .      .      .
85      18      3      3      .      8      1      0      .      2      0
86      .      .      .      .      line = read_line(stdin);
87      .      .      .      .      .      .      .      .      .
88      6      0      0      .      2      0      0      .      0      0
89      .      .      .      .      if (!line) goto _exit_main;
90      .      .      .      .      4      0      0      .      0      0
91      2      1      1      .      .      .      .      .      .      .
92      .      .      .      .      /* parse dimension */
93      10      1      1      .      1      0      0      .      1      1
94      .      .      .      .      nptr = line;
95      8      1      1      .      3      1      0      .      1      0
96      .      .      .      .      l = strtol(nptr, &endptr, 10);
97      .      .      .      .      3      0      0      .      0      0
98      .      .      .      .      if (errno) {
99      .      .      .      .      .      .      .      .      .      .
100     .      .      .      .      perror("");
101     .      .      .      .      .      .      .      .      .      .
102     .      .      .      .      goto _exit_main;
103     .      .      .      .      .      .      .      .      .      .
104     .      .      .      .      }
105     4      2      2      .      2      0      0      .      0      0
106     .      .      .      .      if (nptr == endptr) {
107     .      .      .      .      fprintf(stderr, "missing
108     dimension");
109     .      .      .      .      goto _exit_main;
110     .      .      .      .      .      .      .      .      .      .
111     .      .      .      .      }
112     3      2      2      .      1      0      0      .      0      0
113     .      .      .      .      if (l < 1) {
114     .      .      .      .      .      .      .      .      .      .
115     .      .      .      .      fprintf(stderr, "invalid
116     dimension");
117     .      .      .      .      goto _exit_main;
118     .      .      .      .      .      .      .      .      .      .
119     .      .      .      .      }
120     2      1      1      .      1      0      0      .      1      0
121     .      .      .      .      n = (size_t) l;
122     .      .      .      .      .      .      .      .      .      .
123     .      .      .      .      #if 0
124     .      .      .      .      .      .      .      .      .      .
125     .      .      .      .      /* parse block size */
126     .      .      .      .      .      .      .      .      .      .
127     .      .      .      .      nptr = endptr;
128     .      .      .      .      .      .      .      .      .      .
129     .      .      .      .      l = strtol(nptr, &endptr, 10);

```

```

110         . . . . . if (errno) {
111         . . . . .     . . . . .
112         . . . . .     perror("");
113         . . . . .     goto _exit_main;
114         . . . . . }
115 -- line 75 -----
116         . . . . . }
117         . . . . .
118         . . . . . bs = (size_t) 1;
119         . . . . .
120         . . . . . if (n % bs) {
121         . . . . .     fprintf(stderr, "block size doesn
't match");
122         . . . . .     goto _exit_main;
123         . . . . . }
124         . . . . . #else
125         . . . . .     2 0 0 1 0 0 1 0
126         . . . . .     0 bs = n;
127         . . . . . #endif
128         . . . . .
129         . . . . . /* load matrix a */
130         . . . . . 11 1 1 5 1 0 1 0
131         . . . . .     if (!(a = create_matrix(n, n)))
132         . . . . .     goto _exit_main;
133         . . . . .
134         . . . . . 170 3 3 67 0 0 17 0
135         . . . . .     for (i=0; i < n*n; i++) {
136         . . . . .         32 0 0 16 0 0 16 0
137         . . . . .         nptr = endptr;
138         . . . . . 144 1 1 48 0 0 16 0
139         . . . . .         e = strtod(nptr, &endptr);
140         . . . . . 128 1 1 48 0 0 0 0
141         . . . . .         if (errno) {
142         . . . . .             . . . . .
143         . . . . .             perror("");
144         . . . . .             goto _exit_main;
145         . . . . .         }
146         . . . . .
147         . . . . .
148         . . . . .
149         . . . . .
150         . . . . .
151         . . . . .
152         . . . . .
153         . . . . .
154         . . . . .
155         . . . . .
156         . . . . .
157         . . . . .
158         . . . . .
159         . . . . .
160         . . . . .
161         . . . . .
162         . . . . .
163         . . . . .
164         . . . . .
165         . . . . .
166         . . . . .
167         . . . . .
168         . . . . .
169         . . . . .
170         . . . . .
171         . . . . .
172         . . . . .
173         . . . . .
174         . . . . .
175         . . . . .
176         . . . . .
177         . . . . .
178         . . . . .
179         . . . . .
180         . . . . .
181         . . . . .
182         . . . . .
183         . . . . .
184         . . . . .
185         . . . . .
186         . . . . .
187         . . . . .
188         . . . . .
189         . . . . .
190         . . . . .
191         . . . . .
192         . . . . .
193         . . . . .
194         . . . . .
195         . . . . .
196         . . . . .
197         . . . . .
198         . . . . .
199         . . . . .
200         . . . . .
201         . . . . .
202         . . . . .
203         . . . . .
204         . . . . .
205         . . . . .
206         . . . . .
207         . . . . .
208         . . . . .
209         . . . . .
210         . . . . .
211         . . . . .
212         . . . . .
213         . . . . .
214         . . . . .
215         . . . . .
216         . . . . .
217         . . . . .
218         . . . . .
219         . . . . .
220         . . . . .
221         . . . . .
222         . . . . .
223         . . . . .
224         . . . . .
225         . . . . .
226         . . . . .
227         . . . . .
228         . . . . .
229         . . . . .
230         . . . . .
231         . . . . .
232         . . . . .
233         . . . . .
234         . . . . .
235         . . . . .
236         . . . . .
237         . . . . .
238         . . . . .
239         . . . . .
240         . . . . .
241         . . . . .
242         . . . . .
243         . . . . .
244         . . . . .
245         . . . . .
246         . . . . .
247         . . . . .
248         . . . . .
249         . . . . .
250         . . . . .
251         . . . . .
252         . . . . .
253         . . . . .
254         . . . . .
255         . . . . .
256         . . . . .
257         . . . . .
258         . . . . .
259         . . . . .
260         . . . . .
261         . . . . .
262         . . . . .
263         . . . . .
264         . . . . .
265         . . . . .
266         . . . . .
267         . . . . .
268         . . . . .
269         . . . . .
270         . . . . .
271         . . . . .
272         . . . . .
273         . . . . .
274         . . . . .
275         . . . . .
276         . . . . .
277         . . . . .
278         . . . . .
279         . . . . .
280         . . . . .
281         . . . . .
282         . . . . .
283         . . . . .
284         . . . . .
285         . . . . .
286         . . . . .
287         . . . . .
288         . . . . .
289         . . . . .
290         . . . . .
291         . . . . .
292         . . . . .
293         . . . . .
294         . . . . .
295         . . . . .
296         . . . . .
297         . . . . .
298         . . . . .
299         . . . . .
300         . . . . .
301         . . . . .
302         . . . . .
303         . . . . .
304         . . . . .
305         . . . . .
306         . . . . .
307         . . . . .
308         . . . . .
309         . . . . .
310         . . . . .
311         . . . . .
312         . . . . .
313         . . . . .
314         . . . . .
315         . . . . .
316         . . . . .
317         . . . . .
318         . . . . .
319         . . . . .
320         . . . . .
321         . . . . .
322         . . . . .
323         . . . . .
324         . . . . .
325         . . . . .
326         . . . . .
327         . . . . .
328         . . . . .
329         . . . . .
330         . . . . .
331         . . . . .
332         . . . . .
333         . . . . .
334         . . . . .
335         . . . . .
336         . . . . .
337         . . . . .
338         . . . . .
339         . . . . .
340         . . . . .
341         . . . . .
342         . . . . .
343         . . . . .
344         . . . . .
345         . . . . .
346         . . . . .
347         . . . . .
348         . . . . .
349         . . . . .
350         . . . . .
351         . . . . .
352         . . . . .
353         . . . . .
354         . . . . .
355         . . . . .
356         . . . . .
357         . . . . .
358         . . . . .
359         . . . . .
360         . . . . .
361         . . . . .
362         . . . . .
363         . . . . .
364         . . . . .
365         . . . . .
366         . . . . .
367         . . . . .
368         . . . . .
369         . . . . .
370         . . . . .
371         . . . . .
372         . . . . .
373         . . . . .
374         . . . . .
375         . . . . .
376         . . . . .
377         . . . . .
378         . . . . .
379         . . . . .
380         . . . . .
381         . . . . .
382         . . . . .
383         . . . . .
384         . . . . .
385         . . . . .
386         . . . . .
387         . . . . .
388         . . . . .
389         . . . . .
390         . . . . .
391         . . . . .
392         . . . . .
393         . . . . .
394         . . . . .
395         . . . . .
396         . . . . .
397         . . . . .
398         . . . . .
399         . . . . .
400         . . . . .
401         . . . . .
402         . . . . .
403         . . . . .
404         . . . . .
405         . . . . .
406         . . . . .
407         . . . . .
408         . . . . .
409         . . . . .
410         . . . . .
411         . . . . .
412         . . . . .
413         . . . . .
414         . . . . .
415         . . . . .
416         . . . . .
417         . . . . .
418         . . . . .
419         . . . . .
420         . . . . .
421         . . . . .
422         . . . . .
423         . . . . .
424         . . . . .
425         . . . . .
426         . . . . .
427         . . . . .
428         . . . . .
429         . . . . .
430         . . . . .
431         . . . . .
432         . . . . .
433         . . . . .
434         . . . . .
435         . . . . .
436         . . . . .
437         . . . . .
438         . . . . .
439         . . . . .
440         . . . . .
441         . . . . .
442         . . . . .
443         . . . . .
444         . . . . .
445         . . . . .
446         . . . . .
447         . . . . .
448         . . . . .
449         . . . . .
450         . . . . .
451         . . . . .
452         . . . . .
453         . . . . .
454         . . . . .
455         . . . . .
456         . . . . .
457         . . . . .
458         . . . . .
459         . . . . .
460         . . . . .
461         . . . . .
462         . . . . .
463         . . . . .
464         . . . . .
465         . . . . .
466         . . . . .
467         . . . . .
468         . . . . .
469         . . . . .
470         . . . . .
471         . . . . .
472         . . . . .
473         . . . . .
474         . . . . .
475         . . . . .
476         . . . . .
477         . . . . .
478         . . . . .
479         . . . . .
480         . . . . .
481         . . . . .
482         . . . . .
483         . . . . .
484         . . . . .
485         . . . . .
486         . . . . .
487         . . . . .
488         . . . . .
489         . . . . .
490         . . . . .
491         . . . . .
492         . . . . .
493         . . . . .
494         . . . . .
495         . . . . .
496         . . . . .
497         . . . . .
498         . . . . .
499         . . . . .
500         . . . . .
501         . . . . .
502         . . . . .
503         . . . . .
504         . . . . .
505         . . . . .
506         . . . . .
507         . . . . .
508         . . . . .
509         . . . . .
510         . . . . .
511         . . . . .
512         . . . . .
513         . . . . .
514         . . . . .
515         . . . . .
516         . . . . .
517         . . . . .
518         . . . . .
519         . . . . .
520         . . . . .
521         . . . . .
522         . . . . .
523         . . . . .
524         . . . . .
525         . . . . .
526         . . . . .
527         . . . . .
528         . . . . .
529         . . . . .
530         . . . . .
531         . . . . .
532         . . . . .
533         . . . . .
534         . . . . .
535         . . . . .
536         . . . . .
537         . . . . .
538         . . . . .
539         . . . . .
540         . . . . .
541         . . . . .
542         . . . . .
543         . . . . .
544         . . . . .
545         . . . . .
546         . . . . .
547         . . . . .
548         . . . . .
549         . . . . .
550         . . . . .
551         . . . . .
552         . . . . .
553         . . . . .
554         . . . . .
555         . . . . .
556         . . . . .
557         . . . . .
558         . . . . .
559         . . . . .
560         . . . . .
561         . . . . .
562         . . . . .
563         . . . . .
564         . . . . .
565         . . . . .
566         . . . . .
567         . . . . .
568         . . . . .
569         . . . . .
570         . . . . .
571         . . . . .
572         . . . . .
573         . . . . .
574         . . . . .
575         . . . . .
576         . . . . .
577         . . . . .
578         . . . . .
579         . . . . .
580         . . . . .
581         . . . . .
582         . . . . .
583         . . . . .
584         . . . . .
585         . . . . .
586         . . . . .
587         . . . . .
588         . . . . .
589         . . . . .
590         . . . . .
591         . . . . .
592         . . . . .
593         . . . . .
594         . . . . .
595         . . . . .
596         . . . . .
597         . . . . .
598         . . . . .
599         . . . . .
600         . . . . .
601         . . . . .
602         . . . . .
603         . . . . .
604         . . . . .
605         . . . . .
606         . . . . .
607         . . . . .
608         . . . . .
609         . . . . .
610         . . . . .
611         . . . . .
612         . . . . .
613         . . . . .
614         . . . . .
615         . . . . .
616         . . . . .
617         . . . . .
618         . . . . .
619         . . . . .
620         . . . . .
621         . . . . .
622         . . . . .
623         . . . . .
624         . . . . .
625         . . . . .
626         . . . . .
627         . . . . .
628         . . . . .
629         . . . . .
630         . . . . .
631         . . . . .
632         . . . . .
633         . . . . .
634         . . . . .
635         . . . . .
636         . . . . .
637         . . . . .
638         . . . . .
639         . . . . .
640         . . . . .
641         . . . . .
642         . . . . .
643         . . . . .
644         . . . . .
645         . . . . .
646         . . . . .
647         . . . . .
648         . . . . .
649         . . . . .
650         . . . . .
651         . . . . .
652         . . . . .
653         . . . . .
654         . . . . .
655         . . . . .
656         . . . . .
657         . . . . .
658         . . . . .
659         . . . . .
660         . . . . .
661         . . . . .
662         . . . . .
663         . . . . .
664         . . . . .
665         . . . . .
666         . . . . .
667         . . . . .
668         . . . . .
669         . . . . .
670         . . . . .
671         . . . . .
672         . . . . .
673         . . . . .
674         . . . . .
675         . . . . .
676         . . . . .
677         . . . . .
678         . . . . .
679         . . . . .
680         . . . . .
681         . . . . .
682         . . . . .
683         . . . . .
684         . . . . .
685         . . . . .
686         . . . . .
687         . . . . .
688         . . . . .
689         . . . . .
690         . . . . .
691         . . . . .
692         . . . . .
693         . . . . .
694         . . . . .
695         . . . . .
696         . . . . .
697         . . . . .
698         . . . . .
699         . . . . .
700         . . . . .
701         . . . . .
702         . . . . .
703         . . . . .
704         . . . . .
705         . . . . .
706         . . . . .
707         . . . . .
708         . . . . .
709         . . . . .
710         . . . . .
711         . . . . .
712         . . . . .
713         . . . . .
714         . . . . .
715         . . . . .
716         . . . . .
717         . . . . .
718         . . . . .
719         . . . . .
720         . . . . .
721         . . . . .
722         . . . . .
723         . . . . .
724         . . . . .
725         . . . . .
726         . . . . .
727         . . . . .
728         . . . . .
729         . . . . .
730         . . . . .
731         . . . . .
732         . . . . .
733         . . . . .
734         . . . . .
735         . . . . .
736         . . . . .
737         . . . . .
738         . . . . .
739         . . . . .
740         . . . . .
741         . . . . .
742         . . . . .
743         . . . . .
744         . . . . .
745         . . . . .
746         . . . . .
747         . . . . .
748         . . . . .
749         . . . . .
750         . . . . .
751         . . . . .
752         . . . . .
753         . . . . .
754         . . . . .
755         . . . . .
756         . . . . .
757         . . . . .
758         . . . . .
759         . . . . .
760         . . . . .
761         . . . . .
762         . . . . .
763         . . . . .
764         . . . . .
765         . . . . .
766         . . . . .
767         . . . . .
768         . . . . .
769         . . . . .
770         . . . . .
771         . . . . .
772         . . . . .
773         . . . . .
774         . . . . .
775         . . . . .
776         . . . . .
777         . . . . .
778         . . . . .
779         . . . . .
780         . . . . .
781         . . . . .
782         . . . . .
783         . . . . .
784         . . . . .
785         . . . . .
786         . . . . .
787         . . . . .
788         . . . . .
789         . . . . .
790         . . . . .
791         . . . . .
792         . . . . .
793         . . . . .
794         . . . . .
795         . . . . .
796         . . . . .
797         . . . . .
798         . . . . .
799         . . . . .
800         . . . . .
801         . . . . .
802         . . . . .
803         . . . . .
804         . . . . .
805         . . . . .
806         . . . . .
807         . . . . .
808         . . . . .
809         . . . . .
810         . . . . .
811         . . . . .
812         . . . . .
813         . . . . .
814         . . . . .
815         . . . . .
816         . . . . .
817         . . . . .
818         . . . . .
819         . . . . .
820         . . . . .
821         . . . . .
822         . . . . .
823         . . . . .
824         . . . . .
825         . . . . .
826         . . . . .
827         . . . . .
828         . . . . .
829         . . . . .
830         . . . . .
831         . . . . .
832         . . . . .
833         . . . . .
834         . . . . .
835         . . . . .
836         . . . . .
837         . . . . .
838         . . . . .
839         . . . . .
840         . . . . .
841         . . . . .
842         . . . . .
843         . . . . .
844         . . . . .
845         . . . . .
846         . . . . .
847         . . . . .
848         . . . . .
849         . . . . .
850         . . . . .
851         . . . . .
852         . . . . .
853         . . . . .
854         . . . . .
855         . . . . .
856         . . . . .
857         . . . . .
858         . . . . .
859         . . . . .
860         . . . . .
861         . . . . .
862         . . . . .
863         . . . . .
864         . . . . .
865         . . . . .
866         . . . . .
867         . . . . .
868         . . . . .
869         . . . . .
870         . . . . .
871         . . . . .
872         . . . . .
873         . . . . .
874         . . . . .
875         . . . . .
876         . . . . .
877         . . . . .
878         . . . . .
879         . . . . .
880         . . . . .
881         . . . . .
882         . . . . .
883         . . . . .
884         . . . . .
885         . . . . .
886         . . . . .
887         . . . . .
888         . . . . .
889         . . . . .
890         . . . . .
891         . . . . .
892         . . . . .
893         . . . . .
894         . . . . .
895         . . . . .
896         . . . . .
897         . . . . .
898         . . . . .
899         . . . . .
900         . . . . .
901         . . . . .
902         . . . . .
903         . . . . .
904         . . . . .
905         . . . . .
906         . . . . .
907         . . . . .
908         . . . . .
909         . . . . .
910         . . . . .
911         . . . . .
912         . . . . .
913         . . . . .
914         . . . . .
915         . . . . .
916         . . . . .
917         . . . . .
918         . . . . .
919         . . . . .
920         . . . . .
921         . . . . .
922         . . . . .
923         . . . . .
924         . . . . .
925         . . . . .
926         . . . . .
927         . . . . .
928         . . . . .
929         . . . . .
930         . . . . .
931         . . . . .
932         . . . . .
933         . . . . .
934         . . . . .
935         . . . . .
936         . . . . .
937         . . . . .
938         . . . . .
939         . . . . .
940         . . . . .
941         . . . . .
942         . . . . .
943         . . . . .
944         . . . . .
945         . . . . .
946         . . . . .
947         . . . . .
948         . . . . .
949         . . . . .
950         . . . . .
951         . . . . .
952         . . . . .
953         . . . . .
954         . . . . .
955         . . . . .
956         . . . . .
957         . . . . .
958         . . . . .
959         . . . . .
960         . . . . .
961         . . . . .
962         . . . . .
963         . . . . .
964         . . . . .
965         . . . . .
966         . . . . .
967         . . . . .
968         . . . . .
969         . . . . .
970         . . . . .
971         . . . . .
972         . . . . .
973         . . . . .
974         . . . . .
975         . . . . .
976         . . . . .
977         . . . . .
978         . . . . .
979         . . . . .
980         . . . . .
981         . . . . .
982         . . . . .
983         . . . . .
984         . . . . .
985         . . . . .
986         . . . . .
987         . . . . .
988         . . . . .
989         . . . . .
990         . . . . .
991         . . . . .
992         . . . . .
993         . . . . .
994         . . . . .
995         . . . . .
996         . . . . .
997         . . . . .
998         . . . . .
999         . . . . .
1000        . . . . .

```

