FACULTAD DE INGENIERIA
Universidad de Buenos Aires

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

El trabajo práctico consistió en un analisis teorico con una posterior simulacion de un benchmark para comparar distintos tipos de caches. Se realizarion 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 utilizo el mismo ambiente MIPS que en el TP1, pero instalando manualmente una version arreglada de valgrind. Que nos permitio correr cachegrind y cg annotate, para asi 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 catedra para generar el ejecutbale 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 si, 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 mas rapidas, al mappearse direcciones de forma directa 1 a 1 y no se necesitan politicas de borrado (LRU, por ejemplo). Pero esto obviamente tiene un contraefecto muy fuerte: si en la ejecucion de un programa, se producen conflictos de manera permanente, el miss rate es mas grande de lo que uno supondria. Para poner un ejemplo, un progrma que simplemente carga 2 valores en memoria, pero ambos colisionan a la misma direccion 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 demas espacios de cache libre, lo que haria 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=çgrind$_{11}x1.txt$"$--I1 = 32768, 4, 32-$
$-D1 = 32768, 1, 32--tool = cachegrind/tmp/02-mmult < /root/CARPETA/tp2-$
$2020-2q-src/input_1x1.txt > cgrind_{11}x1.txt; /opt/valgrind/bin/cg_annotatecachegrind.out.*$
$/root/CARPETA/tp2 - 2020 - 2q - src/main.c >> cgrind_{11}x1.txt;$

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.

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

```
#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
```

Podemos ver con claridad en el output del cg annotate, que la mayoria 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 codigo, que no parece hacer nada util para la multiplicacion 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 hipotesis iniciales.

En una futura reentrega esperamos poder realizar las correcciones necesarias para poder realizar un analisis mas exhaustivo y cumplir con los objetivos del trabajo practico.

5. Código del script

5.1. correr-simulacion.sh

6. Salida de las corridas

6.1. Direct mapping:

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

```
==598== D1  miss rate:          1.8% (       0.0%     +
    6.3%  )
==598== LLd miss rate:          1.8% (       0.0%     +
    6.2%  )
==598==
==598== LL refs:          1,356,387  (    27,145 rd   +
    1,329,242 wr)
==598== LL misses:        1,316,510  (     5,266 rd   +
    1,311,244 wr)
==598== 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.598
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
--------------------------------------------------------------------------------

147,098,909 2,390 2,372 52,506,753 24,755 2,894 21,011,449
    1,329,242 1,311,244  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

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

Ir          I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
         DLmw

-- 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          .    .    .          .      .    .          .          .          .      /* 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           .     .     .           .       .     .           .
                 .          .         perror("");
112           .     .     .           .       .     .           .
                 .          .         goto _exit_main;
113           .     .     .           .       .     .           .
                 .          .       }
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116           .     .     .           .       .     .           .
                 .          .       }
117           .     .     .           .       .     .           .
                 .          .       bs = (size_t) l;
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          .     .     .           .       .     .           .
                 .           .           goto _exit_main;
159          .     .     .           .       .     .           .
                 .           .         }
```

```
160          7     1     1           4       0     0           1
                 0           0         b->array[i] = e;
161          .     .     .           .       .     .           .
                 .           .       }
162          .     .     .           .       .     .           .
                 .           .
163          8     1     1           2       0     0           0
                 0           0       clock_gettime(CLOCK_REALTIME, &t0
                 );
164          .     .     .           .       .     .           .
                 .           .
165          .     .     .           .       .     .           .
                 .           .       /* multiply matrixes */
166         13     2     2           6       1     1           1
                 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", (
                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          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           .     .     .           .       .     .           .
                .           .   #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  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          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           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 16   2    2          5      0    0          2
        0          0    for(i=0; i<n; i++)
310 16   2    2          5      0    0          2
        0          0      for (j=0; j<n; j++)
311 22   3    3          10     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
322 --------------------------------------------------------------------------------
323 Ir           I1mr ILmr Dr          D1mr    DLmr Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------
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%     +
```

```
    6.3%  )
==601== LLd miss rate:          1.8% (       0.0%     +
    6.2%  )
==601==
==601== LL refs:          1,356,884  (    27,554 rd   +
    1,329,330 wr)
==601== LL misses:        1,316,528  (     5,281 rd   +
    1,311,247 wr)
==601== 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.601
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
--------------------------------------------------------------------------------

147,109,308 2,401 2,383 52,509,330 25,153 2,898 21,012,702
    1,329,330 1,311,247  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
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           .    .    .          .      .    .          .
                .          .      /* 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             .     .    .        .       .    .          .
                .          .         perror("");
112             .     .    .        .       .    .          .
                .          .         goto _exit_main;
113             .     .    .        .       .    .          .
                .          .       }
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116             .     .    .        .       .    .          .
                .          .       }
117             .     .    .        .       .    .          .
                .          .       bs = (size_t) l;
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         50    3    3            19       0    0          5
                0          0       for (i=0; i < n*n; i++) {
132          8    0    0             4       0    0          4
                 0          0         nptr = endptr;
133         36    1    1            12       0    0          4
                0          0         e = strtod(nptr, &endptr);
134         32    1    1            12       0    0          0
                0          0         if (errno) {
```

```
135          .     .     .           .       .     .           .
                 .           .           perror("");
136          .     .     .           .       .     .           .
                 .           .           goto _exit_main;
137          .     .     .           .       .     .           .
                 .           .         }
138         16     2     2           8       0     0           0
             0           0          if (nptr == endptr) {
139          .     .     .           .       .     .           .
                 .           .           fprintf(stderr, "missing A
             matrix element");
140          .     .     .           .       .     .           .
                 .           .           goto _exit_main;
141          .     .     .           .       .     .           .
                 .           .         }
142         28     1     1          16       0     0           4
             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         50     2     2          19       0     0           5
             0           0        for (i=0; i < n*n; i++) {
150          8     1     1           4       0     0           4
              0           0         nptr = endptr;
151         36     1     1          12       0     0           4
             0           0          e = strtod(nptr, &endptr);
152         32     1     1          12       0     0           0
             0           0          if (errno) {
153          .     .     .           .       .     .           .
                 .           .           perror("");
154          .     .     .           .       .     .           .
                 .           .           goto _exit_main;
155          .     .     .           .       .     .           .
                 .           .         }
156         16     1     1           8       0     0           0
             0           0          if (nptr == endptr) {
157          .     .     .           .       .     .           .
                 .           .           fprintf(stderr, "missing B
             matrix element");
158          .     .     .           .       .     .           .
                 .           .           goto _exit_main;
159          .     .     .           .       .     .           .
                 .           .         }
160         28     1     1          16       0     0           4
             1           1          b->array[i] = e;
```

```
.    .    .    .    .    .    .    .    .    }
.    .    .    .    .    .    .    .    .
8    1    1    2    0    0    0    0    0    clock_gettime(CLOCK_REALTIME, &t0);
.    .    .    .    .    .    .    .    .
.    .    .    .    .    .    .    .    .    /* multiply matrixes */
13   2    2    6    1    1    1    0    0    if (!(c = matrix_multiply(a, b, bs)))
.    .    .    .    .    .    .    .    .        goto _exit_main;
.    .    .    .    .    .    .    .    .
8    1    1    2    0    0    0    0    0    clock_gettime(CLOCK_REALTIME, &t1);
.    .    .    .    .    .    .    .    .
7    1    1    2    1    1    1    1    1    dt = (float) (t1.tv_sec - t0.tv_sec);
12   1    1    5    2    2    1    0    0    dt = dt + ((float)(t1.tv_nsec - t0.tv_nsec)) / 1.0e9;
.    .    .    .    .    .    .    .    .
13   3    2    5    2    2    0    0    0    if(print_matrix(stdout, c) == -1)
.    .    .    .    .    .    .    .    .        goto _exit_main;
.    .    .    .    .    .    .    .    .
12   1    1    7    0    0    0    0    0    fprintf(stderr, "time: %g\n", dt);
.    .    .    .    .    .    .    .    .