```

138         64    2    2    .    }
139         .    .    .    .    if (nptr == endptr) {
140         .    .    .    .    fprintf(stderr, "missing A
141         .    .    .    .    matrix element");
142         .    .    .    .    goto _exit_main;
143         .    .    .    .    }
144         112    1    1    .    64    0    0    16    3
145         .    .    .    .    a->array[i] = e;
146         .    .    .    .    }
147         .    .    .    .    /* load matrix b*/
148         .    .    .    .    5    0    0    1    0
149         .    .    .    .    if (!(b = create_matrix(n, n)))
150         .    .    .    .    goto _exit_main;
151         .    .    .    .    .
152         170    2    2    .    67    0    0    17    0
153         .    .    .    .    for (i=0; i < n*n; i++) {
154         .    .    .    .    32    1    1    16    0    0    16    0
155         .    .    .    .    0    nptr = endptr;
156         144    1    1    .    48    0    0    16    0
157         .    .    .    .    e = strtod(nptr, &endptr);
158         128    1    1    .    48    0    0    0    0
159         .    .    .    .    if (errno) {
160         .    .    .    .    perror("");
161         .    .    .    .    goto _exit_main;
162         .    .    .    .    }
163         64    1    1    .    32    0    0    0    0
164         .    .    .    .    if (nptr == endptr) {
165         .    .    .    .    fprintf(stderr, "missing B
166         .    .    .    .    matrix element");
167         .    .    .    .    goto _exit_main;
168         .    .    .    .    }
169         112    1    1    .    64    0    0    16    4
170         .    .    .    .    4    b->array[i] = e;
171         .    .    .    .    }
172         .    .    .    .    .
173         8    1    1    .    2    0    0    0    0

```

```

164         .      .      .      0      clock_gettime(CLOCK_REALTIME, &t0);
165         .      .      .      .      .      .      .      .      .
166         .      .      .      .      .      .      .      .      .
167         .      .      .      .      .      .      .      .      .
168         .      .      .      .      .      .      .      .      .
169         .      .      .      .      .      .      .      .      .
170         .      .      .      .      .      .      .      .      .
171         .      .      .      .      .      .      .      .      .
172         .      .      .      .      .      .      .      .      .
173         .      .      .      .      .      .      .      .      .
174         .      .      .      .      .      .      .      .      .
175         .      .      .      .      .      .      .      .      .
176         .      .      .      .      .      .      .      .      .
177         .      .      .      .      .      .      .      .      .
178         .      .      .      .      .      .      .      .      .
179         .      .      .      .      .      .      .      .      .
180         .      .      .      .      .      .      .      .      .
181         .      .      .      .      .      .      .      .      .
182         .      .      .      .      .      .      .      .      .
183         .      .      .      .      .      .      .      .      .
184         .      .      .      .      .      .      .      .      .
185         .      .      .      .      .      .      .      .      .
186         .      .      .      .      .      .      .      .      .
187         .      .      .      .      .      .      .      .      .
188         .      .      .      .      .      .      .      .      .

```

				unsigned) lineno);					
189
			.	free(line);				.	.
190
			.	destroy_matrix(a);				.	.
191
			.	destroy_matrix(b);				.	.
192
			.	destroy_matrix(c);				.	.
193
			.	exit(1);				.	.
194	6	1	1	2	0	0	0	0	0
			0 }						
195
		
196
			.	matrix_t*				.	.
197
			.	matrix_multiply(matrix_t* m1, matrix_t*				.	.
			m2, int bs)					.	.
198	11	2	2	0	0	0	6	0	
			0 {						
199
			.	size_t n, en, i, j, k, kk, jj;				.	.
200
			.	double sum;				.	.
201
			.	matrix_t* mr;				.	.
202
		
203	3	1	1	2	0	0	1	0	
			0	n = m1->rows;					
204
		
205	11	1	1	5	0	0	1	0	
			0	if(!(mr = create_matrix(n,n))) return					
				NULL;					
206
		
207	9	2	2	3	0	0	1	0	
			0	en = bs*(n/bs);					
208
		
209	40	2	2	14	0	0	5	0	
			0	for(i=0; i<n; i++)					
210	160	1	1	56	0	0	20	0	
			0	for(j=0; j<n; j++)					
211	176	1	1	80	0	0	32	3	
			3	mr->array[i*n+j] = 0.0;					
212
		
213
			.	#if 1					
214

```

215         .         .         .         .         .         .         .         .
216         .         .         .         .         .         .         .         .
217         .         .         .         .         .         .         .         .
218         .         .         .         .         .         .         .         .
219         .         .         .         .         .         .         .         .
220         .         .         .         .         .         .         .         .
221         .         .         .         .         .         .         .         .
222         .         .         .         .         .         .         .         .
223         .         .         .         .         .         .         .         .
224         .         .         .         .         .         .         .         .
225         .         .         .         .         .         .         .         .
226         .         .         .         .         .         .         .         .
227         .         .         .         .         .         .         .         .
228         .         .         .         .         .         .         .         .
229         .         .         .         .         .         .         .         .
230         .         .         .         .         .         .         .         .
231         .         .         .         .         .         .         .         .
232         .         .         .         .         .         .         .         .
233         .         .         .         .         .         .         .         .
234         .         .         .         .         .         .         .         .
235         .         .         .         .         .         .         .         .
236         .         .         .         .         .         .         .         .
237         .         .         .         .         .         .         .         .
238         .         .         .         .         .         .         .         .
239         .         .         .         .         .         .         .         .
240         .         .         .         .         .         .         .         .

```

```

        if (1) {
            size_t j;
            size_t dim = 1024*1024*10;
            int *v = malloc(dim*sizeof(int));
            for (j = 0; j < dim; ++j)
                v[j] = -1;
            free(v);
        }
    #endif

    for(kk=0; kk<en; kk+=bs)
        for(jj=0; jj<en; jj+=bs)
            for(i=0; i<n; i++)
                for(j=jj; j<jj+bs; j++) {
                    sum = mr->array[i*n+j];
                    for(k=kk; k<kk+bs; k++)
                        sum += m1->array[i*n+k] * m2->
                            array[k*n+j];
                    mr->array[i*n+j] = sum;
                }
    return mr;
}

char*
read_line(FILE *fp)
{
    #define DEF_LINE_SZ 1024

```



```

241      .      .      .      .      .      .      .      .
242      6      0      0      .      0      0      0      4      0
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      14     1      1      .      6      1      0      2      1
246      .      .      .      .      .      .      .      .
247      .      .      .      .      .      .      .      .
248      .      .      .      .      .      .      .      .
249      .      .      .      .      .      .      .      .
250      .      .      .      .      .      .      .      .
251      1,001  5      5      .      334    1      0      67      0
252      .      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      .      .      .      .      .      .      .      .
259      .      .      .      .      .      .      .      .
260      .      .      .      .      .      .      .      .
261      .      .      .      .      .      .      .      .
262      8      1      1      .      2      0      0      0      0
263      .      .      .      .      .      .      .      .
264      .      .      .      .      .      .      .      .
265      12     1      1      .      4      0      0      4      0
266      .      .      .      .      .      .      .      .
267      12     1      1      .      4      0      0      0      0

```

```

268         0 }
269     . . . . .
270     . . . . .
271     . matrix_t*
272     . create_matrix(size_t rows, size_t cols)
273     30 1 1 0 0 0 15 0
274     0 {
275     . . . . .
276     . matrix_t * m;
277     . . . . .
278     .
279     30 2 2 9 1 0 3 0
280     0 if (!(m = malloc(sizeof(matrix_t)))) {
281     . . . . .
282     . perror("");
283     . . . . .
284     . return NULL;
285     . . . . .
286     . }
287     . . . . .
288     9 1 1 6 0 0 3 0
289     0 m->rows = rows;
290     9 1 1 6 0 0 3 0
291     0 m->cols = cols;
292     51 2 2 21 0 0 3 0
293     0 if (!(m->array = malloc(sizeof(double)
294     *rows*cols))) {
295     . . . . .
296     . free(m);
297     . . . . .
298     . perror("");
299     . . . . .
300     . return NULL;
301     . . . . .
302     . }
303     . . . . .
304     .
305     3 1 1 3 0 0 0 0
306     0 return m;
307     18 1 1 6 0 0 0 0
308     0 }
309     . . . . .
310     .
311     . void
312     . . . . .
313     . destroy_matrix(matrix_t* m)
314     27 1 1 0 0 0 12 0
315     0 {

```

```

294      9    0    0      3    0    0      0    0
      0    if (!m) return;
295      24   1    1      12    0    0      0    0
      0    free(m->array);
296      24   1    1      9     0    0      0    0
      0    free(m);
297      18    0    0      6     0    0      0    0
      0 }
298      .    .    .      .    .    .      .    .
      .
299      .    .    .      .    .    .      .    .
      .    int
300      .    .    .      .    .    .      .    .
      .    print_matrix(FILE* fp, matrix_t* m)
301      10    2    2      0     0    0      5     0
      0 {
302      .    .    .      .    .    .      .    .
      .    size_t i, j;
303      .    .    .      .    .    .      .    .
      .    size_t n;
304      3     1    1      2     0    0      1     0
      0    n = m->rows;
305      13    1    1      6     1    1      0     0
      0    if (fprintf(fp, "%lu", (unsigned long)
      m->rows) < 0) {
306      .    .    .      .    .    .      .    .
      .    perror("");
307      .    .    .      .    .    .      .    .
      .    return -1;
308      .    .    .      .    .    .      .    .
      .    }
309      40    2    2      14    0    0      5     0
      0    for(i=0; i<n; i++)
310      160   2    2      56    0    0      20    0
      0    for (j=0; j<n; j++)
311      352   3    3      160   0    0      0     0
      0    if (fprintf(fp, " %g", m->array[i*n
      +j]) < 0) {
312      .    .    .      .    .    .      .    .
      .    perror("");
313      .    .    .      .    .    .      .    .
      .    return -1;
314      .    .    .      .    .    .      .    .
      .    }
315      10    1    1      4     0    0      0     0
      0    if (fprintf(fp, "\n") < 0) {
316      .    .    .      .    .    .      .    .
      .    perror("");
317      .    .    .      .    .    .      .    .
      .    return -1;
318      .    .    .      .    .    .      .    .
      .    }
319      1     1    1      0     0    0      0     0
      0    return 0;

```

320	6	1	1	2	0	0	0	0	0
321	0 }								
322	-----								
323	Ir	I1mr	I1Lmr	Dr	D1mr	D1Lmr	Dw	D1mw	
324	DLmw								
324	-----								
325	146,808,382	120	119	52,432,069	29	23	20,972,236	1,310,735	
	1,310,733	events annotated							

```
1 5 125 118 38 87 87 115 168 73 98 118 86 89 43 46 93 156 132 47
   79 148 79 110 62 57 137
2 017, and GNU GPL'd, by Nicholas Nethercote et al.
3 ==663== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4 ==663== Command: /tmp/02-mmult
5 ==663== Parent PID: 597
6 ==663==
7 --663-- Warning: Cannot auto-detect cache config, using
   defaults.
8 --663--          Run with -v to see.
9 ==663==
10 ==663== I    refs:          147,187,051
11 ==663== I1   misses:          2,407
12 ==663== L1i  misses:          2,389
13 ==663== I1   miss rate:          0.00%
14 ==663== L1i  miss rate:          0.00%
15 ==663==
16 ==663== D    refs:          73,550,901 (52,528,981 rd +
   21,021,920 wr)
17 ==663== D1   misses:          1,314,830 (    3,517 rd +
   1,311,313 wr)
18 ==663== L1d  misses:          1,314,183 (    2,915 rd +
   1,311,268 wr)
19 ==663== D1   miss rate:          1.8% (    0.0% +
   6.2% )
20 ==663== L1d  miss rate:          1.8% (    0.0% +
   6.2% )
21 ==663==
22 ==663== LL refs:          1,317,237 (    5,924 rd +
   1,311,313 wr)
23 ==663== LL   misses:          1,316,572 (    5,304 rd +
   1,311,268 wr)
24 ==663== LL   miss rate:          0.6% (    0.0% +
   6.2% )
25 -----
26 I1 cache:          32768 B, 32 B, 4-way associative
27 D1 cache:          32768 B, 32 B, 4-way associative
28 LL cache:          524288 B, 32 B, 8-way associative
29 Command:          /tmp/02-mmult
30 Data file:          cachegrind.out.663
```

```

31 Events recorded:  Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
32 Events shown:    Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
34 Thresholds:      0.1 100 100 100 100 100 100 100 100
35 Include dirs:
36 User annotated:   /root/CARPETA/tp2-2020-2q-src/main.c
37 Auto-annotation: off
38
39 -----
40 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
41      DLmw
42 147,187,051 2,407 2,389 52,528,981 3,517 2,915 21,021,920
43      1,311,313 1,311,268 PROGRAM TOTALS
44 -----
45 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
46      DLmw      file:function
47 146,806,731 29 29 52,431,722 23 23 20,971,983
48      1,310,725 1,310,725 /root/CARPETA/tp2-2020-2q-src/main.c:
49      matrix_multiply
50 -----
51 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
52 -----
53 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
54      DLmw
55 -- line 18 -----
56      .      .      .      .      .      .      .
57      .      .      .      .      .      .      .
58      .      .      .      .      .      .      .
59      .      .      .      .      .      .      .
60      .      .      .      .      .      .      .
61      .      .      .      .      .      .      .
62      .      .      .      .      .      .      .
63      .      .      .      .      .      .      .
64      .      .      .      .      .      .      .
65      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .
69      .      .      .      .      .      .      .
70      .      .      .      .      .      .      .
71      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .
76      .      .      .      .      .      .      .
77      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .
81      .      .      .      .      .      .      .
82      .      .      .      .      .      .      .
83      .      .      .      .      .      .      .
84      .      .      .      .      .      .      .
85      .      .      .      .      .      .      .
86      .      .      .      .      .      .      .
87      .      .      .      .      .      .      .
88      .      .      .      .      .      .      .
89      .      .      .      .      .      .      .
90      .      .      .      .      .      .      .
91      .      .      .      .      .      .      .
92      .      .      .      .      .      .      .
93      .      .      .      .      .      .      .
94      .      .      .      .      .      .      .
95      .      .      .      .      .      .      .
96      .      .      .      .      .      .      .
97      .      .      .      .      .      .      .
98      .      .      .      .      .      .      .
99      .      .      .      .      .      .      .
100     .      .      .      .      .      .      .
101     .      .      .      .      .      .      .
102     .      .      .      .      .      .      .
103     .      .      .      .      .      .      .
104     .      .      .      .      .      .      .
105     .      .      .      .      .      .      .
106     .      .      .      .      .      .      .
107     .      .      .      .      .      .      .
108     .      .      .      .      .      .      .
109     .      .      .      .      .      .      .
110     .      .      .      .      .      .      .
111     .      .      .      .      .      .      .
112     .      .      .      .      .      .      .
113     .      .      .      .      .      .      .
114     .      .      .      .      .      .      .
115     .      .      .      .      .      .      .
116     .      .      .      .      .      .      .
117     .      .      .      .      .      .      .
118     .      .      .      .      .      .      .
119     .      .      .      .      .      .      .
120     .      .      .      .      .      .      .
121     .      .      .      .      .      .      .
122     .      .      .      .      .      .      .
123     .      .      .      .      .      .      .
124     .      .      .      .      .      .      .
125     .      .      .      .      .      .      .
126     .      .      .      .      .      .      .
127     .      .      .      .      .      .      .
128     .      .      .      .      .      .      .
129     .      .      .      .      .      .      .
130     .      .      .      .      .      .      .
131     .      .      .      .      .      .      .
132     .      .      .      .      .      .      .
133     .      .      .      .      .      .      .
134     .      .      .      .      .      .      .
135     .      .      .      .      .      .      .
136     .      .      .      .      .      .      .
137     .      .      .      .      .      .      .
138     .      .      .      .      .      .      .
139     .      .      .      .      .      .      .
140     .      .      .      .      .      .      .
141     .      .      .      .      .      .      .
142     .      .      .      .      .      .      .
143     .      .      .      .      .      .      .
144     .      .      .      .      .      .      .
145     .      .      .      .      .      .      .
146     .      .      .      .      .      .      .
147     .      .      .      .      .      .      .
148     .      .      .      .      .      .      .
149     .      .      .      .      .      .      .
150     .      .      .      .      .      .      .
151     .      .      .      .      .      .      .
152     .      .      .      .      .      .      .
153     .      .      .      .      .      .      .
154     .      .      .      .      .      .      .
155     .      .      .      .      .      .      .
156     .      .      .      .      .      .      .
157     .      .      .      .      .      .      .
158     .      .      .      .      .      .      .
159     .      .      .      .      .      .      .
160     .      .      .      .      .      .      .
161     .      .      .      .      .      .      .
162     .      .      .      .      .      .      .
163     .      .      .      .      .      .      .
164     .      .      .      .      .      .      .
165     .      .      .      .      .      .      .
166     .      .      .      .      .      .      .
167     .      .      .      .      .      .      .
168     .      .      .      .      .      .      .
169     .      .      .      .      .      .      .
170     .      .      .      .      .      .      .
171     .      .      .      .      .      .      .
172     .      .      .      .      .      .      .
173     .      .      .      .      .      .      .
174     .      .      .      .      .      .      .
175     .      .      .      .      .      .      .
176     .      .      .      .      .      .      .
177     .      .      .      .      .      .      .
178     .      .      .      .      .      .      .
179     .      .      .      .      .      .      .
180     .      .      .      .      .      .      .
181     .      .      .      .      .      .      .
182     .      .      .      .      .      .      .
183     .      .      .      .      .      .      .
184     .      .      .      .      .      .      .
185     .      .      .      .      .      .      .
186     .      .      .      .      .      .      .
187     .      .      .      .      .      .      .
188     .      .      .      .      .      .      .
189     .      .      .      .      .      .      .
190     .      .      .      .      .      .      .
191     .      .      .      .      .      .      .
192     .      .      .      .      .      .      .
193     .      .      .      .      .      .      .
194     .      .      .      .      .      .      .
195     .      .      .      .      .      .      .
196     .      .      .      .      .      .      .
197     .      .      .      .      .      .      .
198     .      .      .      .      .      .      .
199     .      .      .      .      .      .      .
200     .      .      .      .      .      .      .
201     .      .      .      .      .      .      .
202     .      .      .      .      .      .      .
203     .      .      .      .      .      .      .
204     .      .      .      .      .      .      .
205     .      .      .      .      .      .      .
206     .      .      .      .      .      .      .
207     .      .      .      .      .      .      .
208     .      .      .      .      .      .      .
209     .      .      .      .      .      .      .
210     .      .      .      .      .      .      .
211     .      .      .      .      .      .      .
212     .      .      .      .      .      .      .
213     .      .      .      .      .      .      .
214     .      .      .      .      .      .      .
215     .      .      .      .      .      .      .
216     .      .      .      .      .      .      .
217     .      .      .      .      .      .      .
218     .      .      .      .      .      .      .
219     .      .      .      .      .      .      .
220     .      .      .      .      .      .      .
221     .      .      .      .      .      .      .
222     .      .      .      .      .      .      .
223     .      .      .      .      .      .      .
224     .      .      .      .      .      .      .
225     .      .      .      .      .      .      .
226     .      .      .      .      .      .      .
227     .      .      .      .      .      .      .
228     .      .      .      .      .      .      .
229     .      .      .      .      .      .      .
230     .      .      .      .      .      .      .
231     .      .      .      .      .      .      .
232     .      .      .      .      .      .      .
233     .      .      .      .      .      .      .
234     .      .      .      .      .      .      .
235     .      .      .      .      .      .      .
236     .      .      .      .      .      .      .
237     .      .      .      .      .      .      .
238     .      .      .      .      .      .      .
239     .      .      .      .      .      .      .
240     .      .      .      .      .      .      .
241     .      .      .      .      .      .      .
242     .      .      .      .      .      .      .
243     .      .      .      .      .      .      .
244     .      .      .      .      .      .      .
245     .      .      .      .      .      .      .
246     .      .      .      .      .      .      .
247     .      .      .      .      .      .      .
248     .      .      .      .      .      .      .
249     .      .      .      .      .      .      .
250     .      .      .      .      .      .      .
251     .      .      .      .      .      .      .
252     .      .      .      .      .      .      .
253     .      .      .      .      .      .      .
254     .      .      .      .      .      .      .
255     .      .      .      .      .      .      .
256     .      .      .      .      .      .      .
257     .      .      .      .      .      .      .
258     .      .      .      .      .      .      .
259     .      .      .      .      .      .      .
260     .      .      .      .      .      .      .
261     .      .      .      .      .      .      .
262     .      .      .      .      .      .      .
263     .      .      .      .      .      .      .
264     .      .      .      .      .      .      .
265     .      .      .      .      .      .      .
266     .      .      .      .      .      .      .
267     .      .      .      .      .      .      .
268     .      .      .      .      .      .      .
269     .      .      .      .      .      .      .
270     .      .      .      .      .      .      .
271     .      .      .      .      .      .      .
272     .      .      .      .      .      .      .
273     .      .      .      .      .      .      .
274     .      .      .      .      .      .      .
275     .      .      .      .      .      .      .
276     .      .      .      .      .      .      .
277     .      .      .      .      .      .      .
278     .      .      .      .      .      .      .
279     .      .      .      .      .      .      .
280     .      .      .      .      .      .      .
281     .      .      .      .      .      .      .
282     .      .      .      .      .      .      .
283     .      .      .      .      .      .      .
284     .      .      .      .      .      .      .
285     .      .      .      .      .      .      .
286     .      .      .      .      .      .      .
287     .      .      .      .      .      .      .
288     .      .      .      .      .      .      .
289     .      .      .      .      .      .      .
290     .      .      .      .      .      .      .
291     .      .      .      .      .      .      .
292     .      .      .      .      .      .      .
293     .      .      .      .      .      .      .
294     .      .      .      .      .      .      .
295     .      .      .      .      .      .      .
296     .      .      .      .      .      .      .
297     .      .      .      .      .      .      .
298     .      .      .      .      .      .      .
299     .      .      .      .      .      .      .
300     .      .      .      .      .      .      .
301     .      .      .      .      .      .      .
302     .      .      .      .      .      .      .
303     .      .      .      .      .      .      .
304     .      .      .      .      .      .      .
305     .      .      .      .      .      .      .
306     .      .      .      .      .      .      .
307     .      .      .      .      .      .      .
308     .      .      .      .      .      .      .
309     .      .      .      .      .      .      .
310     .      .      .      .      .      .      .
311     .      .      .      .      .      .      .
312     .      .      .      .      .      .      .
313     .      .      .      .      .      .      .
314     .      .      .      .      .      .      .
315     .      .      .      .      .      .      .
316     .      .      .      .      .      .      .
317     .      .      .      .      .      .      .
318     .      .      .      .      .      .      .
319     .      .      .      .      .      .      .
320     .      .      .      .      .      .      .
321     .      .      .      .      .      .      .
322     .      .      .      .      .      .      .
323     .      .      .      .      .      .      .
324     .      .      .      .      .      .      .
325     .      .      .      .      .      .      .
326     .      .      .      .      .      .      .
327     .      .      .      .      .      .      .
328     .      .      .      .      .      .      .
329     .      .      .      .      .      .      .
330     .      .      .      .      .      .      .
331     .      .      .      .      .      .      .
332     .      .      .      .      .      .      .
333     .      .      .      .      .      .      .
334     .      .      .      .      .      .      .
335     .      .      .      .      .      .      .
336     .      .      .      .      .      .      .
337     .      .      .      .      .      .      .
338     .      .      .      .      .      .      .
339     .      .      .      .      .      .      .
340     .      .      .      .      .      .      .
341     .      .      .      .      .      .      .
342     .      .      .      .      .      .      .
343     .      .      .      .      .      .      .
344     .      .      .      .      .      .      .
345     .      .      .      .      .      .      .
346     .      .      .      .      .      .      .
347     .      .      .      .      .      .      .
348     .      .      .      .      .      .      .
349     .      .      .      .      .      .      .
350     .      .      .      .      .      .      .
351     .      .      .      .      .      .      .
352     .      .      .      .      .      .      .
353     .      .      .      .      .      .      .
354     .      .      .      .      .      .      .
355     .      .      .      .      .      .      .
356     .      .      .      .      .      .      .
357     .      .      .      .      .      .      .
358     .      .      .      .      .      .      .
359     .      .      .      .      .      .      .
360     .      .      .      .      .      .      .
361     .      .      .      .      .      .      .
362     .      .      .      .      .      .      .
363     .      .      .      .      .      .      .
364     .      .      .      .      .      .      .
365     .      .      .      .      .      .      .
366     .      .      .      .      .      .      .
367     .      .      .      .      .      .      .
368     .      .      .      .      .      .      .
369     .      .      .      .      .      .      .
370     .      .      .      .      .      .      .
371     .      .      .      .      .      .      .
372     .      .      .      .      .      .      .
373     .      .      .      .      .      .      .
374     .      .      .      .      .      .      .
375     .      .      .      .      .      .      .
376     .      .      .      .      .      .      .
377     .      .      .      .      .      .      .
378     .      .      .      .      .      .      .
379     .      .      .      .      .      .      .
380     .      .      .      .      .      .      .
381     .      .      .      .      .      .      .
382     .      .      .      .      .      .      .
383     .      .      .      .      .      .      .
384     .      .      .      .      .      .      .
385     .      .      .      .      .      .      .
386     .      .      .      .      .      .      .
387     .      .      .      .      .      .      .
388     .      .      .      .      .      .      .
389     .      .      .      .      .      .      .
390     .      .      .      .      .      .      .
391     .      .      .      .      .      .      .
392     .      .      .      .      .      .      .
393     .      .      .      .      .      .      .
394     .      .      .      .      .      .      .
395     .      .      .      .      .      .      .
396     .      .      .      .      .      .      .
397     .      .      .      .      .      .      .
398     .      .      .      .      .      .      .
399     .      .      .      .      .      .      .
400     .      .      .      .      .      .      .
401     .      .      .      .      .      .      .
402     .      .      .      .      .      .      .
403     .      .      .      .      .      .      .
404     .      .      .      .      .      .      .
405     .      .      .      .      .      .      .
406     .      .      .      .      .      .      .
407     .      .      .      .      .      .      .
408     .      .      .      .      .      .      .
409     .      .      .      .      .      .      .
410     .      .      .      .      .      .      .
411     .      .      .      .      .      .      .
412     .      .      .      .      .      .      .
413     .      .      .      .      .      .      .
414     .      .      .      .      .      .      .
415     .      .      .      .      .      .      .
416     .      .      .      .      .      .      .
417     .      .      .      .      .      .      .
418     .      .      .      .      .      .      .
419     .      .      .      .      .      .      .
420     .      .      .      .      .      .      .
421     .      .      .      .      .      .      .
422     .      .      .      .      .      .      .
423     .      .      .      .      .      .      .
424     .      .      .      .      .      .      .
425     .      .      .      .      .      .      .
426     .      .      .      .      .      .      .
427     .      .      .      .      .      .      .
428     .      .      .      .      .      .      .
429     .      .      .      .      .      .      .
430     .      .      .      .      .      .      .
431     .      .      .      .      .      .      .
432     .      .      .      .      .      .      .
433     .      .      .      .      .      .      .
434     .      .      .      .      .      .      .
435     .      .      .      .      .      .      .
436     .      .      .      .      .      .      .
437     .      .      .      .      .      .      .
438     .      .      .      .      .      .      .
439     .      .      .      .      .      .      .
440     .      .      .      .      .      .      .
441     .      .      .      .      .      .      .
442     .      .      .      .      .      .      .
443     .      .      .      .      .      .      .
444     .      .      .      .      .      .      .
445     .      .      .      .      .      .      .
446     .      .      .      .      .      .      .
447     .      .      .      .      .      .      .
448     .      .      .      .      .      .      .
449     .      .      .      .      .      .      .
450     .      .      .      .      .      .      .
451     .      .      .      .      .      .      .
452     .      .      .      .      .      .      .
453     .      .      .      .      .      .      .
454     .      .      .      .      .      .      .
455     .      .      .      .      .      .      .
456     .      .      .      .      .      .      .
457     .      .      .      .      .      .      .
458     .      .      .      .      .      .      .
459     .      .      .      .      .      .      .
460     .      .      .      .      .      .      .
461     .      .      .      .      .      .      .
462     .      .      .      .      .      .      .
463     .      .      .      .      .      .      .
464     .      .      .      .      .      .      .
465     .      .      .      .      .      .      .
466     .      .      .      .      .      .      .
467     .      .      .      .      .      .      .
468     .      .      .      .      .      .      .
469     .      .      .      .      .      .      .
470     .      .      .      .      .      .      .
471     .      .      .      .      .      .      .
472     .      .      .      .      .      .      .
473     .      .      .      .      .      .      .
474     .      .      .      .      .      .      .
475     .      .      .      .      .      .      .
476     .      .      .      .      .      .      .
477     .      .      .      .      .      .      .
478     .      .      .      .      .      .      .
479     .      .      .      .      .      .      .
480     .      .      .      .      .      .      .
481     .      .      .      .      .      .      .
482     .      .      .      .      .      .      .
483     .      .      .      .      .      .      .
484     .      .      .      .      .      .      .
485     .      .      .      .      .      .      .
486     .      .      .      .      .      .      .
487     .      .      .      .      .      .      .
488     .      .      .      .      .      .      .
489     .      .      .      .      .      .      .
490     .      .      .      .      .      .      .
491     .      .      .      .      .      .      .
492     .      .      .      .      .      .      .
493     .      .      .      .      .      .      .
494     .      .      .      .      .      .      .
495     .      .      .      .      .      .      .
496     .      .      .      .      .      .      .
497     .      .      .      .      .      .      .
498     .      .      .      .      .      .      .
499     .      .      .      .      .      .      .
500     .      .      .      .      .      .      .
501     .      .      .      .      .      .      .
502     .      .      .      .      .      .      .
503     .      .      .      .      .      .      .
504     .      .      .      .      .      .      .
505     .      .      .      .      .      .      .
506     .      .      .      .      .      .      .
507     .      .      .      .      .      .      .
508     .      .      .      .      .      .      .
509     .      .      .      .      .      .      .
510     .      .      .      .      .      .      .
511     .      .      .      .      .      .      .
512     .      .      .      .      .      .      .
513     .      .      .      .      .      .      .
514     .      .      .      .      .      .      .
515     .      .      .      .      .      .      .
516     .      .      .      .      .      .      .
517     .      .      .      .      .      .      .
518     .      .      .      .      .      .      .
519     .      .      .      .      .      .      .
520     .      .      .      .      .      .      .
521     .      .      .      .      .      .      .
522     .      .      .      .      .      .      .
523     .      .      .      .      .      .      .
524     .      .      .      .      .      .      .
525     .      .      .      .      .      .      .
526     .      .      .      .      .      .      .
527     .      .      .      .      .      .      .
528     .      .      .      .      .      .      .
529     .      .      .      .      .      .      .
530     .      .      .      .      .      .      .
531     .      .      .      .      .      .      .
532     .      .      .      .      .      .      .
533     .      .      .      .      .      .      .
534     .      .      .      .      .      .      .
535     .      .      .      .      .      .      .
536     .      .      .      .      .      .      .
537     .      .      .      .      .      .      .
538     .      .      .      .      .      .      .
539     .      .      .      .      .      .      .
540     .      .      .      .      .      .      .
541     .      .      .      .      .      .      .
542     .      .      .      .      .      .      .
543     .      .      .      .      .      .      .
544     .      .      .      .      .      .      .
545     .      .      .      .      .      .      .
546     .      .      .      .      .      .      .
547     .      .      .      .      .      .      .
548     .      .      .      .      .      .      .
549     .      .      .      .      .      .      .
550     .      .      .      .      .      .      .
551     .      .      .      .      .      .      .
552     .      .      .      .      .      .      .
553     .      .      .      .      .      .      .
554     .      .      .      .      .      .      .
555     .      .      .      .      .      .      .
556     .      .      .      .      .      .      .
557     .      .      .      .      .      .      .
558     .      .      .      .      .      .      .
559     .      .      .      .      .      .      .
560     .      .      .      .      .      .      .
561     .      .      .      .      .      .      .
562     .      .      .      .      .      .      .
563     .      .      .      .      .      .      .
564     .      .      .      .      .      .      .
565     .      .      .      .      .      .      .
566     .      .      .      .      .      .      .
567     .      .      .      .      .      .      .
568     .      .      .      .      .      .      .
569     .      .      .      .      .      .      .
570     .      .      .      .      .      .      .
571     .      .      .      .      .      .      .
572     .      .      .      .      .      .      .
573     .      .      .      .      .      .      .
574     .      .      .      .      .      .      .
575     .      .      .      .      .      .      .
576     .      .      .      .      .      .      .
577     .      .      .      .      .      .      .
578     .      .      .      .      .      .      .
579     .      .      .      .      .      .      .
580     .      .      .      .      .      .      .
581     .      .      .      .      .      .      .
582     .      .      .      .      .      .      .
583     .      .      .      .      .      .      .
584     .      .      .      .      .      .      .
585     .      .      .      .      .      .      .
586     .      .      .      .      .      .      .
587     .      .      .      .      .      .      .
588     .      .      .      .      .      .      .
589     .      .      .      .      .      .      .
590     .      .      .      .      .      .      .
591     .      .      .      .      .      .      .
592     .      .      .      .      .      .      .
593     .      .      .      .      .      .      .
594     .      .      .      .      .      .      .
595     .      .      .      .      .      .      .
596     .      .      .      .      .      .      .
597     .      .      .      .      .      .      .
598     .      .      .      .      .      .      .
599     .      .      .      .      .      .      .
600     .      .      .      .      .      .      .
601     .      .      .      .      .      .      .
602     .      .      .      .      .      .      .
603     .      .      .      .      .      .      .
604     .      .      .      .      .      .      .
605     .      .      .      .      .      .      .
606     .      .      .      .      .      .      .
607     .      .      .      .      .      .      .
608     .      .      .      .      .      .      .
609     .      .      .      .      .      .      .
610     .      .      .      .      .      .      .
611     .      .      .      .      .      .      .
612     .      .      .      .      .      .      .
613     .      .      .      .      .      .      .
614     .      .      .      .      .      .      .
615     .      .      .      .      .      .      .
616     .      .      .      .      .      .      .
617     .      .      .      .      .      .      .
618     .      .      .      .      .      .      .
619     .      .      .      .      .      .      .
620     .      .      .      .      .      .      .
621     .      .      .      .      .      .      .
622     .      .      .      .      .      .      .
623     .      .      .      .      .      .      .
624     .      .      .      .      .      .      .
625     .      .      .      .      .      .      .
626     .      .      .      .      .      .      .
627     .      .      .      .      .      .      .
628     .      .      .      .      .      .      .
629     .      .      .      .      .      .      .
630     .      .      .      .      .      .      .
631     .      .      .      .      .      .      .
632     .      .      .      .      .      .      .
633     .      .      .      .      .      .      .
634     .      .      .      .      .      .      .
635     .      .      .      .      .      .      .
636     .      .      .      .      .      .      .
637     .      .      .      .      .      .      .
638     .      .      .      .      .      .      .
639     .      .      .      .      .      .      .
640     .      .      .      .      .      .      .
641     .      .      .      .      .      .      .
642     .      .      .      .      .      .      .
643     .      .      .      .      .      .      .
644     .      .      .      .      .      .      .
645     .      .      .      .      .      .      .
646     .      .      .      .      .      .      .
647     .      .      .      .      .      .      .
648     .      .      .      .      .      .      .
649     .      .      .      .      .      .      .
650     .      .      .      .      .      .      .
651     .      .      .      .      .      .      .
652     .      .      .      .      .      .      .
653     .      .      .      .      .      .      .
654     .      .      .      .      .      .      .
655     .      .      .      .      .      .      .
656     .      .      .      .      .      .      .
657     .      .      .      .      .      .      .
658     .      .      .      .      .      .      .
659     .      .      .      .      .      .      .
660     .      .      .      .      .      .      .
661     .      .      .      .      .      .      .
662     .      .      .      .      .      .      .
663     .      .      .      .      .      .      .
664     .      .      .      .      .      .      .
665     .      .      .      .      .      .      .
666     .      .      .      .      .      .      .
667     .      .      .      .      .      .      .
668     .      .      .      .      .      .      .
669     .      .      .      .      .      .      .
670     .      .      .      .      .      .      .
671     .      .      .      .      .      .      .
672     .      .      .      .      .      .      .
673     .      .      .      .      .      .      .
674     .      .      .      .      .      .      .
675     .      .      .      .      .      .      .
676     .      .      .      .      .      .      .
677     .      .      .      .      .      .      .
678     .      .      .      .      .      .      .
679     .      .      .      .      .      .      .
680     .      .      .      .      .      .      .
681     .      .      .      .      .      .      .
682     .      .      .      .      .      .      .
683     .      .      .      .      .      .      .
684     .      .      .      .      .      .      .
685     .      .      .      .      .      .      .
686     .      .      .      .      .      .      .
687     .      .      .      .      .      .      .
688     .      .      .      .      .      .      .
689     .      .      .      .      .      .      .
690     .      .      .      .      .      .      .
691     .      .      .      .      .      .      .
692     .      .      .      .      .      .      .
693     .      .      .      .      .      .      .
694     .      .      .      .      .      .      .
695     .      .      .      .      .      .      .
696     .      .      .      .      .      .      .
697     .      .      .      .      .      .      .
698     .      .      .      .      .      .      .
699     .      .      .      .      .      .      .
700     .      .      .      .      .      .      .
701     .      .      .      .      .      .      .
702     .      .      .      .      .      .      .
703     .      .      .      .      .      .      .
704     .      .      .      .      .      .      .
705     .      .      .      .      .      .      .
706     .      .      .      .      .      .      .
707     .      .      .      .      .      .      .
708     .      .      .      .      .      .      .
70
```

```

63      10      2      0      {      0      0      0      5      0
64      .      .      .      .      .      .      .      .      .
65      .      .      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .      .
69      1      0      0      0      0      0      0      1      0
70      .      .      .      .      .      .      .      .      .
71      .      .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .      .
75      2      1      1      0      0      0      1      0
76      .      .      .      .      .      .      .      .      .
77      .      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .      .
79      .      .      .      .      .      .      .      .      .
80      .      .      .      .      .      .      .      .      .
81      25      4      4      9      0      0      1      0
82      10      1      1      4      0      0      6      0
83      .      .      .      .      .      .      .      .      .
84      18      3      3      8      1      0      2      0
85      .      .      .      .      .      .      .      .      .
86      6      0      0      2      0      0      0      0
87      9      0      0      4      0      0      0      0
88      .      .      .      .      .      .      .      .      .
89      .      .      .      .      .      .      .      .      .

```

```

90      2  1  1      .      /* parse dimension */
91      10  1  1      0      nptr = line;
92      8  1  1      0      l = strtol(nptr, &endptr, 10);
93      .  .  .      0      if (errno) {
94      .  .  .      .      perror("");
95      .  .  .      .      goto _exit_main;
96      4  2  2      .      }
97      .  .  .      0      if (nptr == endptr) {
98      .  .  .      .      fprintf(stderr, "missing
dimension");
99      .  .  .      .      goto _exit_main;
100     3  2  2      .      }
101     .  .  .      0      if (l < 1) {
102     .  .  .      .      fprintf(stderr, "invalid
dimension");
103     .  .  .      .      goto _exit_main;
104     2  1  1      .      }
105     .  .  .      0      n = (size_t) l;
106     .  .  .      .      #if 0
107     .  .  .      .      /* parse block size */
108     .  .  .      .      nptr = endptr;
109     .  .  .      .      l = strtol(nptr, &endptr, 10);
110     .  .  .      .      if (errno) {
111     .  .  .      .      perror("");
112     .  .  .      .      goto _exit_main;
113     .  .  .      .      }
114     -- line 75 -----
115     .  .  .      .      }
116     .  .  .      .      }

```

```

117         .         .         .         .         .         .         .         .
118         .         .         .         .         .         .         .         .
119         .         .         .         .         .         .         .         .
120         .         .         .         .         .         .         .         .
121         .         .         .         .         .         .         .         .
122         .         .         .         .         .         .         .         .
123         .         .         .         .         .         .         .         .
124         .         .         .         .         .         .         .         .
125         .         .         .         .         .         .         .         .
126         .         .         .         .         .         .         .         .
127         .         .         .         .         .         .         .         .
128         .         .         .         .         .         .         .         .
129         .         .         .         .         .         .         .         .
130         .         .         .         .         .         .         .         .
131         .         .         .         .         .         .         .         .
132         .         .         .         .         .         .         .         .
133         .         .         .         .         .         .         .         .
134         .         .         .         .         .         .         .         .
135         .         .         .         .         .         .         .         .
136         .         .         .         .         .         .         .         .
137         .         .         .         .         .         .         .         .
138         .         .         .         .         .         .         .         .
139         .         .         .         .         .         .         .         .
140         .         .         .         .         .         .         .         .
141         .         .         .         .         .         .         .         .
142         .         .         .         .         .         .         .         .

        bs = (size_t) 1;

        if (n % bs) {
            fprintf(stderr, "block size doesn't match");
            goto _exit_main;
        }
    #else
        2    0    0        1    0    0        1    0
        0        bs = n;
    #endif