6    1    1    3    0    0    0    0    0    free(line);
6    1    1    3    0    0    0    0    0    destroy_matrix(a);
6    1    1    3    0    0    0    0    0    destroy_matrix(b);
6    0    0    3    0    0    0    0    0    destroy_matrix(c);
.    .    .    .    .    .    .    .    .    }
.    .    .    .    .    .    .    .    .
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          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           .     .     .           .       .     .           .
                .          .   #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  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          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           .     .     .           .       .     .           .
                .          .   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         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           .     .     .           .       .     .           .
                .           .        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           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            24     2     2           8       0     0           3
                0           0    for(i=0; i<n; i++)
310            48     2     2          16       0     0           6
                0           0      for (j=0; j<n; j++)
311            88     3     3          40       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
322 --------------------------------------------------------------------------------
323 Ir           I1mr ILmr Dr          D1mr    DLmr Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------
325 146,802,968  120  119 52,429,703 20,500   14 20,971,776
    1,328,644 1,310,722  events annotated
```

```
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
       Nethercote et al.
4  ==605== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
       copyright info
5  ==605== Command: /tmp/02-mmult
6  ==605== Parent PID: 597
7  ==605==
8  --605-- Warning: Cannot auto-detect cache config, using
       defaults.
9  --605--          Run with -v to see.
10 ==605==
11 ==605== I   refs:      147,126,965
12 ==605== I1  misses:          2,407
13 ==605== LLi misses:          2,389
14 ==605== I1  miss rate:        0.00%
15 ==605== LLi miss rate:        0.00%
16 ==605==
17 ==605== D   refs:       73,528,535  (52,513,741 rd   +
       21,014,794 wr)
18 ==605== D1  misses:      1,354,211  (    24,912 rd   +
       1,329,299 wr)
19 ==605== LLd misses:      1,314,153  (     2,901 rd   +
       1,311,252 wr)
20 ==605== D1  miss rate:         1.8% (       0.0%     +
       6.3%  )
21 ==605== LLd miss rate:         1.8% (       0.0%     +
       6.2%  )
22 ==605==
23 ==605== LL refs:         1,356,618  (    27,319 rd   +
       1,329,299 wr)
24 ==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              .     .     .           .       .     .           .
                .           .   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      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           .     .     .           .       .     .           .
                 .          .         perror("");
112           .     .     .           .       .     .           .
                 .          .         goto _exit_main;
113           .     .     .           .       .     .           .
                 .          .       }
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116           .     .     .           .       .     .           .
                 .          .       }
117           .     .     .           .       .     .           .
                 .          .       bs = (size_t) l;
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         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          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, &t0
                );
```

```
164 .    .    .          .      .    .          .
        .          .
165 .    .    .          .      .    .          .
        .          .      /* multiply matrixes */
166 13   2    2          6      1    1          1
    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", (
```

```
                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          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 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           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             .     .     .           .       .     .           .
                .           .   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         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           .    .    .          .      .    .          .
                .          .  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       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
                0          0    return 0;
320          6    1    1          2      0    0          0
                0          0  }
321
322 --------------------------------------------------------------------------------

323 Ir           I1mr ILmr Dr          D1mr    DLmr Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------

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
```

```
LL cache:          524288 B, 32 B, 8-way associative
Command:           /tmp/02-mmult
Data file:         cachegrind.out.611
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
--------------------------------------------------------------------------------
147,153,617 2,405 2,387 52,520,450 24,800 2,907 21,017,968
    1,329,245 1,311,259  PROGRAM TOTALS

--------------------------------------------------------------------------------
Ir           I1mr ILmr Dr          D1mr    DLmr Dw          D1mw
         DLmw        file:function
--------------------------------------------------------------------------------
146,804,064   29   29 52,430,421 20,496   16 20,971,794
    1,328,643 1,310,723  /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);
         .    .    .           .       .    .           .
             .          .
         .    .    .           .       .    .           .
             .          .  int print_matrix(FILE* fp, matrix_t*
             m);
         .    .    .           .       .    .           .
             .          .
         .    .    .           .       .    .           .
```

```
                .           .   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          .    .    .             .       .    .           .
                .          .       /* 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         .    .    .             .       .    .           .
                .          .         perror("");
112         .    .    .             .       .    .           .
                .          .         goto _exit_main;
113         .    .    .             .       .    .           .
                .          .       }
114 -- line 75 ----------------------------------------
```

```
115 -- line 83 ----------------------------------------
116           .     .     .           .       .     .           .
                .           .       }
117           .     .     .           .       .     .           .
                .           .       bs = (size_t) l;
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         170     3     3          67       0     0          17
                0           0       for (i=0; i < n*n; i++) {
132          32     0     0          16       0     0          16
                0           0         nptr = endptr;
133         144     1     1          48       0     0          16
                0           0         e = strtod(nptr, &endptr);
134         128     1     1          48       0     0           0
                0           0         if (errno) {
135           .     .     .           .       .     .           .
                .           .           perror("");
136           .     .     .           .       .     .           .
                .           .           goto _exit_main;
137           .     .     .           .       .     .           .
                .           .         }
138          64     2     2          32       0     0           0
                0           0         if (nptr == endptr) {
139           .     .     .           .       .     .           .
                .           .           fprintf(stderr, "missing A
              matrix element");
140           .     .     .           .       .     .           .
                .           .           goto _exit_main;
141           .     .     .           .       .     .           .
```

```
                .           .           }
142         112     1     1          64       0     0          16
                3           3          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         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           .     .     .           .       .     .           .
                .           .              perror("");
154           .     .     .           .       .     .           .
                .           .              goto _exit_main;
155           .     .     .           .       .     .           .
                .           .            }
156          64     1     1          32       0     0           0
                0           0          if (nptr == endptr) {
157           .     .     .           .       .     .           .
                .           .              fprintf(stderr, "missing B
                matrix element");
158           .     .     .           .       .     .           .
                .           .              goto _exit_main;
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           .     .     .           .       .     .           .
                .           .          /* multiply matrixes */
166          13     2     2           6       1     1           1
                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", (
                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         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          .     .     .           .       .     .           .
                .           .     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 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
                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       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           .     .     .           .       .     .           .
                .           .      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          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
322 ------------------------------------------------------------------------------
```

```
Ir           I1mr ILmr Dr          D1mr    DLmr Dw          D1mw
         DLmw
--------------------------------------------------------------------------------

146,808,382  120  119 52,432,069 20,509   23 20,972,236
    1,328,655 1,310,733  events annotated
```

```
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
017, and GNU GPL'd, by Nicholas Nethercote et al.
==614== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
==614== Command: /tmp/02-mmult
==614== Parent PID: 597
==614==
--614-- Warning: Cannot auto-detect cache config, using
    defaults.
--614--          Run with -v to see.
==614==
==614== I   refs:      147,187,039
==614== I1  misses:          2,407
==614== LLi misses:          2,389
==614== I1  miss rate:        0.00%
==614== LLi miss rate:        0.00%
==614==
==614== D   refs:       73,550,897  (52,528,979 rd   +
    21,021,918 wr)
==614== D1  misses:      1,354,198  (    24,881 rd   +
    1,329,317 wr)
==614== LLd misses:      1,314,183  (     2,915 rd   +
    1,311,268 wr)
==614== D1  miss rate:         1.8% (       0.0%     +
    6.3%  )
==614== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==614==
==614== LL refs:         1,356,605  (    27,288 rd   +
    1,329,317 wr)
==614== LL misses:       1,316,572  (     5,304 rd   +
    1,311,268 wr)
==614== 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.614
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
--------------------------------------------------------------------------------
147,187,039 2,407 2,389 52,528,979 24,881 2,915 21,021,918
    1,329,317 1,311,268  PROGRAM TOTALS

--------------------------------------------------------------------------------
Ir           I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
         DLmw       file:function
--------------------------------------------------------------------------------
146,806,731   29   29 52,431,722 20,503   23 20,971,983
    1,328,645 1,310,725  /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);
          .    .    .           .      .    .           .
              .          .
          .    .    .           .      .    .           .
              .          .  int print_matrix(FILE* fp, matrix_t*
              m);
          .    .    .           .      .    .           .
              .          .
          .    .    .           .      .    .           .
              .          .  int
          .    .    .           .      .    .           .
              .          .  main(int argc, char** argv)
         10    2    2           0      0    0           5
            0          0  {
          .    .    .           .      .    .           .
              .          .    /* matrixes */
          .    .    .           .      .    .           .
```

```
                .           .     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          .     .     .           .       .     .           .
                .           .     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    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) l;
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) l;
117         .    .    .          .      .    .          .
                .          .
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           1
                0           0       bs = n;
124             .     .     .           .       .     .           .
                .           .   #endif
125             .     .     .           .       .     .           .
                .           .
126             .     .     .           .       .     .           .
                .           .       /* load matrix a */
127          11     1     1           5       1     0           1
                0           0       if (!(a = create_matrix(n, n)))
128             .     .     .           .       .     .           .
                .           .         goto _exit_main;
129             .     .     .           .       .     .           .
                .           .
130         260     3     3         103       0     0          26
                0           0       for (i=0; i < n*n; i++) {
131          50     0     0          25       0     0          25
                0           0         nptr = endptr;
132         225     1     1          75       0     0          25
                0           0         e = strtod(nptr, &endptr);
133         200     1     1          75       0     0           0
                0           0         if (errno) {
134             .     .     .           .       .     .           .
                .           .           perror("");
135             .     .     .           .       .     .           .
                .           .           goto _exit_main;
136             .     .     .           .       .     .           .
                .           .         }
137         100     2     2          50       0     0           0
                0           0         if (nptr == endptr) {
138             .     .     .           .       .     .           .
                .           .           fprintf(stderr, "missing A
                matrix element");
139             .     .     .           .       .     .           .
                .           .           goto _exit_main;
140             .     .     .           .       .     .           .
                .           .         }
141         175     1     1         100       0     0          25
                5           5         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          260     2     2         103       0     0          26
                 0           0       for (i=0; i < n*n; i++) {
149           50     1     1          25       0     0          25
                 0           0         nptr = endptr;
150          225     1     1          75       0     0          25
                 0           0         e = strtod(nptr, &endptr);
151          200     1     1          75       0     0           0
                 0           0         if (errno) {
152           .     .     .           .       .     .           .
                .           .           perror("");
153           .     .     .           .       .     .           .
                .           .           goto _exit_main;
154           .     .     .           .       .     .           .
                .           .         }
155          100     1     1          50       0     0           0
                 0           0         if (nptr == endptr) {
156           .     .     .           .       .     .           .
                .           .           fprintf(stderr, "missing B
                matrix element");
157           .     .     .           .       .     .           .
                .           .           goto _exit_main;
158           .     .     .           .       .     .           .
                .           .         }
159          175     1     1         100       0     0          25
                 5           5         b->array[i] = e;
160           .     .     .           .       .     .           .
                .           .       }
161           .     .     .           .       .     .           .
                .           .
162           8     1     1           2       0     0           0
                 0           0       clock_gettime(CLOCK_REALTIME, &t0
                );
163           .     .     .           .       .     .           .
                .           .
164           .     .     .           .       .     .           .
                .           .       /* multiply matrixes */
165          13     2     2           6       1     1           1
                 0           0       if (!(c = matrix_multiply(a, b, bs
                )))
166           .     .     .           .       .     .           .
                .           .         goto _exit_main;
167           .     .     .           .       .     .           .
                .           .
168           8     1     1           2       0     0           0
                 0           0       clock_gettime(CLOCK_REALTIME, &t1
                );
169           .     .     .           .       .     .           .
```

```
                .           .
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   }
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           .     .     .           .       .     .           .
                .           .    double sum;
200           .     .     .           .       .     .           .
                .           .    matrix_t* mr;
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           .     .     .           .       .     .           .
                .           .  #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  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          .     .     .           .       .     .           .
                .           .   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             .     .     .           .       .     .           .
                .           .           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           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) m->rows) < 0) {
305            .     .     .           .       .     .           .
                  .          .       perror("");
306            .     .     .           .       .     .           .
                  .          .       return -1;
307            .     .     .           .       .     .           .
                  .          .     }
308           48     2     2          17       0     0           6
                 0          0     for(i=0; i<n; i++)
309          240     2     2          85       0     0          30
                 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     if (fprintf(fp, "\n") < 0) {
315            .     .     .           .       .     .           .
                  .          .       perror("");
316            .     .     .           .       .     .           .
                  .          .       return -1;
317            .     .     .           .       .     .           .
                  .          .     }
318            1     1     1           0       0     0           0
                 0          0     return 0;
319            6     1     1           2       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
```

```
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
e et al.
==617== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
==617== Command: /tmp/02-mmult
==617== Parent PID: 597
==617==
--617-- Warning: Cannot auto-detect cache config, using
    defaults.
--617--          Run with -v to see.
==617==
==617== I   refs:      147,225,930
==617== I1  misses:          2,407
==617== LLi misses:          2,389
==617== I1  miss rate:        0.00%
==617== LLi miss rate:        0.00%
==617==
==617== D   refs:       73,565,860  (52,539,342 rd   +
    21,026,518 wr)
==617== D1  misses:      1,354,213  (    24,897 rd   +
    1,329,316 wr)
==617== LLd misses:      1,314,203  (     2,925 rd   +
    1,311,278 wr)
==617== D1  miss rate:         1.8% (       0.0%     +
    6.3%  )
==617== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==617==
==617== LL refs:         1,356,620  (    27,304 rd   +
    1,329,316 wr)
==617== LL misses:       1,316,592  (     5,314 rd   +
    1,311,278 wr)
==617== 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.617
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
41 --------------------------------------------------------------------------------

42 147,225,930 2,407 2,389 52,539,342 24,897 2,925 21,026,518
       1,329,316 1,311,278  PROGRAM TOTALS
43
44 --------------------------------------------------------------------------------

45 Ir           I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
         DLmw       file:function
46 --------------------------------------------------------------------------------

47 146,810,542   29   29 52,433,589 20,511   31 20,972,246
       1,328,648 1,310,728  /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          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          .     .     .           .       .     .           .
                .           .    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   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) l;
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) l;
117         .   .   .       .       .   .   .       .
                .           .
118         .   .   .       .       .   .   .       .
                .           .       if (n % bs) {
119         .   .   .       .       .   .   .       .
                .           .           fprintf(stderr, "block size
            doesn't match");
120         .   .   .       .       .   .   .       .
                .           .           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++) {
 72   0  0   36  0  0  36  0  0            nptr = endptr;
324   1  1  108  0  0  36  0  0            e = strtod(nptr, &endptr);
288   1  1  108  0  0   0  0  0            if (errno) {
  .   .  .    .  .  .   .  .  .                perror("");
  .   .  .    .  .  .   .  .  .                goto _exit_main;
  .   .  .    .  .  .   .  .  .            }
144   2  2   72  0  0   0  0  0            if (nptr == endptr) {
  .   .  .    .  .  .   .  .  .                fprintf(stderr, "missing A
        matrix element");
  .   .  .    .  .  .   .  .  .                goto _exit_main;
  .   .  .    .  .  .   .  .  .            }
252   1  1  144  0  0  36  8  8            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;
  .   .  .    .  .  .   .  .  .
```

```
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         nptr = endptr;
150        324     1     1         108       0     0          36
               0           0         e = strtod(nptr, &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 (nptr == endptr) {
156          .     .     .           .       .     .           .
                   .           .           fprintf(stderr, "missing B
             matrix element");
157          .     .     .           .       .     .           .
                   .           .           goto _exit_main;
158          .     .     .           .       .     .           .
                   .           .         }
159        252     1     1         144       0     0          36
               9           9         b->array[i] = e;
160          .     .     .           .       .     .           .
                   .           .       }
161          .     .     .           .       .     .           .
                   .           .
162          8     1     1           2       0     0           0
                  0           0       clock_gettime(CLOCK_REALTIME, &t0
                );
163          .     .     .           .       .     .           .
                   .           .
164          .     .     .           .       .     .           .
                   .           .       /* multiply matrixes */
165         13     2     2           6       1     1           1
               0           0       if (!(c = matrix_multiply(a, b, bs
             )))
166          .     .     .           .       .     .           .
                   .           .         goto _exit_main;
167          .     .     .           .       .     .           .
                   .           .
168          8     1     1           2       0     0           0
                  0           0       clock_gettime(CLOCK_REALTIME, &t1
                );
169          .     .     .           .       .     .           .