        /* load matrix a */
        11    1    1        5    1    0        1    0
        0        if (!(a = create_matrix(n, n)))
            goto _exit_main;

        260    3    3        103    0    0        26    0
        0        for (i=0; i < n*n; i++) {
        50    0    0        25    0    0        25    0
        0        nptr = endptr;
        225    1    1        75    0    0        25    0
        0        e = strtod(nptr, &endptr);
        200    1    1        75    0    0        0    0
        0        if (errno) {
            perror("");
            goto _exit_main;
        }
        100    2    2        50    0    0        0    0
        0        if (nptr == endptr) {
            fprintf(stderr, "missing A
matrix element");
            goto _exit_main;
        }
        175    1    1        100    0    0        25    5
        5        a->array[i] = e;

```



```

143         .      .      .      .      .      .      .      .      .
144         .      .      .      .      .      .      .      .      .
145         11      1      1      .      .      .      .      .      .
146         .      .      .      .      .      .      .      .      .
147         .      .      .      .      .      .      .      .      .
148         260     2      2      .      .      .      .      .      .
149         50      1      1      .      .      .      .      .      .
150         225     1      1      .      .      .      .      .      .
151         200     1      1      .      .      .      .      .      .
152         .      .      .      .      .      .      .      .      .
153         .      .      .      .      .      .      .      .      .
154         .      .      .      .      .      .      .      .      .
155         100     1      1      .      .      .      .      .      .
156         .      .      .      .      .      .      .      .      .
157         .      .      .      .      .      .      .      .      .
158         .      .      .      .      .      .      .      .      .
159         175     1      1      .      .      .      .      .      .
160         .      .      .      .      .      .      .      .      .
161         .      .      .      .      .      .      .      .      .
162         8       1      1      .      .      .      .      .      .
163         .      .      .      .      .      .      .      .      .
164         .      .      .      .      .      .      .      .      .
165         13      2      2      .      .      .      .      .      .
166         .      .      .      .      .      .      .      .      .
167         .      .      .      .      .      .      .      .      .
168         8       1      1      .      .      .      .      .      .

```

```

    }
    /* load matrix b */
    if (!(b = create_matrix(n, n)))
        goto _exit_main;

    for (i=0; i < n*n; i++) {
        nptr = endptr;
        e = strtod(nptr, &endptr);
        if (errno) {
            perror("");
            goto _exit_main;
        }
        if (nptr == endptr) {
            fprintf(stderr, "missing B
matrix element");
            goto _exit_main;
        }
        b->array[i] = e;
    }

    clock_gettime(CLOCK_REALTIME, &t0);

    /* multiply matrixes */
    if (!(c = matrix_multiply(a, b, bs))
    )
        goto _exit_main;

```

```

169         0      clock_gettime(CLOCK_REALTIME, &t1);
170         .      .      .      .      .      .      .      .
171         7      1      1      2      1      1      1      1
172         .      .      .      .      .      .      .      .
173         .      .      .      .      .      .      .      .
174         .      .      .      .      .      .      .      .
175         .      .      .      .      .      .      .      .
176         12     1      1      5      2      2      1      0
177         .      .      .      .      .      .      .      .
178         .      .      .      .      .      .      .      .
179         6      1      1      3      0      0      0      0
180         .      .      .      .      .      .      .      .
181         .      .      .      .      .      .      .      .
182         .      .      .      .      .      .      .      .
183         .      .      .      .      .      .      .      .
184         3      1      1      0      0      0      0      0
185         .      .      .      .      .      .      .      .
186         .      .      .      .      .      .      .      .
187         .      .      .      .      .      .      .      .
188         .      .      .      .      .      .      .      .
189         .      .      .      .      .      .      .      .
190         .      .      .      .      .      .      .      .
191         .      .      .      .      .      .      .      .
192         .      .      .      .      .      .      .      .
193         6      1      1      2      0      0      0      0
        0      }

```

194									
195									
196									
197	11	2	2	0	0	0	6	0	
198									
199									
200									
201									
202	3	1	1	2	0	0	1	0	
203									
204	11	1	1	5	0	0	1	0	
205									
206	9	2	2	3	0	0	1	0	
207									
208	48	2	2	17	0	0	6	0	
209	240	1	1	85	0	0	30	0	
210	275	1	1	125	0	0	50	5	
211									
212									
213									
214									
215	2	0	0	0	0	0	1	0	
216	9	1	1	3	0	0	1	0	
217	83,886,088	2	2	31,457,282	0	0	10,485,761	0	
218	62,914,560	0	0	20,971,520	0	0	10,485,760	1,310,720	
219	1,310,720	6	1	1	3	2	2	0	0

```

220      .      .      .      .      .      .      .      .
221      .      .      .      .      .      .      .      .
222      .      .      .      .      .      .      .      .
223      17      2      2      6      0      0      2      0
224      .      .      .      .      .      .      .      .
225      48      1      1      17      0      0      6      0
226      305      2      2      120      0      0      30      0
227      275      1      1      150      7      7      25      0
228      1,525      3      3      600      0      0      150      0
229      3,000      2      2      1,625      14      14      125      0
230      275      1      1      150      0      0      25      0
231      .      .      .      .      .      .      .      .
232      1      0      0      1      0      0      0      0
233      6      1      1      2      0      0      0      0
234      .      .      .      .      .      .      .      .
235      .      .      .      .      .      .      .      .
236      .      .      .      .      .      .      .      .
237      18      3      3      0      0      0      8      0
238      .      .      .      .      .      .      .      .
239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      6      0      0      0      0      0      4      0
242      .      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .      .
244      14      1      1      6      1      0      2      1
245      6      0      0      2      0      0      0      0
246      .      .      .      .      .      .      .      .

```

```

247         perror("");
248         return NULL;
249     }
250     1,541     5     5     514     1     0     103     0
251     808     2     2     303     0     0     202     3
252     505     0     0     202     0     0     0     0
253         while (EOF != (c=fgetc(fp)) && c != '\n')
254         {
255             3             str[len++]=c;
256             0             if (len==tam-1) {
257                 str = realloc(str, tam *= 2);
258                 if (!str) {
259                     perror("");
260                     return NULL;
261                 }
262             }
263         }
264     8     1     1     2     0     0     0     0
265     7     0     0     0     if (c != EOF)
266     0             str[len++]='\n';
267     12     1     1     4     0     0     4     0
268     0             str[len++]='\0';
269     2     0     0     2     0     0     0     0
270     0             return str;
271     12     1     1     4     0     0     0     0
272     0     }
273
274     matrix_t*
275     create_matrix(size_t rows, size_t cols)
276     30     1     1     0     0     0     15     0
277     0     {
278         matrix_t * m;

```

```

273 . . . . . . . .
274 30 2 2 . 9 1 0 3 0
      0 if (!(m = malloc(sizeof(matrix_t)))) {
275 . . . . . . . .
      perror("");
276 . . . . . . . .
      return NULL;
277 . . . . . . . .
      }
278 . . . . . . . .
      .
279 9 1 1 . 6 0 0 3 0
      0 m->rows = rows;
280 9 1 1 . 6 0 0 3 0
      0 m->cols = cols;
281 51 2 2 . 21 0 0 3 0
      0 if (!(m->array = malloc(sizeof(double)
      *rows*cols))) {
282 . . . . . . . .
      free(m);
283 . . . . . . . .
      perror("");
284 . . . . . . . .
      return NULL;
285 . . . . . . . .
      }
286 . . . . . . . .
      .
287 3 1 1 . 3 0 0 0 0
      0 return m;
288 18 1 1 . 6 0 0 0 0
      0 }
289 . . . . . . . .
      .
290 . . . . . . . .
      void
291 . . . . . . . .
      destroy_matrix(matrix_t* m)
292 27 1 1 . 0 0 0 12 0
      0 {
293 9 0 0 . 3 0 0 0 0
      0 if (!m) return;
294 24 1 1 . 12 0 0 0 0
      0 free(m->array);
295 24 1 1 . 9 0 0 0 0
      0 free(m);
296 18 0 0 . 6 0 0 0 0
      0 }
297 . . . . . . . .
      .
298 . . . . . . . .
      int
299 . . . . . . . .

```

```

300      . print_matrix(FILE* fp, matrix_t* m)
301      10      2      2      0      0      0      5      0
302      0 {
303      .      .      .      .      .      .      .      .
304      .      .      .      .      .      .      .      .
305      .      .      .      .      .      .      .      .
306      .      .      .      .      .      .      .      .
307      .      .      .      .      .      .      .      .
308      48      2      2      17      0      0      6      0
309      240      2      2      85      0      0      30      0
310      550      3      3      250      0      0      0      0
311      0      if (fprintf(fp, "%lu", (unsigned long)
312      +j]) < 0) {
313      .      .      .      .      .      .      .      .
314      .      .      .      .      .      .      .      .
315      .      .      .      .      .      .      .      .
316      .      .      .      .      .      .      .      .
317      .      .      .      .      .      .      .      .
318      10      1      1      4      0      0      0      0
319      0      if (fprintf(fp, "\n") < 0) {
320      .      .      .      .      .      .      .      .
321      .      .      .      .      .      .      .      .
322      .      .      .      .      .      .      .      .
323      .      .      .      .      .      .      .      .
324      1      1      1      0      0      0      0      0
325      0      return 0;
326      6      1      1      2      0      0      0      0
327      0 }
328
329 -----
330 Ir      I1mr ILmr Dr      D1mr DLmr Dw      D1mw
331      DLmw
332
333 -----
334 146,813,063 120 119 52,434,158 36 30 20,972,616 1,310,741
335 1,310,739 events annotated

```

```

1 6 68 18 73 48 69 59 126 52 95 128 111 73 82 68 104 68 105 100
   134 76 81 150 129 82 150 77 118 115 171 110 145 53 74 146
   123 104
2 e et al.
3 ==666== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
   copyright info
4 ==666== Command: /tmp/02-mmult
5 ==666== Parent PID: 597
6 ==666==
7 --666-- Warning: Cannot auto-detect cache config, using
   defaults.
8 --666-- Run with -v to see.
9 ==666==
10 ==666== I refs: 147,225,941
11 ==666== I1 misses: 2,409
12 ==666== LLi misses: 2,389
13 ==666== I1 miss rate: 0.00%
14 ==666== LLi miss rate: 0.00%
15 ==666==
16 ==666== D refs: 73,565,863 (52,539,344 rd +
   21,026,519 wr)
17 ==666== D1 misses: 1,314,850 ( 3,527 rd +
   1,311,323 wr)
18 ==666== LLd misses: 1,314,203 ( 2,925 rd +
   1,311,278 wr)
19 ==666== D1 miss rate: 1.8% ( 0.0% +
   6.2% )
20 ==666== LLd miss rate: 1.8% ( 0.0% +
   6.2% )
21 ==666==
22 ==666== LL refs: 1,317,259 ( 5,936 rd +
   1,311,323 wr)
23 ==666== LL misses: 1,316,592 ( 5,314 rd +
   1,311,278 wr)
24 ==666== LL miss rate: 0.6% ( 0.0% +
   6.2% )
25 -----
26 I1 cache: 32768 B, 32 B, 4-way associative
27 D1 cache: 32768 B, 32 B, 4-way associative
28 LL cache: 524288 B, 32 B, 8-way associative
29 Command: /tmp/02-mmult
30 Data file: cachegrind.out.666
31 Events recorded: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
32 Events shown: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
33 Event sort order: Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw DLmw
34 Thresholds: 0.1 100 100 100 100 100 100 100 100
35 Include dirs:
36 User annotated: /root/CARPETA/tp2-2020-2q-src/main.c
37 Auto-annotation: off
38
39 -----
40 Ir I1mr I1Lmr Dr D1mr DLmr Dw D1mw

```


41	DLmw										
42	147,225,941 2,409 2,389 52,539,344 3,527 2,925 21,026,519										
43	1,311,323 1,311,278 PROGRAM TOTALS										
44											
45	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw			
46		DLmw		file:function							
47	146,810,542 29 29 52,433,589 31 31 20,972,246										
48	1,310,728 1,310,728 /root/CARPETA/tp2-2020-2q-src/main.c:										
49	matrix_multiply										
50	-- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c										
51											
52	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw			
53		DLmw									
54	-- line 18 -----										
55		
56			
57		
58		
59		
60		
61		
62		
63	10	2	2	0	0	0	5	0			
64		
65		
66		
67		
68		

```

        .      /* line buffer (init to null to
simplify freeing on error) */
69      1      0      0      0      0      0      1      0
        .      0      char *line = NULL;
70      .      .      .      .      .      .      .      .
        .      .      .      /* line parsing auxiliar pointers */
71      .      .      .      .      .      .      .      .
        .      .      .      char *nptr, *endptr;
72      .      .      .      .      .      .      .      .
        .      .      .      /* auxiliar variables */
73      .      .      .      .      .      .      .      .
        .      .      .      long l;
74      .      .      .      .      .      .      .      .
        .      .      .      double e;
75      2      1      1      0      0      0      1      0
        .      .      .      0      size_t lineno = 1;
76      .      .      .      .      .      .      .      .
        .      .      .      struct timespec t0;
77      .      .      .      .      .      .      .      .
        .      .      .      struct timespec t1;
78      .      .      .      .      .      .      .      .
        .      .      .      double dt;
79      .      .      .      .      .      .      .      .
        .      .      .      size_t i;
80      .      .      .      .      .      .      .      .
        .      .      .      .
81      25     4      4      9      0      0      1      0
        .      .      .      0      for(, !feof(stdin); lineno++) {
82      10     1      1      4      0      0      6      0
        .      .      .      0      a=b=c=NULL;
83      .      .      .      .      .      .      .      .
        .      .      .      .
84      18     3      3      8      1      0      2      0
        .      .      .      0      line = read_line(stdin);
85      .      .      .      .      .      .      .      .
        .      .      .      .
86      6      0      0      2      0      0      0      0
        .      .      .      0      if (!line) goto _exit_main;
87      9      0      0      4      0      0      0      0
        .      .      .      0      if (line[0] == 0) break;
88      .      .      .      .      .      .      .      .
        .      .      .      .
89      .      .      .      .      .      .      .      .
        .      .      .      .      /* parse dimension */
90      2      1      1      1      0      0      1      1
        .      .      .      0      nptr = line;
91      10     1      1      3      1      0      1      0
        .      .      .      0      l = strtol(nptr, &endptr, 10);
92      8      1      1      3      0      0      0      0
        .      .      .      0      if (errno) {
93      .      .      .      .      .      .      .      .
        .      .      .      .      perror("");
94      .      .      .      .      .      .      .      .
        .      .      .      .      goto _exit_main;

```

```

95         . . . } . . .
96     4 2 2 2 0 2 0 0 0 0
97         . . . if (nptr == endptr) {
98         . . .     fprintf(stderr, "missing
dimension");
99         . . .     goto _exit_main;
100        . . . }
101    3 2 2 1 0 0 0 0
102        . . . if (l < 1) {
103        . . .     fprintf(stderr, "invalid
dimension");
104        . . .     goto _exit_main;
105        . . . }
106    2 1 1 1 0 0 1 0
107        . . . 0 n = (size_t) l;
108        . . . #if 0
109        . . .     /* parse block size */
110        . . .     nptr = endptr;
111        . . .     l = strtol(nptr, &endptr, 10);
112        . . .     if (errno) {
113        . . .         perror("");
114        . . .         goto _exit_main;
115        . . .     }
116    -- line 75 -----
117    -- line 83 -----
118        . . . }
119        . . . bs = (size_t) l;
120        . . .
121        . . . if (n % bs) {
122        . . .     fprintf(stderr, "block size doesn
't match");
123        . . .     goto _exit_main;

```

```

122         .      .      .      .      .      .      .      .
123         .      .      .      #else      .      .      .      .
124         .      .      .      .      .      .      .      .
125         .      .      .      .      .      .      .      .
126         .      .      .      .      .      .      .      .
127         .      .      .      .      .      .      .      .
128         .      .      .      .      .      .      .      .
129         .      .      .      .      .      .      .      .
130         .      .      .      .      .      .      .      .
131         .      .      .      .      .      .      .      .
132         .      .      .      .      .      .      .      .
133         .      .      .      .      .      .      .      .
134         .      .      .      .      .      .      .      .
135         .      .      .      .      .      .      .      .
136         .      .      .      .      .      .      .      .
137         .      .      .      .      .      .      .      .
138         .      .      .      .      .      .      .      .
139         .      .      .      .      .      .      .      .
140         .      .      .      .      .      .      .      .
141         .      .      .      .      .      .      .      .
142         .      .      .      .      .      .      .      .
143         .      .      .      .      .      .      .      .
144         .      .      .      .      .      .      .      .
145         .      .      .      .      .      .      .      .
146         .      .      .      .      .      .      .      .
147         .      .      .      .      .      .      .      .

```

```

148      370      2      2      147      0      0      37      0
149      72      1      1      for (i=0; i < n*n; i++) {
150      324      1      1      36      0      0      36      0
151      288      1      1      nptr = endptr;
152      .      .      .      108      0      0      36      0
153      .      .      .      e = strtod(nptr, &endptr);
154      .      .      .      108      0      0      0      0
155      144      1      1      if (errno) {
156      .      .      .      .      .      .      .      .
157      .      .      .      perror("");
158      .      .      .      .      .      .      .      .
159      252      1      1      goto _exit_main;
160      .      .      .      .      .      .      .      .
161      .      .      .      }
162      8      1      1      72      0      0      0      0
163      .      .      .      if (nptr == endptr) {
164      .      .      .      .      .      .      .      .
165      13      2      2      fprintf(stderr, "missing B
166      .      .      .      matrix element");
167      .      .      .      .      .      .      .      .
168      8      1      1      goto _exit_main;
169      .      .      .      .      .      .      .      .
170      7      1      1      }
171      12      1      1      144      0      0      36      9
172      .      .      .      b->array[i] = e;
173      .      .      .      .      .      .      .      .
174      .      .      .      }
175      .      .      .      .      .      .      .      .
176      8      1      1      2      0      0      0      0
177      .      .      .      clock_gettime(CLOCK_REALTIME, &t0);
178      .      .      .      .      .      .      .      .
179      .      .      .      .      .      .      .      .
180      13      2      2      /* multiply matrixes */
181      .      .      .      6      1      1      1      0
182      .      .      .      if (!(c = matrix_multiply(a, b, bs))
183      .      .      .      )
184      .      .      .      .      .      .      .      .
185      .      .      .      goto _exit_main;
186      .      .      .      .      .      .      .      .
187      8      1      1      2      0      0      0      0
188      .      .      .      clock_gettime(CLOCK_REALTIME, &t1);
189      .      .      .      .      .      .      .      .
190      7      1      1      2      1      1      1      1
191      .      .      .      dt = (float) (t1.tv_sec - t0.tv_sec
192      .      .      .      );
193      12      1      1      5      2      2      1      0
194      .      .      .      dt = dt + ((float)(t1.tv_nsec - t0.
195      .      .      .      tv_nsec)) / 1.0e9;
196      .      .      .      .      .      .      .      .
197      .      .      .      .      .      .      .      .

```

173		13	3	2	5	2	2	0	0
				0	if(print_matrix(stdout, c) == -1)				0
174
175	goto _exit_main;
				.					
176		12	2	1	7	0	0	0	0
				0	fprintf(stderr, "time: %g\n", dt);				
177
				.					
178		6	1	1	3	0	0	0	0
				0	free(line);				
179		6	1	1	3	0	0	0	0
				0	destroy_matrix(a);				
180		6	1	1	3	0	0	0	0
				0	destroy_matrix(b);				
181		6	0	0	3	0	0	0	0
				0	destroy_matrix(c);				
182
				.	}				
183
				.					
184		3	1	1	0	0	0	0	0
				0	return 0;				
185
				.					
186
				.	_exit_main:				
187
				.	fprintf(stderr, " at line %u\n", (
				unsigned) lineno);					
188
				.	free(line);				
189
				.	destroy_matrix(a);				
190
				.	destroy_matrix(b);				
191
				.	destroy_matrix(c);				
192
				.	exit(1);				
193		6	1	1	2	0	0	0	0
				0 }					
194
				.					
195
				.	matrix_t*				
196
				.	matrix_multiply(matrix_t* m1, matrix_t*				
				m2, int bs)					
197		11	2	2	0	0	0	6	0
				0 {					
198
				.	size_t n, en, i, j, k, kk, jj;				

```

199      .      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .      .
201      .      .      .      .      .      .      .      .
202      3      1      1      2      0      0      1      0
203      .      .      .      .      .      .      .      .
204      11     1      1      5      0      0      1      0
205      .      .      .      .      .      .      .      .
206      9      2      2      3      0      0      1      0
207      .      .      .      .      .      .      .      .
208      56     2      2      20     0      0      7      0
209      336    1      1      120    0      0      42     0
210      396    1      1      180    0      0      72     8
211      .      .      .      .      .      .      .      .
212      .      .      .      .      .      .      .      .
213      .      .      .      .      .      .      .      .
214      .      .      .      .      .      .      .      .
215      2      0      0      0      0      0      1      0
216      9      1      1      3      0      0      1      0
217      83,886,088 2      2 31,457,282 0      0 10,485,761 0
218      62,914,560 0      0 20,971,520 0      0 10,485,760 1,310,720
219      1,310,720 6      1      3      2      2      0      0
220      .      .      .      .      .      .      .      .
221      .      .      .      .      .      .      .      .
222      .      .      .      .      .      .      .      .
223      17     2      2      6      0      0      2      0
224      17     2      2      6      0      0      2      0
225      56     1      1      20     0      0      7      0

```

```

double sum;

matrix_t* mr;

n = m1->rows;

if(!(mr = create_matrix(n,n))) return
NULL;

en = bs*(n/bs);

for(i=0; i<n; i++)
for(j=0; j<n; j++)
mr->array[i*n+j] = 0.0;

#if 1
if (1) {
size_t j;
size_t dim = 1024*1024*10;
int *v = malloc(dim*sizeof(int));
for (j = 0; j < dim; ++j)
v[j] = -1;
free(v);
}
#endif

for(kk=0; kk<en; kk+=bs)
for(jj=0; jj<en; jj+=bs)

```

```

226         0         for(i=0; i<n; i++)
226         426      2      2         168      0      0         42         0
227         0         for(j=jj; j<jj+bs; j++) {
227         396      1      1         216      10      10         36         0
228         0         sum = mr->array[i*n+j];
228         2,556    3      3         1,008    0      0         252         0
229         0         for(k=kk; k<kk+bs; k++)
229         5,184    2      2         2,808    19      19         216         0
229         0         sum += m1->array[i*n+k] * m2->
array[k*n+j];
230         396      1      1         216      0      0         36         0
230         0         mr->array[i*n+j] = sum;
231         .         .         .         .         .         .         .         .
231         .         .         .         .         .         .         .         .
232         1      0      0         1      0      0         0         0
232         0         return mr;
233         6      1      1         2      0      0         0         0
233         0      }
234         .         .         .         .         .         .         .         .
235         .         .         .         .         .         .         .         .
236         .         .         .         .         .         .         .         .
237         18      3      3         0      0      0         8         0
237         0      {
238         .         .         .         .         .         .         .         .
239         .         .         .         .         .         .         .         .
240         .         .         .         .         .         .         .         .
241         6      0      0         0      0      0         4         0
241         0         size_t len = 0, tam = DEF_LINE_SZ;
242         .         .         .         .         .         .         .         .
243         .         .         .         .         .         .         .         .
244         14      1      1         6      1      0         2         1
244         0         str = malloc(tam);
245         6      0      0         2      0      0         0         0
245         0         if (!str) {
246         .         .         .         .         .         .         .         .
247         .         .         .         .         .         .         .         .
248         .         .         .         .         .         .         .         .
249         .         .         .         .         .         .         .         .
250         2,201    5      5         734      1      0         147         0
250         0         while (EOF != (c=fgetc(fp)) && c != '\n')
251         {
251         1,160    2      2         435      0      0         290         4

```



```

252         4      str[len++] = c;
253         725    0 0      290 0 0      0 0
254         0      if (len == tam - 1) {
255         .      .      .      .      .      .
256         .      .      .      str = realloc(str, tam * 2);
257         .      .      .      if (!str) {
258         .      .      .      perror("");
259         .      .      .      return NULL;
260         .      .      .      }
261         .      .      .      }
262         .      .      .      }
263         .      .      .      }
264         8 1 1      2 0 0      0 0
265         0      if (c != EOF)
266         7 0 0      2 0 0      2 0
267         0      str[len++] = '\n';
268         .      .      .      .      .      .
269         12 1 1      4 0 0      4 0
270         0      str[len++] = '\0';
271         2 0 0      2 0 0      0 0
272         0      return str;
273         12 1 1      4 0 0      0 0
274         0 }
275         .      .      .      .      .      .
276         .      .      .      .      .      .
277         .      .      .      .      .      .
278         .      .      .      .      .      .
279         .      .      .      .      .      .
280         .      .      .      .      .      .
281         .      .      .      .      .      .
282         .      .      .      .      .      .
283         .      .      .      .      .      .
284         .      .      .      .      .      .
285         .      .      .      .      .      .
286         .      .      .      .      .      .
287         .      .      .      .      .      .
288         .      .      .      .      .      .
289         .      .      .      .      .      .
290         .      .      .      .      .      .
291         .      .      .      .      .      .
292         .      .      .      .      .      .
293         .      .      .      .      .      .
294         .      .      .      .      .      .
295         .      .      .      .      .      .
296         .      .      .      .      .      .
297         .      .      .      .      .      .
298         .      .      .      .      .      .
299         .      .      .      .      .      .
300         .      .      .      .      .      .
301         .      .      .      .      .      .
302         .      .      .      .      .      .
303         .      .      .      .      .      .
304         .      .      .      .      .      .
305         .      .      .      .      .      .
306         .      .      .      .      .      .
307         .      .      .      .      .      .
308         .      .      .      .      .      .
309         .      .      .      .      .      .
310         .      .      .      .      .      .
311         .      .      .      .      .      .
312         .      .      .      .      .      .
313         .      .      .      .      .      .
314         .      .      .      .      .      .
315         .      .      .      .      .      .
316         .      .      .      .      .      .
317         .      .      .      .      .      .
318         .      .      .      .      .      .
319         .      .      .      .      .      .
320         .      .      .      .      .      .
321         .      .      .      .      .      .
322         .      .      .      .      .      .
323         .      .      .      .      .      .
324         .      .      .      .      .      .
325         .      .      .      .      .      .
326         .      .      .      .      .      .
327         .      .      .      .      .      .
328         .      .      .      .      .      .
329         .      .      .      .      .      .
330         .      .      .      .      .      .
331         .      .      .      .      .      .
332         .      .      .      .      .      .
333         .      .      .      .      .      .
334         .      .      .      .      .      .
335         .      .      .      .      .      .
336         .      .      .      .      .      .
337         .      .      .      .      .      .
338         .      .      .      .      .      .
339         .      .      .      .      .      .
340         .      .      .      .      .      .
341         .      .      .      .      .      .
342         .      .      .      .      .      .
343         .      .      .      .      .      .
344         .      .      .      .      .      .
345         .      .      .      .      .      .
346         .      .      .      .      .      .
347         .      .      .      .      .      .
348         .      .      .      .      .      .
349         .      .      .      .      .      .
350         .      .      .      .      .      .
351         .      .      .      .      .      .
352         .      .      .      .      .      .
353         .      .      .      .      .      .
354         .      .      .      .      .      .
355         .      .      .      .      .      .
356         .      .      .      .      .      .
357         .      .      .      .      .      .
358         .      .      .      .      .      .
359         .      .      .      .      .      .
360         .      .      .      .      .      .
361         .      .      .      .      .      .
362         .      .      .      .      .      .
363         .      .      .      .      .      .
364         .      .      .      .      .      .
365         .      .      .      .      .      .
366         .      .      .      .      .      .
367         .      .      .      .      .      .
368         .      .      .      .      .      .
369         .      .      .      .      .      .
370         .      .      .      .      .      .
371         .      .      .      .      .      .
372         .      .      .      .      .      .
373         .      .      .      .      .      .
374         .      .      .      .      .      .
375         .      .      .      .      .      .
376         .      .      .      .      .      .
377         .      .      .      .      .      .
378         .      .      .      .      .      .
379         .      .      .      .      .      .
380         .      .      .      .      .      .
381         .      .      .      .      .      .
382         .      .      .      .      .      .
383         .      .      .      .      .      .
384         .      .      .      .      .      .
385         .      .      .      .      .      .
386         .      .      .      .      .      .
387         .      .      .      .      .      .
388         .      .      .      .      .      .
389         .      .      .      .      .      .
390         .      .      .      .      .      .
391         .      .      .      .      .      .
392         .      .      .      .      .      .
393         .      .      .      .      .      .
394         .      .      .      .      .      .
395         .      .      .      .      .      .
396         .      .      .      .      .      .
397         .      .      .      .      .      .
398         .      .      .      .      .      .
399         .      .      .      .      .      .
400         .      .      .      .      .      .
401         .      .      .      .      .      .
402         .      .      .      .      .      .
403         .      .      .      .      .      .
404         .      .      .      .      .      .
405         .      .      .      .      .      .
406         .      .      .      .      .      .
407         .      .      .      .      .      .
408         .      .      .      .      .      .
409         .      .      .      .      .      .
410         .      .      .      .      .      .
411         .      .      .      .      .      .
412         .      .      .      .      .      .
413         .      .      .      .      .      .
414         .      .      .      .      .      .
415         .      .      .      .      .      .
416         .      .      .      .      .      .
417         .      .      .      .      .      .
418         .      .      .      .      .      .
419         .      .      .      .      .      .
420         .      .      .      .      .      .
421         .      .      .      .      .      .
422         .      .      .      .      .      .
423         .      .      .      .      .      .
424         .      .      .      .      .      .
425         .      .      .      .      .      .
426         .      .      .      .      .      .
427         .      .      .      .      .      .
428         .      .      .      .      .      .
429         .      .      .      .      .      .
430         .      .      .      .      .      .
431         .      .      .      .      .      .
432         .      .      .      .      .      .
433         .      .      .      .      .      .
434         .      .      .      .      .      .
435         .      .      .      .      .      .
436         .      .      .      .      .      .
437         .      .      .      .      .      .
438         .      .      .      .      .      .
439         .      .      .      .      .      .
440         .      .      .      .      .      .
441         .      .      .      .      .      .
442         .      .      .      .      .      .
443         .      .      .      .      .      .
444         .      .      .      .      .      .
445         .      .      .      .      .      .
446         .      .      .      .      .      .
447         .      .      .      .      .      .
448         .      .      .      .      .      .
449         .      .      .      .      .      .
450         .      .      .      .      .      .
451         .      .      .      .      .      .
452         .      .      .      .      .      .
453         .      .      .      .      .      .
454         .      .      .      .      .      .
455         .      .      .      .      .      .
456         .      .      .      .      .      .
457         .      .      .      .      .      .
458         .      .      .      .      .      .
459         .      .      .      .      .      .
460         .      .      .      .      .      .
461         .      .      .      .      .      .
462         .      .      .      .      .      .
463         .      .      .      .      .      .
464         .      .      .      .      .      .
465         .      .      .      .      .      .
466         .      .      .      .      .      .
467         .      .      .      .      .      .
468         .      .      .      .      .      .
469         .      .      .      .      .      .
470         .      .      .      .      .      .
471         .      .      .      .      .      .
472         .      .      .      .      .      .
473         .      .      .      .      .      .
474         .      .      .      .      .      .
475         .      .      .      .      .      .
476         .      .      .      .      .      .
477         .      .      .      .      .      .
478         .      .      .      .      .      .
479         .      .      .      .      .      .
480         .      .      .      .      .      .
481         .      .      .      .      .      .
482         .      .      .      .      .      .
483         .      .      .      .      .      .
484         .      .      .      .      .      .
485         .      .      .      .      .      .
486         .      .      .      .      .      .
487         .      .      .      .      .      .
488         .      .      .      .      .      .
489         .      .      .      .      .      .
490         .      .      .      .      .      .
491         .      .      .      .      .      .
492         .      .      .      .      .      .
493         .      .      .      .      .      .
494         .      .      .      .      .      .
495         .      .      .      .      .      .
496         .      .      .      .      .      .
497         .      .      .      .      .      .
498         .      .      .      .      .      .
499         .      .      .      .      .      .
500         .      .      .      .      .      .
501         .      .      .      .      .      .
502         .      .      .      .      .      .
503         .      .      .      .      .      .
504         .      .      .      .      .      .
505         .      .      .      .      .      .
506         .      .      .      .      .      .
507         .      .      .      .      .      .
508         .      .      .      .      .      .
509         .      .      .      .      .      .
510         .      .      .      .      .      .
511         .      .      .      .      .      .
512         .      .      .      .      .      .
513         .      .      .      .      .      .
514         .      .      .      .      .      .
515         .      .      .      .      .      .
516         .      .      .      .      .      .
517         .      .      .      .      .      .
518         .      .      .      .      .      .
519         .      .      .      .      .      .
520         .      .      .      .      .      .
521         .      .      .      .      .      .
522         .      .      .      .      .      .
523         .      .      .      .      .      .
524         .      .      .      .      .      .
525         .      .      .      .      .      .
526         .      .      .      .      .      .
527         .      .      .      .      .      .
528         .      .      .      .      .      .
529         .      .      .      .      .      .
530         .      .      .      .      .      .
531         .      .      .      .      .      .
532         .      .      .      .      .      .
533         .      .      .      .      .      .
534         .      .      .      .      .      .
535         .      .      .      .      .      .
536         .      .      .      .      .      .
537         .      .      .      .      .      .
538         .      .      .      .      .      .
539         .      .      .      .      .      .
540         .      .      .      .      .      .
541         .      .      .      .      .      .
542         .      .      .      .      .      .
543         .      .      .      .      .      .
544         .      .      .      .      .      .
545         .      .      .      .      .      .
546         .      .      .      .      .      .
547         .      .      .      .      .      .
548         .      .      .      .      .      .
549         .      .      .      .      .      .
550         .      .      .      .      .      .
551         .      .      .      .      .      .
552         .      .      .      .      .      .
553         .      .      .      .      .      .
554         .      .      .      .      .      .
555         .      .      .      .      .      .
556         .      .      .      .      .      .
557         .      .      .      .      .      .
558         .      .      .      .      .      .
559         .      .      .      .      .      .
560         .      .      .      .      .      .
561         .      .      .      .      .      .
562         .      .      .      .      .      .
563         .      .      .      .      .      .
564         .      .      .      .      .      .
565         .      .      .      .      .      .
566         .      .      .      .      .      .
567         .      .      .      .      .      .
568         .      .      .      .      .      .
569         .      .      .      .      .      .
570         .      .      .      .      .      .
571         .      .      .      .      .      .
572         .      .      .      .      .      .
573         .      .      .      .      .      .
574         .      .      .      .      .      .
575         .      .      .      .      .      .
576         .      .      .      .      .      .
577         .      .      .      .      .      .
578         .      .      .      .      .      .
579         .      .      .      .      .      .
580         .      .      .      .      .      .
581         .      .      .      .      .      .
582         .      .      .      .      .      .
583         .      .      .      .      .      .
584         .      .      .      .      .      .
585         .      .      .      .      .      .
586         .      .      .      .      .      .
587         .      .      .      .      .      .
588         .      .      .      .      .      .
589         .      .      .      .      .      .
590         .      .      .      .      .      .
591         .      .      .      .      .      .
592         .      .      .      .      .      .
593         .      .      .      .      .      .
594         .      .      .      .      .      .
595         .      .      .      .      .      .
596         .      .      .      .      .      .
597         .      .      .      .      .      .
598         .      .      .      .      .      .
599         .      .      .      .      .      .
600         .      .      .      .      .      .
601         .      .      .      .      .      .
602         .      .      .      .      .      .
603         .      .      .      .      .      .
604         .      .      .      .      .      .
605         .      .      .      .      .      .
606         .      .      .      .      .      .
607         .      .      .      .      .      .
608         .      .      .      .      .      .
609         .      .      .      .      .      .
610         .      .      .      .      .      .
611         .      .      .      .      .      .
612         .      .      .      .      .      .
613         .      .      .      .      .      .
614         .      .      .      .      .      .
615         .      .      .      .      .      .
616         .      .      .      .      .      .
617         .      .      .      .      .      .
618         .      .      .      .      .      .
619         .      .      .      .      .      .
620         .      .      .      .      .      .
621         .      .      .      .      .      .
622         .      .      .      .      .      .
623         .      .      .      .      .      .
624         .      .      .      .      .      .
625         .      .      .      .      .      .
626         .      .      .      .      .      .
627         .      .      .      .      .      .
628         .      .      .      .      .      .
629         .      .      .      .      .      .
630         .      .      .      .      .      .
631         .      .      .      .      .      .
632         .      .      .      .      .      .
633         .      .      .      .      .      .
634         .      .      .      .      .      .
635         .      .      .      .      .      .
636         .      .      .      .      .      .
637         .      .      .      .      .      .
638         .      .      .      .      .      .
639         .      .      .      .      .      .
640         .      .      .      .      .      .
641         .      .      .      .      .      .
642         .      .      .      .      .      .
643         .      .      .      .      .      .
644         .      .      .      .      .      .
645         .      .      .      .      .      .
646         .      .      .      .      .      .
647         .      .      .      .      .      .
648         .      .      .      .      .      .
649         .      .      .      .      .      .
650         .      .      .      .      .      .
651         .      .      .      .      .      .
652         .      .      .      .      .      .
653         .      .      .      .      .      .
654         .      .      .      .      .      .
655         .      .      .      .      .      .
656         .      .      .      .      .      .
657         .      .      .      .      .      .
658         .      .      .      .      .      .
659         .      .      .      .      .      .
660         .      .      .      .      .      .
661         .      .      .      .      .      .
662         .      .      .      .      .      .
663         .      .      .      .      .      .
664         .      .      .      .      .      .
665         .      .      .      .      .      .
666         .      .      .      .      .      .
667         .      .      .      .      .      .
668         .      .      .      .      .      .
669         .      .      .      .      .      .
670         .      .      .      .      .      .
671         .      .      .      .      .      .
672         .      .      .      .      .      .
673         .      .      .      .      .      .
674         .      .      .      .      .      .
675         .      .      .      .      .      .
676         .      .      .      .      .      .
677         .      .      .      .      .      .
678         .      .      .      .      .      .
679         .      .      .      .      .      .
680         .      .      .      .      .      .
681         .      .      .      .      .      .
682         .      .      .      .      .      .
683         .      .      .      .      .      .
684         .      .      .      .      .      .
685         .      .      .      .      .      .
686         .      .      .      .      .      .
687         .      .      .      .      .      .
688         .      .      .      .      .      .
689         .      .      .      .      .      .
690         .      .      .      .      .      .
691         .      .      .      .      .      .
692         .      .      .      .      .      .
693         .      .      .      .      .      .
694         .      .      .      .      .      .
695         .      .      .      .      .      .
696         .      .      .      .      .      .
697         .      .      .      .      .      .
698         .      .      .      .      .      .
699         .      .      .      .      .      .
700         .      .      .      .      .      .
701         .      .      .      .      .      .
702         .      .      .      .      .      .
703         .      .      .      .      .      .
704         .      .      .      .      .      .
705         .      .      .      .      .      .
706         .      .      .      .      .      .
707         .      .      .      .      .      .
708         .      .      .      .      .      .
709         .      .      .      .      .      .
710         .      .      .      .      .      .
711         .      .      .      .      .      .
712         .      .      .      .      .      .
713         .      .      .      .      .      .
714         .      .      .      .      .      .
715         .      .      .      .      .      .
716         .      .      .      .      .      .
717         .      .      .      .      .      .
718         .      .      .      .      .      .
719         .      .      .      .      .      .
720         .      .      .      .      .      .
721         .      .      .      .      .      .
722         .      .      .      .      .      .
723         .      .      .      .      .      .
724         .      .      .      .      .      .
725         .      .      .      .      .      .
726         .      .      .      .      .      .
727         .      .      .      .      .      .
728         .      .      .      .      .      .
729         .      .      .      .      .      .
730         .      .      .      .      .      .
731         .      .      .      .      .      .
732         .      .      .      .      .      .
733         .      .      .      .      .      .
734         .      .      .      .      .      .
735         .      .      .      .      .      .
736         .      .      .      .      .      .
737         .      .      .      .      .      .
738         .      .      .      .      .      .
739         .      .      .      .      .      .
740         .      .      .      .      .      .
741         .      .      .      .      .      .
742         .      .      .      .      .      .
743         .      .      .      .      .      .
744         .      .      .      .      .      .
745         .      .      .      .      .      .
746         .      .      .      .      .      .
747         .      .      .      .      .      .
748         .      .      .      .      .      .
749         .      .      .      .      .      .
750         .      .      .      .      .      .
751         .      .      .      .      .      .
752         .      .      .      .      .      .
753         .      .      .      .      .      .
754         .      .      .      .      .      .
755         .      .      .      .      .      .
756         .      .      .      .      .      .
757         .      .      .      .      .      .
758         .      .      .      .      .      .
759         .      .      .      .      .      .
760         .      .      .      .      .      .
761         .      .      .      .      .      .
762         .      .      .      .      .      .
763         .      .      .      .      .      .
764         .      .      .      .      .      .
765         .      .      .      .      .      .
766         .      .      .      .      .      .
767         .      .      .      .      .      .
768         .      .      .      .      .      .
769         .      .      .      .      .      .
770         .      .      .      .      .      .
771         .      .      .      .      .      .
772         .      .      .      .      .      .
773         .      .      .      .      .      .
774         .      .      .      .      .      .
775         .      .      .      .      .      .
776         .      .      .      .      .      .
777         .      .      .      .      .      .
778         .      .      .      .      .      .
779         .      .      .      .      .      .
780         .      .      .      .      .      .
781         .      .      .      .      .      .
782         .      .      .      .      .      .
783         .      .      .      .      .      .
784         .      .      .      .      .      .
785         .      .      .      .      .      .
786         .      .      .      .      .      .
787         .      .      .      .      .      .
788         .      .      .      .      .      .
789         .      .      .      .      .      .
790         .      .      .      .      .      .
791         .      .      .      .      .      .
792         .      .      .      .      .      .
793         .      .      .      .      .      .
794         .      .      .      .      .      .
795         .      .      .      .      .      .
796         .      .      .      .      .      .
797         .      .      .      .      .      .
798         .      .      .      .      .      .
799         .      .      .      .      .      .
800         .      .      .      .      .      .
801         .      .      .      .      .      .
802         .      .      .      .      .      .
803         .      .      .      .      .      .
804         .      .      .      .      .      .
805         .      .      .      .      .      .
806         .      .      .      .      .      .
807         .      .      .      .      .      .
808         .      .      .      .      .      .
809         .      .      .      .      .      .
810         .      .      .      .      .      .
811         .      .      .      .      .      .
812         .      .      .      .      .      .
813         .      .      .      .      .      .
814         .      .      .      .      .      .
815         .      .      .      .      .      .
816         .      .      .      .      .      .
817         .      .      .      .      .      .
818         .      .      .      .      .      .
819         .      .      .      .      .      .
820         .      .      .      .      .      .
821         .      .      .      .      .      .
822         .      .      .      .      .      .
823         .      .      .      .      .      .
824         .      .      .      .      .      .
825         .      .      .      .      .      .
826         .      .      .      .      .      .
827         .      .      .      .      .      .
828         .      .      .      .      .      .
829         .      .      .      .      .      .
830         .      .      .      .      .      .
831         .      .      .      .      .      .
832         .      .      .      .      .      .
833         .      .      .      .      .      .
834         .      .      .      .      .      .
835         .      .      .      .      .      .
836         .      .      .      .      .      .
837         .      .      .      .      .      .
838         .      .      .      .      .      .
839         .      .      .      .      .      .
840         .      .      .      .      .      .
841         .      .      .      .      .      .
842         .      .      .      .      .      .
843         .      .      .      .      .      .
844         .      .      .      .      .      .
845         .      .      .      .      .      .
846         .      .      .      .      .      .
847         .      .      .      .      .      .
848         .      .      .      .      .      .
849         .      .      .      .      .      .
850         .      .      .      .      .      .
851         .      .      .      .      .      .
852         .      .      .      .      .      .
853         .      .      .      .      .      .
854         .      .      .      .      .      .
855         .      .      .      .      .      .
856         .      .      .      .      .      .
857         .      .      .      .      .      .
858         .      .      .      .      .      .
859         .      .      .      .      .      .
860         .      .      .      .      .      .
861         .      .      .      .      .      .
862         .      .      .      .      .      .
863         .      .      .      .      .      .
864         .      .      .      .      .      .
865         .      .      .      .      .      .
866         .      .      .      .      .      .
867         .      .      .      .      .      .
868         .      .      .      .      .      .
869         .      .      .      .      .      .
870         .      .      .      .      .      .
871         .      .      .      .      .      .
872         .      .      .      .      .      .
873         .      .      .      .      .      .
874         .      .      .      .      .      .
875         .      .      .      .      .      .
876         .      .      .      .      .      .
877         .      .      .      .      .      .
878         .      .      .      .      .      .
879         .      .      .      .      .      .
880         .      .      .      .      .      .
881         .      .      .      .      .      .
882         .      .      .      .      .      .
883         .      .      .      .      .      .
884         .      .      .      .      .      .
885         .      .      .      .      .      .
886         .      .      .      .      .      .
887         .      .      .      .      .      .
888         .      .      .      .      .      .
889         .      .      .      .      .      .
890         .      .      .      .      .      .
891         .      .      .      .      .      .
892         .      .      .      .      .      .
893         .      .      .      .      .      .
894         .      .      .      .      .      .
895         .      .      .      .      .      .
896         .      .      .      .      .      .
897         .      .      .      .      .      .
898         .      .      .      .      .      .
899         .      .      .      .      .      .
900         .      .      .      .      .      .
901         .      .      .      .      .      .
902         .      .      .      .      .      .
903         .      .      .      .      .      .
904         .      .      .      .      .      .
905         .      .      .      .      .      .
906         .      .      .      .      .      .
907         .      .      .      .      .      .
908         .      .      .      .      .      .
909         .      .      .      .      .      .
910         .      .      .      .      .      .
911         .      .      .      .      .      .
912         .      .      .      .      .      .
913         .      .      .      .      .      .
914         .      .      .      .      .      .
915         .      .      .      .      .      .
916         .      .      .      .      .      .
917         .      .      .      .      .      .
918         .      .      .      .      .      .
919         .      .      .      .      .      .
920         .      .      .      .      .      .
921         .      .      .      .      .      .
922         .      .      .      .      .      .
923         .      .      .      .      .      .
924         .      .      .      .      .      .
925         .      .      .      .      .      .
926         .      .      .      .      .      .
927         .      .      .      .      .      .
928         .      .      .      .      .      .
929         .      .      .      .      .      .
930         .      .      .      .      .      .
931         .      .      .      .      .      .
932         .      .      .      .      .      .
933         .      .      .      .      .      .
934         .      .      .      .      .      .
935         .      .      .      .      .      .
936         .      .      .      .      .      .
937         .      .      .      .      .      .
938         .      .      .      .      .      .
939         .      .      .      .      .      .
940         .      .      .      .      .      .
941         .      .      .      .      .      .
942         .      .      .      .      .      .
943         .      .      .      .      .      .
944         .      .      .      .      .      .
945         .      .      .      .      .      .
946         .      .      .      .      .      .
947         .      .      .      .      .      .
948         .      .      .      .      .      .
949         .      .      .      .      .      .
950         .      .      .      .      .      .
951         .      .      .      .      .      .
952         .      .      .      .      .      .
953         .      .      .      .      .      .
954         .      .      .      .      .      .
955         .      .      .      .      .      .
956         .      .      .      .      .      .
957         .      .      .      .      .      .
958         .      .      .      .      .      .
959         .      .      .      .      .      .
960         .      .      .      .      .      .
961         .      .      .      .      .      .
962         .      .      .      .      .      .
963         .      .      .      .      .      .
964         .      .      .      .      .      .
965         .      .      .      .      .      .
966         .      .      .      .      .      .
967         .      .      .      .      .      .
968         .      .      .      .      .      .
969         .      .      .      .      .      .
970         .      .      .      .      .      .
971         .      .      .      .      .      .
972         .      .      .      .      .      .
973         .      .      .      .      .      .
974         .      .      .      .      .      .
975         .      .      .      .      .      .
976         .      .      .      .      .      .
977         .      .      .      .      .      .
978         .      .      .      .      .      .
979         .      .      .      .      .      .
980         .      .      .      .      .      .
981         .      .      .      .      .      .
982         .      .      .      .      .      .
983         .      .      .      .      .      .
984         .      .      .      .      .      .
985         .      .      .      .      .      .
986         .      .      .      .      .      .
987         .      .      .      .      .      .
988         .      .      .      .      .      .
989         .      .      .      .      .      .
990         .      .      .      .      .      .
991         .      .      .      .      .      .
992         .      .      .      .      .      .
993         .      .      .      .      .      .
994         .      .      .      .      .      .
995         .      .      .      .      .      .
996         .      .      .      .      .      .
997         .      .      .      .      .      .
998         .      .      .      .      .      .
999         .      .      .      .      .      .
1000        .      .      .      .      .      .

```