                   .           .
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  }
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         .     .     .           .       .     .           .
                  .          .     double sum;
200         .     .     .           .       .     .           .
                  .          .     matrix_t* mr;
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        56     2     2          20       0     0           7
                  0          0     for(i=0; i<n; i++)
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  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          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           .     .     .           .       .     .           .                .          .          char*
236           .     .     .           .       .     .           .                .          .          read_line(FILE *fp)
237          18     3     3           0       0     0           8                0          0          {
238           .     .     .           .       .     .           .                .          .          #define DEF_LINE_SZ 1024
239           .     .     .           .       .     .           .                .          .
240           .     .     .           .       .     .           .                .          .
241           6     0     0           0       0     0           4                0          0            int c;
242           .     .     .           .       .     .           .                .          .            size_t len = 0, tam = DEF_LINE_SZ;
243           .     .     .           .       .     .           .                .          .            char* str;
244          14     1     1           6       1     0           2                1          0
245           6     0     0           2       0     0           0                0          0            str = malloc(tam);
246           .     .     .           .       .     .           .                .          .            if (!str) {
247           .     .     .           .       .     .           .                .          .              perror("");
248           .     .     .           .       .     .           .                .          .              return NULL;
249           .     .     .           .       .     .           .                .          .            }
250       2,201     5     5         734       1     0         147                0          0
```

```
         0          0    while (EOF != (c=fgetc(fp)) && c != '\n
        ') {
251   1,160     2    2         435       0    0         290
          4          4      str[len++]=c;
252     725     0    0         290       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           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) m->rows) < 0) {
305           .     .     .          .       .     .           .
                 .           .       perror("");
306           .     .     .          .       .     .           .
                 .           .       return -1;
307           .     .     .          .       .     .           .
                 .           .     }
308           56    2     2          20      0     0           7
                 0           0     for(i=0; i<n; i++)
309           336   2     2          120     0     0           42
                 0           0       for (j=0; j<n; j++)
310           792   3     3          360     40    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     if (fprintf(fp, "\n") < 0) {
315           .     .     .          .       .     .           .
                 .           .       perror("");
316           .     .     .          .       .     .           .
                 .           .       return -1;
317           .     .     .          .       .     .           .
                 .           .     }
318           1     1     1          0       0     0           0
                 0           0     return 0;
319           6     1     1          2       0     0           0
                 0           0   }
320
321 --------------------------------------------------------------------------------

322 Ir           I1mr  ILmr  Dr          D1mr    DLmr  Dw          D1mw
          DLmw
323 --------------------------------------------------------------------------------

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==
```

```
--620-- Warning: Cannot auto-detect cache config, using defaults.
--620--          Run with -v to see.
==620==
==620== I   refs:      147,273,130
==620== I1  misses:          2,405
==620== LLi misses:          2,387
==620== I1  miss rate:        0.00%
==620== LLi miss rate:        0.00%
==620==
==620== D   refs:       73,583,952  (52,551,913 rd   + 21,032,039 wr)
==620== D1  misses:      1,354,741  (    25,330 rd   + 1,329,411 wr)
==620== LLd misses:      1,314,228  (     2,936 rd   + 1,311,292 wr)
==620== D1  miss rate:         1.8% (       0.0%     + 6.3%  )
==620== LLd miss rate:         1.8% (       0.0%     + 6.2%  )
==620==
==620== LL refs:         1,357,146  (    27,735 rd   + 1,329,411 wr)
==620== LL misses:       1,316,615  (     5,323 rd   + 1,311,292 wr)
==620== 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.620
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
--------------------------------------------------------------------------------

147,273,130 2,405 2,387 52,551,913 25,330 2,936 21,032,039
    1,329,411 1,311,292  PROGRAM TOTALS

--------------------------------------------------------------------------------

Ir           I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
```

```
         DLmw       file:function
--------------------------------------------------------------------------------

146,815,701   29   29 52,436,124 20,521   41 20,972,595
   1,328,651 1,310,731  /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);
          .     .     .           .       .     .           .
              .          .
          .     .     .           .       .     .           .
              .          .  int print_matrix(FILE* fp, matrix_t*
             m);
          .     .     .           .       .     .           .
              .          .
          .     .     .           .       .     .           .
              .          .  int
          .     .     .           .       .     .           .
              .          .  main(int argc, char** argv)
         10     2     2           0       0     0           5
             0          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) */
          1     0     0           0       0     0           1
             0          0    char *line = NULL;
          .     .     .           .       .     .           .
              .          .    /* line parsing auxiliar pointers
             */
          .     .     .           .       .     .           .
              .          .    char *nptr, *endptr;
```

```
71          .     .     .          .       .     .          .
                .          .     /* auxiliar variables */
72          .     .     .          .       .     .          .
                .          .     long l;
73          .     .     .          .       .     .          .
                .          .     double e;
74          2     1     1          0       0     0          1
                0          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          0     for(; !feof(stdin); lineno++) {
81         10     1     1          4       0     0          6
             0          0       a=b=c=NULL;
82          .     .     .          .       .     .          .
                .          .
83         18     3     3          8       1     0          2
             0          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          .     .     .          .       .     .          .
                .          .       /* parse dimension */
89          2     1     1          1       0     0          1
                1          0       nptr = line;
90         10     1     1          3       1     0          1
             0          0       l = strtol(nptr, &endptr, 10);
91          8     1     1          3       0     0          0
                0          0       if (errno) {
92          .     .     .          .       .     .          .
                .          .         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         .   .   .       .       .   .       .
                .           .   #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) 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         .   .   .       .       .   .       .
                .           .   #endif
124         .   .   .       .       .   .       .
```

```
                .           .
125           .     .     .           .       .     .           .
                .           .       /* load matrix a */
126          11     1     1           5       1     0           1
                0           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
              matrix element");
138           .     .     .           .       .     .           .
                .           .           goto _exit_main;
139           .     .     .           .       .     .           .
                .           .         }
140         343     1     1         196       0     0          49
               11          11         a->array[i] = e;
141           .     .     .           .       .     .           .
                .           .       }
142           .     .     .           .       .     .           .
                .           .
143           .     .     .           .       .     .           .
                .           .       /* load matrix b */
144          11     1     1           5       0     0           1
                0           0       if (!(b = create_matrix(n, n)))
145           .     .     .           .       .     .           .
                .           .         goto _exit_main;
146           .     .     .           .       .     .           .
                .           .
147         500     2     2         199       0     0          50
                0           0       for (i=0; i < n*n; i++) {
148          98     1     1          49       0     0          49
                0           0         nptr = endptr;
149         441     1     1         147       0     0          49
                0           0         e = strtod(nptr, &endptr);
150         392     1     1         147       0     0           0
                0           0         if (errno) {
```

```
151  .    .    .          .      .    .          .
         .          .          perror("");
152  .    .    .          .      .    .          .
         .          .          goto _exit_main;
153  .    .    .          .      .    .          .
         .          .        }
154  196    1    1         98      0    0          0
       0          0        if (nptr == endptr) {
155  .    .    .          .      .    .          .
         .          .          fprintf(stderr, "missing B
         matrix element");
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          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           .    .    .          .      .    .          .
                .          .    destroy_matrix(b);
190           .    .    .          .      .    .          .
                .          .    destroy_matrix(c);
191           .    .    .          .      .    .          .
                .          .    exit(1);
192           6    1    1          2      1    0          0
                0          0  }
193           .    .    .          .      .    .          .
                .          .
194           .    .    .          .      .    .          .
                .          .  matrix_t*
195           .    .    .          .      .    .          .
                .          .  matrix_multiply(matrix_t* m1,
                matrix_t* m2, int bs)
196          11    2    2          0      0    0          6
                0          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           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           .     .     .           .       .     .           .
                  .           .   #if 1
212           .     .     .           .       .     .           .
                  .           .     if (1) {
213           .     .     .           .       .     .           .
                  .           .       size_t j;
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           .     .     .           .       .     .           .
                  .           .   #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      .      .      .            .        .      .            .
                 .          .    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           0       if (c != EOF)
261             7     0     0           2       0     0           2
                0           0         str[len++]='\n';
262             .     .     .           .       .     .           .
                .           .
263            12     1     1           4       0     0           4
               0            0       str[len++]='\0';
264             2     0     0           2       0     0           0
                0           0       return str;
265            12     1     1           4       0     0           0
               0            0   }
266             .     .     .           .       .     .           .
                .           .
267             .     .     .           .       .     .           .
                .           .
268             .     .     .           .       .     .           .
                .           .   matrix_t*
269             .     .     .           .       .     .           .
                .           .   create_matrix(size_t rows, size_t
                 cols)
270            30     1     1           0       0     0          15
               0            0   {
271             .     .     .           .       .     .           .
                .           .     matrix_t * m;
272             .     .     .           .       .     .           .
                .           .
273            30     2     2           9       1     0           3
               0            0     if (!(m = malloc(sizeof(matrix_t))))
                 {
274             .     .     .           .       .     .           .
                .           .       perror("");
275             .     .     .           .       .     .           .
                .           .       return NULL;
276             .     .     .           .       .     .           .
                .           .     }
277             .     .     .           .       .     .           .
                .           .
278             9     1     1           6       0     0           3
                0           0     m->rows = rows;
279             9     1     1           6       0     0           3
```

```
                0          0    m->cols = cols;
280         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  {
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  {
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          .    .    .           .       .    .          .
```

```
              .          .       return -1;
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     49    0          0
            0          0         if (fprintf(fp, " %g", m->array[i*n
           +j]) < 0) {
310           .    .    .          .       .    .          .
              .          .           perror("");
311           .    .    .          .       .    .          .
              .          .           return -1;
312           .    .    .          .       .    .          .
              .          .         }
313         10    1    1          4      0    0          0
               0          0    if (fprintf(fp, "\n") < 0) {
314           .    .    .          .       .    .          .
              .          .       perror("");
315           .    .    .          .       .    .          .
              .          .       return -1;
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  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%
```

```
==623==
==623== D   refs:       73,608,453  (52,568,667 rd   + 21,039,786 wr)
==623== D1  misses:      1,354,925  (    25,352 rd   + 1,329,573 wr)
==623== LLd misses:      1,314,257  (     2,949 rd   + 1,311,308 wr)
==623== D1  miss rate:         1.8% (       0.0%     + 6.3%  )
==623== LLd miss rate:         1.8% (       0.0%     + 6.2%  )
==623==
==623== LL refs:         1,357,329  (    27,756 rd   + 1,329,573 wr)
==623== LL misses:       1,316,641  (     5,333 rd   + 1,311,308 wr)
==623== 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.623
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
--------------------------------------------------------------------------------

147,337,268 2,404 2,384 52,568,667 25,352 2,949 21,039,786
    1,329,573 1,311,308  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------
```

```
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          .    .    .           .       .    .           .
               .          .    /* 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           1
               0          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           1
               0          0    size_t lineno = 1;
73          .    .    .           .       .    .           .
```

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

```
                .           .         goto _exit_main;
100           .     .     .           .       .     .           .
                .           .       }
101           2     1     1           1       0     0           1
                0           0       n = (size_t) l;
102           .     .     .           .       .     .           .
                .           .   #if 0
103           .     .     .           .       .     .           .
                .           .       /* parse block size */
104           .     .     .           .       .     .           .
                .           .       nptr = endptr;
105           .     .     .           .       .     .           .
                .           .       l = strtol(nptr, &endptr, 10);
106           .     .     .           .       .     .           .
                .           .       if (errno) {
107           .     .     .           .       .     .           .
                .           .         perror("");
108           .     .     .           .       .     .           .
                .           .         goto _exit_main;
109           .     .     .           .       .     .           .
                .           .       }
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           0       bs = n;
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           0       for (i=0; i < n*n; i++) {
128        128     0     0           64       0     0          64
               0           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           0       if (!(b = create_matrix(n, n)))
143          .     .     .            .       .     .           .
                 .           .         goto _exit_main;
144          .     .     .            .       .     .           .
                 .           .
145        650     2     2          259       0     0          65
               0           0       for (i=0; i < n*n; i++) {
146        128     1     1           64       0     0          64
               0           0         nptr = endptr;
147        576     1     1          192       0     0          64
               0           0         e = strtod(nptr, &endptr);
148        512     1     1          192       0     0           0
               0           0         if (errno) {
149          .     .     .            .       .     .           .
                 .           .           perror("");
150          .     .     .            .       .     .           .
                 .           .           goto _exit_main;
151          .     .     .            .       .     .           .
                 .           .         }
152        256     1     1          128       0     0           0
               0           0         if (nptr == endptr) {
153          .     .     .            .       .     .           .
```

```
                .           .           fprintf(stderr, "missing B
                matrix element");
154             .     .     .           .       .     .           .
                .           .           goto _exit_main;
155             .     .     .           .       .     .           .
                .           .         }
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             .     .     .           .       .     .           .
                .           .       /* multiply matrixes */
162            13     2     2           6       1     1           1
                0           0       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
```

```
                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* m2, int bs)
194         11     2     2           0       0     0           6
            0          0   {
195           .    .     .           .       .     .           .
                .          .     size_t n, en, i, j, k, kk, jj;
196           .    .     .           .       .     .           .
                .          .     double sum;
197           .    .     .           .       .     .           .
                .          .     matrix_t* mr;
198           .    .     .           .       .     .           .
                .          .
199           3    1     1           2       0     0           1
                0          0     n = m1->rows;
200           .    .     .           .       .     .           .
                .          .
201         11     1     1           5       0     0           1
            0          0     if(!(mr = create_matrix(n,n)))
              return NULL;
202           .    .     .           .       .     .           .
                .          .
```

```
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
```

```
              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 .     .     .           .       .     .           .
        .           .       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           0   {
290           9     0     0           3       0     0           0
                0           0     if (!m) return;
291          24     1     1          12       0     0           0
                0           0     free(m->array);
292          24     1     1           9       0     0           0
                0           0     free(m);
293          18     0     0           6       0     0           0
                0           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   121   119 52,445,273 20,545    59 20,974,500
    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)
```

```
==626== D1  misses:       1,315,006  (     3,709 rd   +     1,311,297 wr)
==626== LLd misses:       1,314,138  (     2,894 rd   +     1,311,244 wr)
==626== D1  miss rate:          1.8% (       0.0%     +           6.2%  )
==626== LLd miss rate:          1.8% (       0.0%     +           6.2%  )
==626==
==626== LL refs:          1,317,396  (     6,099 rd   +     1,311,297 wr)
==626== LL misses:        1,316,510  (     5,266 rd   +     1,311,244 wr)
==626== LL miss rate:           0.6% (       0.0%     +           6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.626
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
--------------------------------------------------------------------------------

147,098,977 2,390 2,372 52,506,759 3,709 2,894 21,011,454
    1,311,297 1,311,244  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
          .     .     .          .     .     .          .
                    .
          .     .     .          .     .     .          .
                    .  int print_matrix(FILE* fp, matrix_t* m)
              ;
          .     .     .          .     .     .          .
                    .
          .     .     .          .     .     .          .
                    .  int
          .     .     .          .     .     .          .
                    .  main(int argc, char** argv)
         10     2     2          0     0     0          5          0
                    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) */
          1     0     0          0     0     0          1          0
                    0    char *line = NULL;
          .     .     .          .     .     .          .
                    .    /* line parsing auxiliar pointers */
          .     .     .          .     .     .          .
                    .    char *nptr, *endptr;
          .     .     .          .     .     .          .
                    .    /* auxiliar variables */
          .     .     .          .     .     .          .
                    .    long l;
          .     .     .          .     .     .          .
                    .    double e;
          2     1     1          0     0     0          1          0
                    0    size_t lineno = 1;
          .     .     .          .     .     .          .
                    .    struct timespec t0;
          .     .     .          .     .     .          .
                    .    struct timespec t1;
```

```
      .     .     .          .     .     .          .          .
                   .     double dt;
      .     .     .          .     .     .          .          .
                   .     size_t i;
      .     .     .          .     .     .          .          .
                   .
     25     4     4          9     0     0          1          0
                  0     for(; !feof(stdin); lineno++) {
     10     1     1          4     0     0          6          0
                  0       a=b=c=NULL;
      .     .     .          .     .     .          .          .