```

279      9      1      1      .      6      0      0      3      0
      0      m->rows = rows;
280      9      1      1      .      6      0      0      3      0
      0      m->cols = cols;
281      51     2      2      .      21     0      0      3      0
      0      if (!(m->array = malloc(sizeof(double)
      *rows*cols))) {
282      .      .      .      .      .      .      .      .      .
      .      .      .      .      free(m);
283      .      .      .      .      .      .      .      .      .
      .      .      .      .      perror("");
284      .      .      .      .      .      .      .      .      .
      .      .      .      .      return NULL;
285      .      .      .      .      .      .      .      .      .
      .      .      .      .      }
286      .      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .      .
287      3      1      1      .      3      0      0      0      0
      0      return m;
288      18     1      1      .      6      0      0      0      0
      0 }
289      .      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .      .
290      .      .      .      .      .      .      .      .      .
      .      .      .      .      void
291      .      .      .      .      .      .      .      .      .
      .      .      .      .      destroy_matrix(matrix_t* m)
292      27     1      1      .      0      0      0      12     0
      0 {
293      9      0      0      .      3      0      0      0      0
      0      if (!m) return;
294      24     1      1      .      12     0      0      0      0
      0      free(m->array);
295      24     1      1      .      9      0      0      0      0
      0      free(m);
296      18     0      0      .      6      0      0      0      0
      0 }
297      .      .      .      .      .      .      .      .      .
      .      .      .      .      .      .      .      .      .
298      .      .      .      .      .      .      .      .      .
      .      .      .      .      int
299      .      .      .      .      .      .      .      .      .
      .      .      .      .      print_matrix(FILE* fp, matrix_t* m)
300      10     2      2      .      0      0      0      5      0
      0 {
301      .      .      .      .      .      .      .      .      .
      .      .      .      .      size_t i, j;
302      .      .      .      .      .      .      .      .      .
      .      .      .      .      size_t n;
303      3      1      1      .      2      0      0      1      0
      0      n = m->rows;
304      13     1      1      .      6      1      1      0      0
      0      if (fprintf(fp, "%lu", (unsigned long)

```

```

305         m->rows) < 0) {
306             . . . . .
307             . . . . .
308             . . . . .
309             . . . . .
310             . . . . .
311             . . . . .
312             . . . . .
313             . . . . .
314             . . . . .
315             . . . . .
316             . . . . .
317             . . . . .
318             . . . . .
319             . . . . .
320             . . . . .
321             . . . . .
322             . . . . .
323             . . . . .
324             . . . . .
325             . . . . .
326             . . . . .
327             . . . . .
328             . . . . .
329             . . . . .
330             . . . . .
331             . . . . .
332             . . . . .
333             . . . . .
334             . . . . .
335             . . . . .
336             . . . . .
337             . . . . .
338             . . . . .
339             . . . . .
340             . . . . .
341             . . . . .
342             . . . . .
343             . . . . .
344             . . . . .
345             . . . . .
346             . . . . .
347             . . . . .
348             . . . . .
349             . . . . .
350             . . . . .
351             . . . . .
352             . . . . .
353             . . . . .
354             . . . . .
355             . . . . .
356             . . . . .
357             . . . . .
358             . . . . .
359             . . . . .
360             . . . . .
361             . . . . .
362             . . . . .
363             . . . . .
364             . . . . .
365             . . . . .
366             . . . . .
367             . . . . .
368             . . . . .
369             . . . . .
370             . . . . .
371             . . . . .
372             . . . . .
373             . . . . .
374             . . . . .
375             . . . . .
376             . . . . .
377             . . . . .
378             . . . . .
379             . . . . .
380             . . . . .
381             . . . . .
382             . . . . .
383             . . . . .
384             . . . . .
385             . . . . .
386             . . . . .
387             . . . . .
388             . . . . .
389             . . . . .
390             . . . . .
391             . . . . .
392             . . . . .
393             . . . . .
394             . . . . .
395             . . . . .
396             . . . . .
397             . . . . .
398             . . . . .
399             . . . . .
400             . . . . .
401             . . . . .
402             . . . . .
403             . . . . .
404             . . . . .
405             . . . . .
406             . . . . .
407             . . . . .
408             . . . . .
409             . . . . .
410             . . . . .
411             . . . . .
412             . . . . .
413             . . . . .
414             . . . . .
415             . . . . .
416             . . . . .
417             . . . . .
418             . . . . .
419             . . . . .
420             . . . . .
421             . . . . .
422             . . . . .
423             . . . . .
424             . . . . .
425             . . . . .
426             . . . . .
427             . . . . .
428             . . . . .
429             . . . . .
430             . . . . .
431             . . . . .
432             . . . . .
433             . . . . .
434             . . . . .
435             . . . . .
436             . . . . .
437             . . . . .
438             . . . . .
439             . . . . .
440             . . . . .
441             . . . . .
442             . . . . .
443             . . . . .
444             . . . . .
445             . . . . .
446             . . . . .
447             . . . . .
448             . . . . .
449             . . . . .
450             . . . . .
451             . . . . .
452             . . . . .
453             . . . . .
454             . . . . .
455             . . . . .
456             . . . . .
457             . . . . .
458             . . . . .
459             . . . . .
460             . . . . .
461             . . . . .
462             . . . . .
463             . . . . .
464             . . . . .
465             . . . . .
466             . . . . .
467             . . . . .
468             . . . . .
469             . . . . .
470             . . . . .
471             . . . . .
472             . . . . .
473             . . . . .
474             . . . . .
475             . . . . .
476             . . . . .
477             . . . . .
478             . . . . .
479             . . . . .
480             . . . . .
481             . . . . .
482             . . . . .
483             . . . . .
484             . . . . .
485             . . . . .
486             . . . . .
487             . . . . .
488             . . . . .
489             . . . . .
490             . . . . .
491             . . . . .
492             . . . . .
493             . . . . .
494             . . . . .
495             . . . . .
496             . . . . .
497             . . . . .
498             . . . . .
499             . . . . .
500             . . . . .
501             . . . . .
502             . . . . .
503             . . . . .
504             . . . . .
505             . . . . .
506             . . . . .
507             . . . . .
508             . . . . .
509             . . . . .
510             . . . . .
511             . . . . .
512             . . . . .
513             . . . . .
514             . . . . .
515             . . . . .
516             . . . . .
517             . . . . .
518             . . . . .
519             . . . . .
520             . . . . .
521             . . . . .
522             . . . . .
523             . . . . .
524             . . . . .
525             . . . . .
526             . . . . .
527             . . . . .
528             . . . . .
529             . . . . .
530             . . . . .
531             . . . . .
532             . . . . .
533             . . . . .
534             . . . . .
535             . . . . .
536             . . . . .
537             . . . . .
538             . . . . .
539             . . . . .
540             . . . . .
541             . . . . .
542             . . . . .
543             . . . . .
544             . . . . .
545             . . . . .
546             . . . . .
547             . . . . .
548             . . . . .
549             . . . . .
550             . . . . .
551             . . . . .
552             . . . . .
553             . . . . .
554             . . . . .
555             . . . . .
556             . . . . .
557             . . . . .
558             . . . . .
559             . . . . .
560             . . . . .
561             . . . . .
562             . . . . .
563             . . . . .
564             . . . . .
565             . . . . .
566             . . . . .
567             . . . . .
568             . . . . .
569             . . . . .
570             . . . . .
571             . . . . .
572             . . . . .
573             . . . . .
574             . . . . .
575             . . . . .
576             . . . . .
577             . . . . .
578             . . . . .
579             . . . . .
580             . . . . .
581             . . . . .
582             . . . . .
583             . . . . .
584             . . . . .
585             . . . . .
586             . . . . .
587             . . . . .
588             . . . . .
589             . . . . .
590             . . . . .
591             . . . . .
592             . . . . .
593             . . . . .
594             . . . . .
595             . . . . .
596             . . . . .
597             . . . . .
598             . . . . .
599             . . . . .
600             . . . . .
601             . . . . .
602             . . . . .
603             . . . . .
604             . . . . .
605             . . . . .
606             . . . . .
607             . . . . .
608             . . . . .
609             . . . . .
610             . . . . .
611             . . . . .
612             . . . . .
613             . . . . .
614             . . . . .
615             . . . . .
616             . . . . .
617             . . . . .
618             . . . . .
619             . . . . .
620             . . . . .
621             . . . . .
622             . . . . .
623             . . . . .
624             . . . . .
625             . . . . .
626             . . . . .
627             . . . . .
628             . . . . .
629             . . . . .
630             . . . . .
631             . . . . .
632             . . . . .
633             . . . . .
634             . . . . .
635             . . . . .
636             . . . . .
637             . . . . .
638             . . . . .
639             . . . . .
640             . . . . .
641             . . . . .
642             . . . . .
643             . . . . .
644             . . . . .
645             . . . . .
646             . . . . .
647             . . . . .
648             . . . . .
649             . . . . .
650             . . . . .
651             . . . . .
652             . . . . .
653             . . . . .
654             . . . . .
655             . . . . .
656             . . . . .
657             . . . . .
658             . . . . .
659             . . . . .
660             . . . . .
661             . . . . .
662             . . . . .
663             . . . . .
664             . . . . .
665             . . . . .
666             . . . . .
667             . . . . .
668             . . . . .
669             . . . . .
670             . . . . .
671             . . . . .
672             . . . . .
673             . . . . .
674             . . . . .
675             . . . . .
676             . . . . .
677             . . . . .
678             . . . . .
679             . . . . .
680             . . . . .
681             . . . . .
682             . . . . .
683             . . . . .
684             . . . . .
685             . . . . .
686             . . . . .
687             . . . . .
688             . . . . .
689             . . . . .
690             . . . . .
691             . . . . .
692             . . . . .
693             . . . . .
694             . . . . .
695             . . . . .
696             . . . . .
697             . . . . .
698             . . . . .
699             . . . . .
700             . . . . .
701             . . . . .
702             . . . . .
703             . . . . .
704             . . . . .
705             . . . . .
706             . . . . .
707             . . . . .
708             . . . . .
709             . . . . .
710             . . . . .
711             . . . . .
712             . . . . .
713             . . . . .
714             . . . . .
715             . . . . .
716             . . . . .
717             . . . . .
718             . . . . .
719             . . . . .
720             . . . . .
721             . . . . .
722             . . . . .
723             . . . . .
724             . . . . .
725             . . . . .
726             . . . . .
727             . . . . .
728             . . . . .
729             . . . . .
730             . . . . .
731             . . . . .
732             . . . . .
733             . . . . .
734             . . . . .
735             . . . . .
736             . . . . .
737             . . . . .
738             . . . . .
739             . . . . .
740             . . . . .
741             . . . . .
742             . . . . .
743             . . . . .
744             . . . . .
745             . . . . .
746             . . . . .
747             . . . . .
748             . . . . .
749             . . . . .
750             . . . . .
751             . . . . .
752             . . . . .
753             . . . . .
754             . . . . .
755             . . . . .
756             . . . . .
757             . . . . .
758             . . . . .
759             . . . . .
760             . . . . .
761             . . . . .
762             . . . . .
763             . . . . .
764             . . . . .
765             . . . . .
766             . . . . .
767             . . . . .
768             . . . . .
769             . . . . .
770             . . . . .
771             . . . . .
772             . . . . .
773             . . . . .
774             . . . . .
775             . . . . .
776             . . . . .
777             . . . . .
778             . . . . .
779             . . . . .
780             . . . . .
781             . . . . .
782             . . . . .
783             . . . . .
784             . . . . .
785             . . . . .
786             . . . . .
787             . . . . .
788             . . . . .
789             . . . . .
790             . . . . .
791             . . . . .
792             . . . . .
793             . . . . .
794             . . . . .
795             . . . . .
796             . . . . .
797             . . . . .
798             . . . . .
799             . . . . .
800             . . . . .
801             . . . . .
802             . . . . .
803             . . . . .
804             . . . . .
805             . . . . .
806             . . . . .
807             . . . . .
808             . . . . .
809             . . . . .
810             . . . . .
811             . . . . .
812             . . . . .
813             . . . . .
814             . . . . .
815             . . . . .
816             . . . . .
817             . . . . .
818             . . . . .
819             . . . . .
820             . . . . .
821             . . . . .
822             . . . . .
823             . . . . .
824             . . . . .
825             . . . . .
826             . . . . .
827             . . . . .
828             . . . . .
829             . . . . .
830             . . . . .
831             . . . . .
832             . . . . .
833             . . . . .
834             . . . . .
835             . . . . .
836             . . . . .
837             . . . . .
838             . . . . .
839             . . . . .
840             . . . . .
841             . . . . .
842             . . . . .
843             . . . . .
844             . . . . .
845             . . . . .
846             . . . . .
847             . . . . .
848             . . . . .
849             . . . . .
850             . . . . .
851             . . . . .
852             . . . . .
853             . . . . .
854             . . . . .
855             . . . . .
856             . . . . .
857             . . . . .
858             . . . . .
859             . . . . .
860             . . . . .
861             . . . . .
862             . . . . .
863             . . . . .
864             . . . . .
865             . . . . .
866             . . . . .
867             . . . . .
868             . . . . .
869             . . . . .
870             . . . . .
871             . . . . .
872             . . . . .
873             . . . . .
874             . . . . .
875             . . . . .
876             . . . . .
877             . . . . .
878             . . . . .
879             . . . . .
880             . . . . .
881             . . . . .
882             . . . . .
883             . . . . .
884             . . . . .
885             . . . . .
886             . . . . .
887             . . . . .
888             . . . . .
889             . . . . .
890             . . . . .
891             . . . . .
892             . . . . .
893             . . . . .
894             . . . . .
895             . . . . .
896             . . . . .
897             . . . . .
898             . . . . .
899             . . . . .
900             . . . . .
901             . . . . .
902             . . . . .
903             . . . . .
904             . . . . .
905             . . . . .
906             . . . . .
907             . . . . .
908             . . . . .
909             . . . . .
910             . . . . .
911             . . . . .
912             . . . . .
913             . . . . .
914             . . . . .
915             . . . . .
916             . . . . .
917             . . . . .
918             . . . . .
919             . . . . .
920             . . . . .
921             . . . . .
922             . . . . .
923             . . . . .
924             . . . . .
925             . . . . .
926             . . . . .
927             . . . . .
928             . . . . .
929             . . . . .
930             . . . . .
931             . . . . .
932             . . . . .
933             . . . . .
934             . . . . .
935             . . . . .
936             . . . . .
937             . . . . .
938             . . . . .
939             . . . . .
940             . . . . .
941             . . . . .
942             . . . . .
943             . . . . .
944             . . . . .
945             . . . . .
946             . . . . .
947             . . . . .
948             . . . . .
949             . . . . .
950             . . . . .
951             . . . . .
952             . . . . .
953             . . . . .
954             . . . . .
955             . . . . .
956             . . . . .
957             . . . . .
958             . . . . .
959             . . . . .
960             . . . . .
961             . . . . .
962             . . . . .
963             . . . . .
964             . . . . .
965             . . . . .
966             . . . . .
967             . . . . .
968             . . . . .
969             . . . . .
970             . . . . .
971             . . . . .
972             . . . . .
973             . . . . .
974             . . . . .
975             . . . . .
976             . . . . .
977             . . . . .
978             . . . . .
979             . . . . .
980             . . . . .
981             . . . . .
982             . . . . .
983             . . . . .
984             . . . . .
985             . . . . .
986             . . . . .
987             . . . . .
988             . . . . .
989             . . . . .
990             . . . . .
991             . . . . .
992             . . . . .
993             . . . . .
994             . . . . .
995             . . . . .
996             . . . . .
997             . . . . .
998             . . . . .
999             . . . . .
1000            . . . . .

```

```

322 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
323     DLmw
324 146,819,332 121 119 52,436,987 44 38 20,973,112 1,310,752
      1,310,750 events annotated

```

```

1 7 56 84 81 58 109 114 54 63 131 99 85 146 137 153 103 107 142
   50 174 152 80 61 102 87 44 122 120 57 110 143 111 93 153
   139 116 90 106 116 61 141 128 92 55 30 64 51 74 71 74
2 bVEX; rerun with -h for copyright info
3 ==669== Command: /tmp/02-mmult
4 ==669== Parent PID: 597
5 ==669==
6 --669-- Warning: Cannot auto-detect cache config, using
      defaults.
7 --669-- Run with -v to see.
8 ==669==
9 ==669== I refs: 147,273,019

```

```

10 ==669== I1 misses:          2,407
11 ==669== LLi misses:        2,389
12 ==669== I1 miss rate:      0.00%
13 ==669== LLi miss rate:     0.00%
14 ==669==
15 ==669== D refs:           73,583,942 (52,551,909 rd +
    21,032,033 wr)
16 ==669== D1 misses:        1,314,875 ( 3,538 rd +
    1,311,337 wr)
17 ==669== LLd misses:        1,314,228 ( 2,936 rd +
    1,311,292 wr)
18 ==669== D1 miss rate:      1.8% ( 0.0% +
    6.2% )
19 ==669== LLd miss rate:     1.8% ( 0.0% +
    6.2% )
20 ==669==
21 ==669== LL refs:           1,317,282 ( 5,945 rd +
    1,311,337 wr)
22 ==669== LL misses:        1,316,617 ( 5,325 rd +
    1,311,292 wr)
23 ==669== LL miss rate:      0.6% ( 0.0% +
    6.2% )
24 -----
25 I1 cache:                  32768 B, 32 B, 4-way associative
26 D1 cache:                  32768 B, 32 B, 4-way associative
27 LL cache:                  524288 B, 32 B, 8-way associative
28 Command:                   /tmp/02-mmult
29 Data file:                  cachegrind.out.669
30 Events recorded:            Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
31 Events shown:              Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
32 Event sort order:          Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
33 Thresholds:                 0.1 100 100 100 100 100 100 100 100
34 Include dirs:
35 User annotated:             /root/CARPETA/tp2-2020-2q-src/main.c
36 Auto-annotation:           off
37
38 -----
39 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
    DLmw
40 -----
41 147,273,019 2,407 2,389 52,551,909 3,538 2,936 21,032,033
    1,311,337 1,311,292 PROGRAM TOTALS
42
43 -----
44 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
    DLmw          file:function
45 -----
46 146,815,701 29 29 52,436,124 41 41 20,972,595
    1,310,731 1,310,731 /root/CARPETA/tp2-2020-2q-src/main.c:

```

```

47      matrix_multiply
48      -----
49      -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
50      -----
51      Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
52      DLmw
53      -- line 18 -----
54      .          .          .          .          .          .          .
55      .          .          .          .          .          .          .
56      .          .          .          .          .          .          .
57      .          .          .          .          .          .          .
58      .          .          .          .          .          .          .
59      .          .          .          .          .          .          .
60      .          .          .          .          .          .          .
61      .          .          .          .          .          .          .
62      10      2      2      0      0      0      5      0
63      .          .          .          .          .          .          .
64      .          .          .          .          .          .          .
65      .          .          .          .          .          .          .
66      .          .          .          .          .          .          .
67      .          .          .          .          .          .          .
68      1      0      0      0      0      0      1      0
69      .          .          .          .          .          .          .
70      .          .          .          .          .          .          .
71      .          .          .          .          .          .          .
72      .          .          .          .          .          .          .
73      .          .          .          .          .          .          .

```

```

74      2      1      1      0      0      0      1      0
75      .      .      .      size_t lineneno = 1;
76      .      .      .      struct timespec t0;
77      .      .      .      struct timespec t1;
78      .      .      .      double dt;
79      .      .      .      size_t i;
80      25     4      4      9      0      0      1      0
81      10     1      1      4      0      0      6      0
82      .      .      .      a=b=c=NULL;
83      18     3      3      8      1      0      2      0
84      .      .      .      line = read_line(stdin);
85      6      0      0      2      0      0      0      0
86      9      0      0      4      0      0      0      0
87      .      .      .      if (!line) goto _exit_main;
88      .      .      .      if (line[0] == 0) break;
89      2      1      1      .      .      .      .      .
90      10     1      1      3      1      0      1      0
91      8      1      1      3      0      0      0      0
92      .      .      .      if (errno) {
93      .      .      .      perror("");
94      .      .      .      goto _exit_main;
95      4      2      2      }
96      .      .      .      if (nptr == endptr) {
97      .      .      .      fprintf(stderr, "missing
98      .      .      .      dimension");
99      3      2      2      1      0      0      0      0
100     .      .      .      if (l < 1) {

```

```

dimension");
101         .           .           .           .           .
102             .           goto _exit_main;
103     }
104     2    1    1        1    0    0        1    0
105         0      n = (size_t) 1;
106         .           .           .           .           .
107         .           #if 0
108         .           /* parse block size */
109         .           nptr = endptr;
110         .           l = strtol(nptr, &endptr, 10);
111         .           if (errno) {
112         .               perror("");
113         .               goto _exit_main;
114         .           }
115 -- line 75 -----
116 -- line 83 -----
117         .           .           .           .           .
118         .           }
119         .           bs = (size_t) 1;
120         .           .           .           .           .
121         .           if (n % bs) {
122         .               fprintf(stderr, "block size doesn
't match");
123         .               goto _exit_main;
124         .           }
125         .           #else
126     2    0    0        1    0    0        1    0
127         0      bs = n;
128         .           #endif
129         .           .           .           .           .
130         .           /* load matrix a */
131     11    1    1        5    1    0        1    0
132         0      if (!(a = create_matrix(n, n)))
133         .           .           .           .           .

```

```

128         .         .         .         goto _exit_main;         .         .
129     500     3     3         199     0     0         50         0
130         .         .         .         for (i=0; i < n*n; i++) {
131     441     1     1         49     0     0         49         0
132     392     1     1         nptr = endptr;
133         .         .         .         e = strtod(nptr, &endptr);
134         .         .         .         147     0     0         0         0
135         .         .         .         if (errno) {
136     196     2     2         .         .         .         perror("");
137         .         .         .         goto _exit_main;
138         .         .         .         }
139         .         .         .         98     0     0         0         0
140     343     1     1         if (nptr == endptr) {
141         .         .         .         fprintf(stderr, "missing A
142         .         .         .         matrix element");
143         .         .         .         goto _exit_main;
144     11     1     1         .         .         .         }
145         .         .         .         196     0     0         49         11
146         .         .         .         a->array[i] = e;
147         .         .         .         }
148         .         .         .         .         .         .
149         .         .         .         /* load matrix b */
150     392     1     1         5     0     0         1         0
151         .         .         .         if (!(b = create_matrix(n, n)))
152         .         .         .         goto _exit_main;
153         .         .         .         .         .         .
154     500     2     2         199     0     0         50         0
155     98     1     1         for (i=0; i < n*n; i++) {
156         .         .         .         49     0     0         49         0
157     441     1     1         nptr = endptr;
158         .         .         .         147     0     0         49         0
159     392     1     1         e = strtod(nptr, &endptr);
160         .         .         .         147     0     0         0         0
161         .         .         .         if (errno) {
162         .         .         .         perror("");
163         .         .         .         goto _exit_main;
164         .         .         .         }
165         .         .         .         }

```



```

154      196      1      1      98      0      0      0      0
          0      if (nptr == endptr) {
155      .      .      .      .      .      .      .      .
          .      fprintf(stderr, "missing B
156      .      .      .      .      .      .      .      .
          .      goto _exit_main;
157      .      .      .      .      .      .      .      .
          .      }
158      343      1      1      196      0      0      49      11
          11      b->array[i] = e;
159      .      .      .      .      .      .      .      .
          .      }
160      .      .      .      .      .      .      .      .
          .      .
161      8      1      1      2      0      0      0      0
          0      clock_gettime(CLOCK_REALTIME, &t0);
162      .      .      .      .      .      .      .      .
          .      .
163      .      .      .      .      .      .      .      .
          .      /* multiply matrixes */
164      13      2      2      6      1      1      1      0
          0      if (!(c = matrix_multiply(a, b, bs))
          )
165      .      .      .      .      .      .      .      .
          .      goto _exit_main;
166      .      .      .      .      .      .      .      .
          .      .
167      8      1      1      2      0      0      0      0
          0      clock_gettime(CLOCK_REALTIME, &t1);
168      .      .      .      .      .      .      .      .
          .      .
169      7      1      1      2      1      1      1      1
          1      dt = (float) (t1.tv_sec - t0.tv_sec
          );
170      12      1      1      5      2      2      1      0
          0      dt = dt + ((float)(t1.tv_nsec - t0.
          tv_nsec)) / 1.0e9;
171      .      .      .      .      .      .      .      .
          .      .
172      13      3      2      5      2      2      0      0
          0      if(print_matrix(stdout, c) == -1)
173      .      .      .      .      .      .      .      .
          .      goto _exit_main;
174      .      .      .      .      .      .      .      .
          .      .
175      12      1      1      7      0      0      0      0
          0      fprintf(stderr, "time: %g\n", dt);
176      .      .      .      .      .      .      .      .
          .      .
177      6      1      1      3      0      0      0      0
          0      free(line);
178      6      1      1      3      0      0      0      0
          0      destroy_matrix(a);

```

```

179      6      1      1      3      0      0      0      0
180      6      0      0      3      0      0      0      0
181      .      .      .      .      .      .      .      .
182      .      .      .      .      .      .      .      .
183      3      1      1      0      0      0      0      0
184      .      .      .      .      .      .      .      .
185      .      .      .      .      .      .      .      .
186      .      .      .      .      .      .      .      .
187      .      .      .      .      .      .      .      .
188      .      .      .      .      .      .      .      .
189      .      .      .      .      .      .      .      .
190      .      .      .      .      .      .      .      .
191      .      .      .      .      .      .      .      .
192      6      1      1      2      0      0      0      0
193      .      .      .      .      .      .      .      .
194      .      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .      .
196      11     2      2      0      0      0      6      0
197      .      .      .      .      .      .      .      .
198      .      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .      .
200      .      .      .      .      .      .      .      .
201      3      1      1      2      0      0      1      0
202      .      .      .      .      .      .      .      .
203      11     1      1      5      0      0      1      0
204      .      .      .      .      .      .      .      .

```

205	9	2	2	0	3	0	0	1	0
206
207	64	2	2	23	0	0	8	0	0
208	448	1	1	0	161	0	0	56	0
209	539	1	1	0	245	0	0	98	11
210	.	.	.	11	mr->array[i*n+j]	=	0.0;	.	.
211
212	.	.	.	#if 1
213	if (1) {
214	2	0	0	0	size_t j;	0	0	1	0
215	9	1	1	0	size_t dim = 1024*1024*10;	3	0	1	0
216	83,886,088	2	2	0	int *v = malloc(dim*sizeof(int));	0	0	10,485,761	0
217	62,914,560	0	0	0	for (j = 0; j < dim; ++j)	0	0	10,485,760	1,310,720
218	1,310,720	6	1	1	v[j] = -1;	3	2	2	0
219	.	.	.	0	free(v);
220	}
221	#endif
222	17	2	2	0	6	0	0	2	0
223	17	2	2	0	for(kk=0; kk<en; kk+=bs)	6	0	0	2
224	64	1	1	0	for(jj=0; jj<en; jj+=bs)	23	0	0	8
225	567	2	2	0	for(i=0; i<n; i++)	224	0	0	56
226	539	1	1	0	for(j=jj; j<jj+bs; j++) {	294	13	13	49
227	3,969	3	3	0	sum = mr->array[i*n+j];	1,568	0	0	392
228	8,232	2	2	0	for(k=kk; k<kk+bs; k++)	4,459	26	26	343
229	539	1	1	0	sum += m1->array[i*n+k] * m2->	294	0	0	49
230	array[k*n+j];	mr->array[i*n+j]	=	sum;	.
	}

```

231      1    0    0      1    0    0      0    0
           0      return mr;
232      6    1    1      2    0    0      0    0
           0    }
233      .    .    .      .    .    .      .    .
           .
234      .    .    .      .    .    .      .    .
           .      char*
235      .    .    .      .    .    .      .    .
           .      read_line(FILE *fp)
236     18    3    3      0    0    0      8    0
           0    {
237      .    .    .      .    .    .      .    .
           .      #define DEF_LINE_SZ 1024
238      .    .    .      .    .    .      .    .
           .
239      .    .    .      .    .    .      .    .
           .      int c;
240      6    0    0      0    0    0      4    0
           0      size_t len = 0, tam = DEF_LINE_SZ;
241      .    .    .      .    .    .      .    .
           .      char* str;
242      .    .    .      .    .    .      .    .
           .
243     14    1    1      6    1    0      2    1
           0      str = malloc(tam);
244      6    0    0      2    0    0      0    0
           0      if (!str) {
245      .    .    .      .    .    .      .    .
           .      perror("");
246      .    .    .      .    .    .      .    .
           .      return NULL;
247      .    .    .      .    .    .      .    .
           .      }
248      .    .    .      .    .    .      .    .
           .
249     2,981   5    5      994    1    0      199    0
           0      while (EOF != (c=fgetc(fp)) && c != '\n')
           {
250     1,576   2    2      591    0    0      394    6
           6      str[len++]=c;
251     985    0    0      394    0    0      0    0
           0      if (len==tam-1) {
252      .    .    .      .    .    .      .    .
           .      str = realloc(str, tam *= 2);
253      .    .    .      .    .    .      .    .
           .      if (!str) {
254      .    .    .      .    .    .      .    .
           .      perror("");
255      .    .    .      .    .    .      .    .
           .      return NULL;
256      .    .    .      .    .    .      .    .
           .      }
257      .    .    .      .    .    .      .    .

```

```

258         .      .      .      .      .      .      .      .
259         .      .      .      .      .      .      .      .
260         8      1      1      .      2      0      0      0      0
261         7      0      0      .      2      0      0      2      0
262         .      .      .      .      .      .      .      .
263         12     1      1      .      4      0      0      4      0
264         2      0      0      .      2      0      0      0      0
265         12     1      1      .      4      0      0      0      0
266         .      .      .      .      .      .      .      .
267         .      .      .      .      .      .      .      .
268         .      .      .      .      .      .      .      .
269         .      .      .      .      .      .      .      .
270         30     1      1      .      0      0      0      15     0
271         .      .      .      .      .      .      .      .
272         .      .      .      .      .      .      .      .
273         30     2      2      .      9      1      0      3      0
274         .      .      .      .      .      .      .      .
275         .      .      .      .      .      .      .      .
276         .      .      .      .      .      .      .      .
277         .      .      .      .      .      .      .      .
278         9      1      1      .      6      0      0      3      0
279         9      1      1      .      6      0      0      3      0
280         51     2      2      .      21     0      0      3      0
281         .      .      .      .      .      .      .      .
282         .      .      .      .      .      .      .      .
283         .      .      .      .      .      .      .      .

```

```

284         .      .      .      .      .      .      .      .
285         .      .      .      .      .      .      .      .
286         3      1      1      .      3      0      0      0      0
287         .      .      .      .      .      .      .      .
288         .      .      .      .      .      .      .      .
289         .      .      .      .      .      .      .      .
290         .      .      .      .      .      .      .      .
291         .      .      .      .      .      .      .      .
292         .      .      .      .      .      .      .      .
293         .      .      .      .      .      .      .      .
294         .      .      .      .      .      .      .      .
295         .      .      .      .      .      .      .      .
296         .      .      .      .      .      .      .      .
297         .      .      .      .      .      .      .      .
298         .      .      .      .      .      .      .      .
299         .      .      .      .      .      .      .      .
300         .      .      .      .      .      .      .      .
301         .      .      .      .      .      .      .      .
302         .      .      .      .      .      .      .      .
303         .      .      .      .      .      .      .      .
304         .      .      .      .      .      .      .      .
305         .      .      .      .      .      .      .      .
306         .      .      .      .      .      .      .      .
307         .      .      .      .      .      .      .      .
308         .      .      .      .      .      .      .      .
309         .      .      .      .      .      .      .      .

```

```

310         . . . . . perror("");
311         . . . . . return -1;
312         . . . . . }
313     10    1    1    4    0    0    0    0
314         . . . . . if (fprintf(fp, "\n") < 0) {
315         . . . . . perror("");
316         . . . . . return -1;
317         . . . . . }
318     1    1    1    0    0    0    0    0
319         . . . . . return 0;
320     6    1    1    2    0    0    0    0
321         . . . . . }

```

```

321 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
322      DLmw
323 146,827,393 120 119 52,440,658 54 48 20,973,736 1,310,762
      1,310,760 events annotated

```