                   .
     18     3     3          8     1     0          2          0
                  0       line = read_line(stdin);
      .     .     .          .     .     .          .          .
                   .
      6     0     0          2     0     0          0          0
                   0       if (!line) goto _exit_main;
      9     0     0          4     0     0          0          0
                   0       if (line[0] == 0) break;
      .     .     .          .     .     .          .          .
                   .
      .     .     .          .     .     .          .          .
                   .       /* parse dimension */
      2     1     1          1     0     0          1          1
                   0       nptr = line;
     10     1     1          3     1     0          1          0
                  0       l = strtol(nptr, &endptr, 10);
      8     1     1          3     0     0          0          0
                   0       if (errno) {
      .     .     .          .     .     .          .          .
                   .         perror("");
      .     .     .          .     .     .          .          .
                   .         goto _exit_main;
      .     .     .          .     .     .          .          .
                   .       }
      4     2     2          2     0     0          0          0
                   0       if (nptr == endptr) {
      .     .     .          .     .     .          .          .
                   .         fprintf(stderr, "missing
    dimension");
      .     .     .          .     .     .          .          .
                   .         goto _exit_main;
      .     .     .          .     .     .          .          .
                   .       }
      3     2     2          1     0     0          0          0
                   0       if (l < 1) {
      .     .     .          .     .     .          .          .
                   .         fprintf(stderr, "invalid
    dimension");
      .     .     .          .     .     .          .          .
                   .         goto _exit_main;
      .     .     .          .     .     .          .          .
                   .       }
```

```
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           .     .     .           .     .     .           .           .
                          .         perror("");
112           .     .     .           .     .     .           .           .
                          .         goto _exit_main;
113           .     .     .           .     .     .           .           .
                          .       }
114   -- line 75 ----------------------------------------
115   -- line 83 ----------------------------------------
116           .     .     .           .     .     .           .           .
                          .       }
117           .     .     .           .     .     .           .           .
                          .       bs = (size_t) l;
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");
```

```
 .    .    .    .    .    .    .    .    .         goto _exit_main;
 .    .    .    .    .    .    .    .    .       }
 7    1    1    0    4    0    0    1    0       b->array[i] = e;
 .    .    .    .    .    .    .    .    .     }
 .    .    .    .    .    .    .    .    .
 8    1    1    0    2    0    0    0    0     clock_gettime(CLOCK_REALTIME, &t0);
 .    .    .    .    .    .    .    .    .
 .    .    .    .    .    .    .    .    .     /* multiply matrixes */
13    2    2    0    6    1    1    1    0     if (!(c = matrix_multiply(a, b, bs)))
 .    .    .    .    .    .    .    .    .       goto _exit_main;
 .    .    .    .    .    .    .    .    .
 8    1    1    0    2    0    0    0    0     clock_gettime(CLOCK_REALTIME, &t1);
 .    .    .    .    .    .    .    .    .
 7    1    1    1    2    1    1    1    1     dt = (float) (t1.tv_sec - t0.tv_sec);
12    1    1    0    5    2    2    1    0     dt = dt + ((float)(t1.tv_nsec - t0.tv_nsec)) / 1.0e9;
 .    .    .    .    .    .    .    .    .
13    3    2    0    5    2    2    0    0     if(print_matrix(stdout, c) == -1)
 .    .    .    .    .    .    .    .    .       goto _exit_main;
 .    .    .    .    .    .    .    .    .
12    1    1    0    7    0    0    0    0     fprintf(stderr, "time: %g\n", dt);
 .    .    .    .    .    .    .    .    .
 6    1    1    0    3    0    0    0    0     free(line);
 6    1    1    0    3    0    0    0    0     destroy_matrix(a);
 6    1    1    0    3    0    0    0    0     destroy_matrix(b);
 6    0    0    0    3    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
                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 |          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 |          .     .     .           .     .     .           .           .
    |                        .   #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 |          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
                  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;
```

```
 .  .  .   .  .  .   .   .   .  }
 .  .  .   .  .  .   .   .   .
 3  1  1   3  0  0   0   0   0  return m;
18  1  1   6  0  0   0   0   0  }
 .  .  .   .  .  .   .   .   .
 .  .  .   .  .  .   .   .   .  void
 .  .  .   .  .  .   .   .   .  destroy_matrix(matrix_t* m)
27  1  1   0  0  0  12   0   0  {
 9  0  0   3  0  0   0   0   0    if (!m) return;
24  1  1  12  0  0   0   0   0    free(m->array);
24  1  1   9  0  0   0   0   0    free(m);
18  0  0   6  0  0   0   0   0  }
 .  .  .   .  .  .   .   .   .
 .  .  .   .  .  .   .   .   .  int
 .  .  .   .  .  .   .   .   .  print_matrix(FILE* fp, matrix_t* m)
10  2  2   0  0  0   5   0   0  {
 .  .  .   .  .  .   .   .   .    size_t i, j;
 .  .  .   .  .  .   .   .   .    size_t n;
 3  1  1   2  0  0   1   0   0    n = m->rows;
13  1  1   6  1  1   0   0   0    if (fprintf(fp, "%lu", (unsigned long)
                                      m->rows) < 0) {
 .  .  .   .  .  .   .   .   .      perror("");
 .  .  .   .  .  .   .   .   .      return -1;
 .  .  .   .  .  .   .   .   .    }
16  2  2   5  0  0   2   0   0    for(i=0; i<n; i++)
16  2  2   5  0  0   2   0   0      for (j=0; j<n; j++)
22  3  3  10  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
322 --------------------------------------------------------------------------------
323 Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------
325 146,801,827  120   119  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  ==629== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas
       Nethercote et al.
4  ==629== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
       copyright info
5  ==629== Command: /tmp/02-mmult
6  ==629== Parent PID: 597
7  ==629==
8  --629-- Warning: Cannot auto-detect cache config, using
       defaults.
9  --629--          Run with -v to see.
10 ==629==
11 ==629== I   refs:      147,109,392
12 ==629== I1  misses:          2,408
13 ==629== LLi misses:          2,388
14 ==629== I1  miss rate:        0.00%
15 ==629== LLi miss rate:        0.00%
16 ==629==
17 ==629== D   refs:       73,522,048  (52,509,341 rd   +
       21,012,707 wr)
18 ==629== D1  misses:      1,314,921  (     3,621 rd   +
       1,311,300 wr)
19 ==629== LLd misses:      1,314,145  (     2,898 rd   +
       1,311,247 wr)
20 ==629== D1  miss rate:         1.8% (       0.0%     +
```

```
    6.2%  )
==629== LLd miss rate:          1.8% (       0.0%     +
    6.2%  )
==629==
==629== LL refs:          1,317,329  (     6,029 rd   +
    1,311,300 wr)
==629== LL misses:        1,316,533  (     5,286 rd   +
    1,311,247 wr)
==629== LL miss rate:           0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.629
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
--------------------------------------------------------------------------------

147,109,392 2,408 2,388 52,509,341 3,621 2,898 21,012,707
    1,311,300 1,311,247  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
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           .    .    .          .    .    .          .          .
                         .      /* 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 |          .     .     .           .     .     .           .          .
    |                        .         perror("");
112 |          .     .     .           .     .     .           .          .
    |                        .         goto _exit_main;
113 |          .     .     .           .     .     .           .          .
    |                        .       }
114 | -- line 75 ----------------------------------------
115 | -- line 83 ----------------------------------------
116 |          .     .     .           .     .     .           .          .
    |                        .       }
117 |          .     .     .           .     .     .           .          .
    |                        .       bs = (size_t) l;
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 |         50     3     3          19     0     0           5          0
    |                        0       for (i=0; i < n*n; i++) {
132 |          8     0     0           4     0     0           4          0
    |                        0         nptr = endptr;
133 |         36     1     1          12     0     0           4          0
    |                        0        e = strtod(nptr, &endptr);
134 |         32     1     1          12     0     0           0          0
    |                        0        if (errno) {
135 |          .     .     .           .     .     .           .          .
```

```
                    .           perror("");
136       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
137       .     .     .           .     .     .           .           .
                    .         }
138     16    2     2           8     0     0           0           0
                    0         if (nptr == endptr) {
139       .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing A
              matrix element");
140       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
141       .     .     .           .     .     .           .           .
                    .         }
142     28    1     1           16    0     0           4           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     50    2     2           19    0     0           5           0
                    0       for (i=0; i < n*n; i++) {
150      8    1     1           4     0     0           4           0
                      0         nptr = endptr;
151     36    1     1           12    0     0           4           0
                    0         e = strtod(nptr, &endptr);
152     32    1     1           12    0     0           0           0
                    0         if (errno) {
153       .     .     .           .     .     .           .           .
                    .           perror("");
154       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
155       .     .     .           .     .     .           .           .
                    .         }
156     16    1     1           8     0     0           0           0
                    0         if (nptr == endptr) {
157       .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing B
              matrix element");
158       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
159       .     .     .           .     .     .           .           .
                    .         }
160     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         .   .   .           .   .   .           .           .
                    .       /* multiply matrixes */
166        13   2   2           6   1   1           1           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   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
                    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           .     .     .           .     .     .           .           .
                        .   #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
                        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           .     .     .           .     .     .           .           .
                        .   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   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  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  .   .   .   .   .   .   .   .   .   .   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
```

```
                    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 24    2     2           8     0     0           3           0
                0    for(i=0; i<n; i++)
310 48    2     2           16    0     0           6           0
                0      for (j=0; j<n; j++)
311 88    3     3           40    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
322 --------------------------------------------------------------------------------
323 Ir           I1mr ILmr Dr          D1mr DLmr Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------
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 --------------------------------------------------------------------------------
```

```
I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.632
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
--------------------------------------------------------------------------------
147,126,985 2,407 2,389 52,513,745 3,625 2,901 21,014,797
    1,311,305 1,311,252  PROGRAM TOTALS

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

--------------------------------------------------------------------------------
-- 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);
         .    .    .           .    .    .          .         .
                   .
         .    .    .           .    .    .          .         .
                    .  int print_matrix(FILE* fp, matrix_t* m)
             ;
         .    .    .           .    .    .          .         .
```

```
                    .
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          .     .     .           .     .     .           .          .
                    .       /* 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         .     .     .           .     .     .           .          .
                    .         perror("");
112         .     .     .           .     .     .           .          .
                    .         goto _exit_main;
113         .     .     .           .     .     .           .          .
                    .       }
```

```
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116           .     .     .           .     .     .           .           .
                    .       }
117           .     .     .           .     .     .           .           .
                    .       bs = (size_t) l;
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         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
                  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, &t0);
164 .     .     .           .     .     .           .           .
                  .
165 .     .     .           .     .     .           .           .
                  .       /* multiply matrixes */
166 13    2     2           6     1     1           1           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", (
    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         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         .     .     .           .     .     .           .         .
                    .   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         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           .     .     .           .     .     .           .           .
```

```
                    .  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     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           0
                         0    return 0;
320           6     1     1           2     0     0           0           0
                         0  }
321
322 ------------------------------------------------------------------------------
323 Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
          DLmw
324 ------------------------------------------------------------------------------
```

```
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 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 --------------------------------------------------------------------------------
43 147,153,677 2,407 2,387 52,520,455 3,630 2,907 21,017,971
    1,311,312 1,311,259  PROGRAM TOTALS
44
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
    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 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          .     .     .           .     .     .           .           .
                    .       /* 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         .    .    .           .    .    .          .          .
                     .        perror("");
112         .    .    .           .    .    .          .          .
                     .        goto _exit_main;
113         .    .    .           .    .    .          .          .
                     .      }
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116         .    .    .           .    .    .          .          .
                     .      }
117         .    .    .           .    .    .          .          .
                     .      bs = (size_t) l;
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         170     3     3          67     0     0          17           0
                            0       for (i=0; i < n*n; i++) {
132          32     0     0          16     0     0          16           0
                             0        nptr = endptr;
133         144     1     1          48     0     0          16           0
                            0        e = strtod(nptr, &endptr);
134         128     1     1          48     0     0           0           0
                            0        if (errno) {
135             .     .     .           .     .     .           .           .
                              .          perror("");
136             .     .     .           .     .     .           .           .
                              .          goto _exit_main;
137             .     .     .           .     .     .           .           .
                              .        }
138          64     2     2          32     0     0           0           0
                             0        if (nptr == endptr) {
139             .     .     .           .     .     .           .           .
                              .          fprintf(stderr, "missing A
                matrix element");
140             .     .     .           .     .     .           .           .
                              .          goto _exit_main;
141             .     .     .           .     .     .           .           .
                              .        }
142         112     1     1          64     0     0          16           3
                            3        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 .     .     .           .     .     .           .           .
                  .
149 170   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 .     .     .           .     .     .           .           .
                  .           perror("");
154 .     .     .           .     .     .           .           .
                  .           goto _exit_main;
155 .     .     .           .     .     .           .           .
                  .         }
156 64    1     1           32    0     0           0           0
                  0         if (nptr == endptr) {
157 .     .     .           .     .     .           .           .
                  .           fprintf(stderr, "missing B
            matrix element");
158 .     .     .           .     .     .           .           .
                  .           goto _exit_main;
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 .     .     .           .     .     .           .           .
                  .       /* multiply matrixes */
166 13    2     2           6     1     1           1           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    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
                          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           .     .     .           .     .     .           .           .
                           .     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       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          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      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          .     .     .            .     .     .           .          .
```

```
                    .       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  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
322  --------------------------------------------------------------------------------

323  Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
           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==
```

```
--638-- Warning: Cannot auto-detect cache config, using defaults.
--638--          Run with -v to see.
==638==
==638== I   refs:      147,186,902
==638== I1  misses:          2,400
==638== LLi misses:          2,382
==638== I1  miss rate:        0.00%
==638== LLi miss rate:        0.00%
==638==
==638== D   refs:       73,550,864  (52,528,959 rd   + 21,021,905 wr)
==638== D1  misses:      1,315,099  (     3,754 rd   + 1,311,345 wr)
==638== LLd misses:      1,314,183  (     2,915 rd   + 1,311,268 wr)
==638== D1  miss rate:         1.8% (       0.0%     +     6.2%  )
==638== LLd miss rate:         1.8% (       0.0%     +     6.2%  )
==638==
==638== LL refs:         1,317,499  (     6,154 rd   + 1,311,345 wr)
==638== LL misses:       1,316,565  (     5,297 rd   + 1,311,268 wr)
==638== LL miss rate:          0.6% (       0.0%     +     6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.638
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
--------------------------------------------------------------------------------

147,186,902 2,400 2,382 52,528,959 3,754 2,915 21,021,905
    1,311,345 1,311,268  PROGRAM TOTALS

--------------------------------------------------------------------------------

Ir           I1mr ILmr Dr          D1mr   DLmr Dw          D1mw
```

```
         DLmw       file:function
--------------------------------------------------------------------------------

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

--------------------------------------------------------------------------------

-- 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);
         .     .     .           .     .     .           .           .
                       .
         .     .     .           .     .     .           .           .
                       .  int print_matrix(FILE* fp, matrix_t* m)
            ;
         .     .     .           .     .     .           .           .
                       .
         .     .     .           .     .     .           .           .
                       .  int
         .     .     .           .     .     .           .           .
                       .  main(int argc, char** argv)
        10     2     2           0     0     0           5           0
                      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) */
         1     0     0           0     0     0           1           0
                       0    char *line = NULL;
         .     .     .           .     .     .           .           .
                       .    /* line parsing auxiliar pointers */
         .     .     .           .     .     .           .           .
                       .    char *nptr, *endptr;
         .     .     .           .     .     .           .           .
```

```
                    .     /* 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     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) l;
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) l;
117         .    .    .          .    .    .          .          .
                 .
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          1          0
                   0    bs = n;
124         .    .    .          .    .    .          .          .
                 .    #endif
125         .    .    .          .    .    .          .          .
                 .
```

```
126 |          .     .     .           .     .     .           .           .
    |                      .       /* load matrix a */
127 |         11     1     1           5     1     0           1           0
    |                      0       if (!(a = create_matrix(n, n)))
128 |          .     .     .           .     .     .           .           .
    |                      .         goto _exit_main;
129 |          .     .     .           .     .     .           .           .
    |                      .