```

1  8 240 184 111 110 129 175 191 230 105 183 118 82 140 181 172
   148 179 136 110 107 151 113 116 165 185 178 143 170 219 133
   177 202 240 232 166 119 161 187 203 268 181 158 128 109
   199 101 128 205 229 254 165 163 208 223 208 251 178 181 135
   176 205 159 197 186
2  PID: 597
3  ==672==
4  --672-- Warning: Cannot auto-detect cache config, using
   defaults.
5  --672--          Run with -v to see.
6  ==672==
7  ==672== I   refs:          147,337,351
8  ==672== I1  misses:          2,407
9  ==672== LLi misses:          2,389
10 ==672== I1  miss rate:          0.00%
11 ==672== LLi miss rate:          0.00%
12 ==672==
13 ==672== D   refs:          73,608,460 (52,568,671 rd +
   21,039,789 wr)
14 ==672== D1  misses:          1,314,908 ( 3,550 rd +
   1,311,358 wr)
15 ==672== LLd misses:          1,314,257 ( 2,949 rd +
   1,311,308 wr)
16 ==672== D1  miss rate:          1.8% ( 0.0% +
   6.2% )
17 ==672== LLd miss rate:          1.8% ( 0.0% +

```

```

6.2% )
18 ==672==
19 ==672== LL refs:          1,317,315 (      5,957 rd  +
      1,311,358 wr)
20 ==672== LL misses:      1,316,646 (      5,338 rd  +
      1,311,308 wr)
21 ==672== LL miss rate:          0.6% (      0.0%  +
      6.2% )
22 -----
23 I1 cache:          32768 B, 32 B, 4-way associative
24 D1 cache:          32768 B, 32 B, 4-way associative
25 LL cache:          524288 B, 32 B, 8-way associative
26 Command:           /tmp/02-mmult
27 Data file:          cachegrind.out.672
28 Events recorded:    Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
29 Events shown:       Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
30 Event sort order:  Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
31 Thresholds:         0.1 100 100 100 100 100 100 100 100
32 Include dirs:
33 User annotated:     /root/CARPETA/tp2-2020-2q-src/main.c
34 Auto-annotation:    off
35
36 -----
37 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw
38 -----
39 147,337,351 2,407 2,389 52,568,671 3,550 2,949 21,039,789
      1,311,358 1,311,308 PROGRAM TOTALS
40
41 -----
42 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw          file:function
43 -----
44 146,822,412 29 29 52,439,429 52 52 20,973,042
      1,310,735 1,310,735 /root/CARPETA/tp2-2020-2q-src/main.c:
      matrix_multiply
45
46 -----
47 -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
48 -----
49 Ir          I1mr ILmr Dr          D1mr DLmr Dw          D1mw
      DLmw
50
51 -- line 18 -----
52 . . . . .
      . matrix_t* create_matrix(size_t rows,
      size_t cols);

```



```

53      .      .      .      .      .      .      .      .
54      .      .      .      .      .      .      .      .
55      .      .      .      .      .      .      .      .
56      .      .      .      .      .      .      .      .
57      .      .      .      .      .      .      .      .
58      .      .      .      .      .      .      .      .
59      .      .      .      .      .      .      .      .
60      .      .      .      .      .      .      .      .
61      .      .      .      .      .      .      .      .
62      .      .      .      .      .      .      .      .
63      .      .      .      .      .      .      .      .
64      .      .      .      .      .      .      .      .
65      .      .      .      .      .      .      .      .
66      .      .      .      .      .      .      .      .
67      .      .      .      .      .      .      .      .
68      .      .      .      .      .      .      .      .
69      .      .      .      .      .      .      .      .
70      .      .      .      .      .      .      .      .
71      .      .      .      .      .      .      .      .
72      .      .      .      .      .      .      .      .
73      .      .      .      .      .      .      .      .
74      .      .      .      .      .      .      .      .
75      .      .      .      .      .      .      .      .
76      .      .      .      .      .      .      .      .
77      .      .      .      .      .      .      .      .
78      .      .      .      .      .      .      .      .

```

```

    void destroy_matrix(matrix_t* m);

    int print_matrix(FILE* fp, matrix_t* m)
;

    int
    main(int argc, char** argv)
10      2      2      0      0      0      5      0
    {
        /* matrixes */
        matrix_t *a, *b, *c;
        /* n (dimension) and block size */
        size_t n, bs;
        /* line buffer (init to null to
simplify freeing on error) */
66      1      0      0      0      0      0      1      0
        char *line = NULL;
        /* line parsing auxiliar pointers */
        char *nptr, *endptr;
        /* auxiliar variables */
        long l;
        double e;
72      2      1      1      0      0      0      1      0
        size_t lineno = 1;
        struct timespec t0;
        struct timespec t1;
        double dt;
        size_t i;
78      25      4      4      9      0      0      1      0
        for(;; !feof(stdin); lineno++) {

```

```

79      10      1      1          4      0      0          6      0
      .      .      .          a=b=c=NULL;
80      .      .      .          .      .      .          .      .
81      18      3      3          8      1      0          2      0
      .      .      .          line = read_line(stdin);
82      .      .      .          .      .      .          .      .
83      6      0      0          2      0      0          0      0
      .      .      .          if (!line) goto _exit_main;
84      9      0      0          4      0      0          0      0
      .      .      .          if (line[0] == 0) break;
85      .      .      .          .      .      .          .      .
86      .      .      .          .      .      .          .      .
87      2      1      1          1      0      0          1      1
      .      .      .          nptr = line;
88      10     1      1          3      1      0          1      0
      .      .      .          l = strtol(nptr, &endptr, 10);
89      8      1      1          3      0      0          0      0
      .      .      .          if (errno) {
90      .      .      .          .      .      .          .      .
91      .      .      .          perror("");
92      .      .      .          goto _exit_main;
93      4      2      2          2      0      0          0      0
      .      .      .          if (nptr == endptr) {
94      .      .      .          fprintf(stderr, "missing
dimension");
95      .      .      .          goto _exit_main;
96      .      .      .          .      .      .          .      .
97      3      2      2          1      0      0          0      0
      .      .      .          if (l < 1) {
98      .      .      .          fprintf(stderr, "invalid
dimension");
99      .      .      .          goto _exit_main;
100     .      .      .          .      .      .          .      .
101     2      1      1          1      0      0          1      0
      .      .      .          }
102     .      .      .          n = (size_t) l;
103     .      .      .          #if 0
104     .      .      .          .      .      .          .      .
          .      .      .          /* parse block size */
          .      .      .          nptr = endptr;

```

```

105         .         .         .         .         .         .         .         .         .         .
106         .         .         .         .         .         .         .         .         .         .
107         .         .         .         .         .         .         .         .         .         .
108         .         .         .         .         .         .         .         .         .         .
109         .         .         .         .         .         .         .         .         .         .
110         .         .         .         .         .         .         .         .         .         .
111         .         .         .         .         .         .         .         .         .         .
112         .         .         .         .         .         .         .         .         .         .
113         .         .         .         .         .         .         .         .         .         .
114         .         .         .         .         .         .         .         .         .         .
115         .         .         .         .         .         .         .         .         .         .
116         .         .         .         .         .         .         .         .         .         .
117         .         .         .         .         .         .         .         .         .         .
118         .         .         .         .         .         .         .         .         .         .
119         .         .         .         .         .         .         .         .         .         .
120         .         .         .         .         .         .         .         .         .         .
121         .         .         .         .         .         .         .         .         .         .
122         .         .         .         .         .         .         .         .         .         .
123         .         .         .         .         .         .         .         .         .         .
124         .         .         .         .         .         .         .         .         .         .
125         .         .         .         .         .         .         .         .         .         .
126         .         .         .         .         .         .         .         .         .         .
127         .         .         .         .         .         .         .         .         .         .
128         .         .         .         .         .         .         .         .         .         .
129         .         .         .         .         .         .         .         .         .         .
130         .         .         .         .         .         .         .         .         .         .
131         .         .         .         .         .         .         .         .         .         .
132         .         .         .         .         .         .         .         .         .         .

        l = strtol(nptr, &endptr, 10);

        if (errno) {
            perror("");
            goto _exit_main;
        }

-- line 75 -----
-- line 83 -----

        }

        bs = (size_t) l;

        if (n % bs) {
            fprintf(stderr, "block size doesn
't match");
            goto _exit_main;
        }

        #else
        2      0      0      1      0      0      1      0
          0      bs = n;
        #endif

        /* load matrix a */
        11     1      1      5      1      0      1      0
          0      if (!(a = create_matrix(n, n)))
            goto _exit_main;

        650    3      3      259    0      0      65      0
          0      for (i=0; i < n*n; i++) {
        128    0      0      64      0      0      64      0
          0      nptr = endptr;
        576    1      1      192    0      0      64      0
          0      e = strtod(npstr, &endptr);
        512    1      1      192    0      0      0      0
          0      if (errno) {
            perror("");
        
```

```

133         goto _exit_main;
134     .   .   .   .   .   .   .   .
135     .   .   .   .   .   .   .   .
136     .   .   .   .   .   .   .   .
137     .   .   .   .   .   .   .   .
138     448   1   1   256   0   0   64   15
139         15   a->array[i] = e;
140     .   .   .   .   .   .   .   .
141     .   .   .   .   .   .   .   .
142     11   1   1   /* load matrix b */
143         0   5   0   0   1   0
144         if (!(b = create_matrix(n, n)))
145             goto _exit_main;
146     .   .   .   .   .   .   .   .
147     650   2   2   259   0   0   65   0
148         0   for (i=0; i < n*n; i++) {
149     128   1   1   64   0   0   64   0
150         0   nptr = endptr;
151     576   1   1   192   0   0   64   0
152         0   e = strtod(nptr, &endptr);
153     512   1   1   192   0   0   0   0
154         0   if (errno) {
155     .   .   .   .   .   .   .   .
156         .   perror("");
157     .   .   .   .   .   .   .   .
158         .   goto _exit_main;
159     .   .   .   .   .   .   .   .
160     .   .   .   .   .   .   .   .
161     .   .   .   .   .   .   .   .
162     256   1   1   128   0   0   0   0
163         0   if (nptr == endptr) {
164     .   .   .   .   .   .   .   .
165         .   fprintf(stderr, "missing B
166     .   .   .   .   .   .   .   .
167         .   matrix element");
168     .   .   .   .   .   .   .   .
169         .   goto _exit_main;
170     .   .   .   .   .   .   .   .
171     .   .   .   .   .   .   .   .
172     .   .   .   .   .   .   .   .
173     .   .   .   .   .   .   .   .
174     448   1   1   256   0   0   64   16
175         16   b->array[i] = e;
176     .   .   .   .   .   .   .   .
177     .   .   .   .   .   .   .   .
178     .   .   .   .   .   .   .   .

```

```

159      8      1      1      .      2      0      0      0      0
160      .      .      .      0      clock_gettime(CLOCK_REALTIME, &t0);
161      .      .      .      .      .      .      .      .      .
162      13     2      2      .      /* multiply matrixes */
163      .      .      .      6      1      1      1      0
164      .      .      .      if (!(c = matrix_multiply(a, b, bs))
165      .      .      .      )
166      .      .      .      goto _exit_main;
167      .      .      .      .      .      .      .      .      .
168      8      1      1      .      2      0      0      0      0
169      .      .      .      0      clock_gettime(CLOCK_REALTIME, &t1);
170      .      .      .      .      .      .      .      .      .
171      7      1      1      .      2      1      1      1      1
172      .      .      .      1      dt = (float) (t1.tv_sec - t0.tv_sec
173      .      .      .      );
174      12     1      1      .      5      2      2      1      0
175      .      .      .      0      dt = dt + ((float)(t1.tv_nsec - t0.
176      .      .      .      tv_nsec)) / 1.0e9;
177      .      .      .      .      .      .      .      .      .
178      13     3      2      .      5      2      2      0      0
179      .      .      .      0      if(print_matrix(stdout, c) == -1)
180      .      .      .      .      .      .      .      .      .
181      .      .      .      .      goto _exit_main;
182      .      .      .      .      .      .      .      .      .
183      12     1      1      .      7      0      0      0      0
184      .      .      .      0      fprintf(stderr, "time: %g\n", dt);
185      .      .      .      .      .      .      .      .      .
186      6      1      1      .      3      0      0      0      0
187      .      .      .      0      free(line);
188      6      1      1      .      3      0      0      0      0
189      .      .      .      0      destroy_matrix(a);
190      6      1      1      .      3      0      0      0      0
191      .      .      .      0      destroy_matrix(b);
192      6      0      0      .      3      0      0      0      0
193      .      .      .      0      destroy_matrix(c);
194      .      .      .      .      .      .      .      .      .
195      .      .      .      .      .      .      .      .      .
196      3      1      1      .      0      0      0      0      0
197      .      .      .      0      return 0;
198      .      .      .      .      .      .      .      .      .
199      .      .      .      .      .      .      .      .      .
200      .      .      .      .      _exit_main:

```

```

184         fprintf(stderr, " at line %u\n", (
unsigned) lineno);
185
186         free(line);
187
188         destroy_matrix(a);
189         destroy_matrix(b);
190         destroy_matrix(c);
191
192         exit(1);
193
194     6   1   1   2   0   0   0   0
195     0 }
196
197     matrix_t*
198     matrix_multiply(matrix_t* m1, matrix_t*
m2, int bs)
199     11  2   2   0   0   0   6   0
200     0 {
201
202         size_t n, en, i, j, k, kk, jj;
203
204         double sum;
205
206         matrix_t* mr;
207
208     3   1   1   2   0   0   1   0
209     0   n = m1->rows;
210
211     11  1   1   5   0   0   1   0
212     0   if(!(mr = create_matrix(n,n))) return
NULL;
213
214     9   2   2   3   0   0   1   0
215     0   en = bs*(n/bs);
216
217     72  2   2   26   0   0   9   0
218     0   for(i=0; i<n; i++)
219     576  1   1   208   0   0   72   0
220     0   for(j=0; j<n; j++)
221     704  1   1   320   0   0   128   15
222     15   mr->array[i*n+j] = 0.0;
223
224
225
226
227
228
229

```

210	#if 1
211	if (1) {
212	2	0	0	0	size_t j;	0	0	0	1	0
213	9	1	1	0	size_t dim = 1024*1024*10;	3	0	0	1	0
214	83,886,088	2	2	31,457,282	int *v = malloc(dim*sizeof(int));	0	0	10,485,761	0	0
215	62,914,560	0	0	20,971,520	for (j = 0; j < dim; ++j)	0	0	10,485,760	1,310,720	0
216	1,310,720	6	1	1	v[j] = -1;	3	2	2	0	0
217	free(v);
218	}
219	#endif
220	17	2	2	0	for(kk=0; kk<en; kk+=bs)	6	0	0	2	0
221	17	2	2	0	for(jj=0; jj<en; jj+=bs)	6	0	0	2	0
222	72	1	1	0	for(i=0; i<n; i++)	26	0	0	9	0
223	728	2	2	0	for(j=jj; j<jj+bs; j++) {	288	0	0	72	0
224	704	1	1	0	sum = mr->array[i*n+j];	384	17	17	64	0
225	5,824	3	3	0	for(k=kk; k<kk+bs; k++)	2,304	0	0	576	0
226	12,288	2	2	0	sum += m1->array[i*n+k] * m2->	6,656	33	33	512	0
227	704	1	1	0	array[k*n+j];	384	0	0	64	0
228	mr->array[i*n+j] = sum;
229	1	0	0	0	}	1	0	0	0	0
230	6	1	1	0	return mr;	2	0	0	0	0
231	}
232	char*
233	read_line(FILE *fp)
234	18	3	3	0	{	0	0	0	8	0
235	#define DEF_LINE_SZ 1024

```

236      .      .      .      .      .      .      .      .
237      .      .      .      .      .      .      .      .
238      6      0      0      .      .      .      .      .
239      .      .      .      .      .      .      .      .
240      .      .      .      .      .      .      .      .
241      14     1      1      .      6      1      0      .      2      1
242      .      .      .      .      .      .      .      .
243      .      .      .      .      .      .      .      .
244      .      .      .      .      .      .      .      .
245      .      .      .      .      .      .      .      .
246      .      .      .      .      .      .      .      .
247      3,881  5      5      .      1,294  1      0      .      259      0
248      {      .      .      .      .      .      .      .      .
249      2,056  2      2      .      771      0      0      .      514      8
250      .      .      .      .      .      .      .      .
251      .      .      .      .      .      .      .      .
252      .      .      .      .      .      .      .      .
253      .      .      .      .      .      .      .      .
254      .      .      .      .      .      .      .      .
255      .      .      .      .      .      .      .      .
256      .      .      .      .      .      .      .      .
257      .      .      .      .      .      .      .      .
258      8      1      1      .      2      0      0      .      0      0
259      7      0      0      .      2      0      0      .      2      0
260      .      .      .      .      .      .      .      .
261      12     1      1      .      4      0      0      .      4      0
262      2      0      0      .      2      0      0      .      0      0

```



```

0      return str;
263    12   1   1       4   0   0           0           0
        0 }
264    .   .   .       .   .   .           .           .
        .
265    .   .   .       .   .   .           .           .
        .
266    .   .   .       .   .   .           .           .
        . matrix_t*
267    .   .   .       .   .   .           .           .
        . create_matrix(size_t rows, size_t cols)
268    30   1   1       0   0   0          15           0
        0 {
269    .   .   .       .   .   .           .           .
        . matrix_t * m;
270    .   .   .       .   .   .           .           .
        .
271    30   2   2       9   1   0           3           0
        0 if (!(m = malloc(sizeof(matrix_t)))) {
272    .   .   .       .   .   .           .           .
        . perror("");
273    .   .   .       .   .   .           .           .
        . return NULL;
274    .   .   .       .   .   .           .           .
        . }
275    .   .   .       .   .   .           .           .
        .
276    9   1   1       6   0   0           3           0
        0 m->rows = rows;
277    9   1   1       6   0   0           3           0
        0 m->cols = cols;
278    51   2   2       21  0   0           3           0
        0 if (!(m->array = malloc(sizeof(double)
        *rows*cols))) {
279    .   .   .       .   .   .           .           .
        . free(m);
280    .   .   .       .   .   .           .           .
        . perror("");
281    .   .   .       .   .   .           .           .
        . return NULL;
282    .   .   .       .   .   .           .           .
        . }
283    .   .   .       .   .   .           .           .
        .
284    3   1   1       3   0   0           0           0
        0 return m;
285    18   1   1       6   0   0           0           0
        0 }
286    .   .   .       .   .   .           .           .
        .
287    .   .   .       .   .   .           .           .
        . void
288    .   .   .       .   .   .           .           .
        . destroy_matrix(matrix_t* m)

```

```

289      27      1      1      0      0      0      12      0
290          0      0      {
291      24      1      1      12      0      0      0      0
          0      if (!m) return;
292      24      1      1      9      0      0      0      0
          0      free(m->array);
293      18      0      0      6      0      0      0      0
          0      free(m);
          0      }
294      .      .      .      .      .      .      .      .
295      .      .      .      .      .      .      .      .
          .      int
296      .      .      .      .      .      .      .      .
          .      print_matrix(FILE* fp, matrix_t* m)
297      10      2      2      0      0      0      5      0
          0      {
298      .      .      .      .      .      .      .      .
          .      size_t i, j;
299      .      .      .      .      .      .      .      .
          .      size_t n;
300      3      1      1      2      0      0      1      0
          0      n = m->rows;
301      13      1      1      6      1      1      0      0
          0      if (fprintf(fp, "%lu", (unsigned long)
m->rows) < 0) {
302      .      .      .      .      .      .      .      .
          .      perror("");
303      .      .      .      .      .      .      .      .
          .      return -1;
304      .      .      .      .      .      .      .      .
          .      }
305      72      2      2      26      0      0      9      0
          0      for(i=0; i<n; i++)
306      576      2      2      208      0      0      72      0
          0      for (j=0; j<n; j++)
307      1,408      3      3      640      0      0      0      0
          0      if (fprintf(fp, " %g", m->array[i*n+j
]) < 0) {
308      .      .      .      .      .      .      .      .
          .      perror("");
309      .      .      .      .      .      .      .      .
          .      return -1;
310      .      .      .      .      .      .      .      .
          .      }
311      10      1      1      4      0      0      0      0
          0      if (fprintf(fp, "\n") < 0) {
312      .      .      .      .      .      .      .      .
          .      perror("");
313      .      .      .      .      .      .      .      .
          .      return -1;
314      .      .      .      .      .      .      .      .
          .      }

```

315	1	1	1	0	0	0	0	0
			0	return	0;			
316	6	1	1	2	0	0	0	0
			0	}				
317								
318								
319	Ir	I1mr	ILmr	Dr	D1mr	DLmr	Dw	D1mw
		DLmw						
320								
321	146,837,450	120	119	52,445,273	65	59	20,974,500	1,310,777
	1,310,775	events	annotated					

7. Enunciado

El enunciado se encuentra anexado al final de este documento.