130 |        260     3     3         103     0     0          26           0
    |                      0       for (i=0; i < n*n; i++) {
131 |         50     0     0          25     0     0          25           0
    |                      0         nptr = endptr;
132 |        225     1     1          75     0     0          25           0
    |                      0         e = strtod(nptr, &endptr);
133 |        200     1     1          75     0     0           0           0
    |                      0         if (errno) {
134 |          .     .     .           .     .     .           .           .
    |                      .           perror("");
135 |          .     .     .           .     .     .           .           .
    |                      .           goto _exit_main;
136 |          .     .     .           .     .     .           .           .
    |                      .         }
137 |        100     2     2          50     0     0           0           0
    |                      0         if (nptr == endptr) {
138 |          .     .     .           .     .     .           .           .
    |                      .           fprintf(stderr, "missing A
    |            matrix element");
139 |          .     .     .           .     .     .           .           .
    |                      .           goto _exit_main;
140 |          .     .     .           .     .     .           .           .
    |                      .         }
141 |        175     1     1         100     0     0          25           5
    |                      5         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 |        260     2     2         103     0     0          26           0
    |                      0       for (i=0; i < n*n; i++) {
149 |         50     1     1          25     0     0          25           0
    |                      0         nptr = endptr;
150 |        225     1     1          75     0     0          25           0
    |                      0         e = strtod(nptr, &endptr);
151 |        200     1     1          75     0     0           0           0
    |                      0         if (errno) {
152 |          .     .     .           .     .     .           .           .
```

```
                    .           perror("");
153          .    .    .           .    .    .           .           .
                    .           goto _exit_main;
154          .    .    .           .    .    .           .           .
                    .         }
155        100    1    1          50    0    0           0           0
                  0         if (nptr == endptr) {
156          .    .    .           .    .    .           .           .
                    .           fprintf(stderr, "missing B
               matrix element");
157          .    .    .           .    .    .           .           .
                    .           goto _exit_main;
158          .    .    .           .    .    .           .           .
                    .         }
159        175    1    1         100    0    0          25           5
                  5         b->array[i] = e;
160          .    .    .           .    .    .           .           .
                    .       }
161          .    .    .           .    .    .           .           .
                    .
162          8    1    1           2    0    0           0           0
                    0       clock_gettime(CLOCK_REALTIME, &t0);
163          .    .    .           .    .    .           .           .
                    .
164          .    .    .           .    .    .           .           .
                    .       /* multiply matrixes */
165         13    2    2           6    1    1           1           0
                    0       if (!(c = matrix_multiply(a, b, bs))
            )
166          .    .    .           .    .    .           .           .
                    .         goto _exit_main;
167          .    .    .           .    .    .           .           .
                    .
168          8    1    1           2    0    0           0           0
                    0       clock_gettime(CLOCK_REALTIME, &t1);
169          .    .    .           .    .    .           .           .
                    .
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  }
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         .    .    .             .    .    .          .          .
                  .    double sum;
200         .    .    .             .    .    .          .          .
                  .    matrix_t* mr;
201         .    .    .             .    .    .          .          .
                  .
202         3    1    1             2    0    0          1          0
                  0    n = m1->rows;
203         .    .    .             .    .    .          .          .
```

```
                                   .
        11      1      1          5      0      0          1          0
                                   0     if(!(mr = create_matrix(n,n))) return
            NULL;
         .      .      .          .      .      .          .          .
                                   .
         9      2      2          3      0      0          1          0
                                   0     en = bs*(n/bs);
         .      .      .          .      .      .          .          .
                                   .
        48      2      2         17      0      0          6          0
                                   0     for(i=0; i<n; i++)
       240      1      1         85      0      0         30          0
                                  0       for(j=0; j<n; j++)
       275      1      1        125      0      0         50          5
                                  5         mr->array[i*n+j] = 0.0;
         .      .      .          .      .      .          .          .
                                   .
         .      .      .          .      .      .          .          .
                                   .   #if 1
         .      .      .          .      .      .          .          .
                                   .     if (1) {
         .      .      .          .      .      .          .          .
                                   .       size_t j;
         2      0      0          0      0      0          1          0
                                   0       size_t dim = 1024*1024*10;
         9      1      1          3      0      0          1          0
                                   0       int *v = malloc(dim*sizeof(int));
83,886,088      2      2 31,457,282      0      0 10,485,761          0
            0          for (j = 0; j < dim; ++j)
62,914,560      0      0 20,971,520      0      0 10,485,760  1,310,720
   1,310,720                    v[j] = -1;
         6      1      1          3      2      2          0          0
                                   0       free(v);
         .      .      .          .      .      .          .          .
                                   .     }
         .      .      .          .      .      .          .          .
                                   .   #endif
         .      .      .          .      .      .          .          .
                                   .
        17      2      2          6      0      0          2          0
                                   0     for(kk=0; kk<en; kk+=bs)
        17      2      2          6      0      0          2          0
                                   0       for(jj=0; jj<en; jj+=bs)
        48      1      1         17      0      0          6          0
                                   0         for(i=0; i<n; i++)
       305      2      2        120      0      0         30          0
                                  0           for(j=jj; j<jj+bs; j++) {
       275      1      1        150      7      7         25          0
                                  0             sum = mr->array[i*n+j];
     1,525      3      3        600      0      0        150          0
                                0               for(k=kk; k<kk+bs; k++)
     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         .     .     .           .     .     .           .          .
                          .   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  .    .    .          .    .    .          .          .
                .          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          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)
    m->rows) < 0) {
305         .   .   .           .   .   .           .           .
                    .       perror("");
306         .   .   .           .   .   .           .           .
                    .       return -1;
307         .   .   .           .   .   .           .           .
                    .   }
308        48   2   2          17   0   0           6           0
                   0   for(i=0; i<n; i++)
```

```
309         240     2     2           85     0     0           30           0
                    0       for (j=0; j<n; j++)
310         550     3     3          250     0     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     if (fprintf(fp, "\n") < 0) {
315           .     .     .            .     .     .            .           .
                      .       perror("");
316           .     .     .            .     .     .            .           .
                      .       return -1;
317           .     .     .            .     .     .            .           .
                      .     }
318           1     1     1            0     0     0            0           0
                    0     return 0;
319           6     1     1            2     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    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)
```

```
==641== D1  misses:       1,315,141  (     3,775 rd   +     1,311,366 wr)
==641== LLd misses:       1,314,203  (     2,925 rd   +     1,311,278 wr)
==641== D1  miss rate:          1.8% (       0.0%     +           6.2%  )
==641== LLd miss rate:          1.8% (       0.0%     +           6.2%  )
==641==
==641== LL refs:          1,317,548  (     6,182 rd   +     1,311,366 wr)
==641== LL misses:        1,316,592  (     5,314 rd   +     1,311,278 wr)
==641== LL miss rate:           0.6% (       0.0%     +           6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.641
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
--------------------------------------------------------------------------------

147,225,983 2,407 2,389 52,539,343 3,775 2,925 21,026,520    1,311,366 1,311,278  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

146,810,542   29   29 52,433,589    31   31 20,972,246    1,310,728 1,310,728  /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);
         .     .     .          .     .     .          .
                     .
         .     .     .          .     .     .          .
                     .  int print_matrix(FILE* fp, matrix_t* m)
             ;
         .     .     .          .     .     .          .
                     .
         .     .     .          .     .     .          .
                     .  int
         .     .     .          .     .     .          .
                     .  main(int argc, char** argv)
        10     2     2          0     0     0          5          0
                       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) */
         1     0     0          0     0     0          1          0
                       0    char *line = NULL;
         .     .     .          .     .     .          .
                     .    /* line parsing auxiliar pointers */
         .     .     .          .     .     .          .
                     .    char *nptr, *endptr;
         .     .     .          .     .     .          .
                     .    /* auxiliar variables */
         .     .     .          .     .     .          .
                     .    long l;
         .     .     .          .     .     .          .
                     .    double e;
         2     1     1          0     0     0          1          0
                       0    size_t lineno = 1;
         .     .     .          .     .     .          .
                     .    struct timespec t0;
         .     .     .          .     .     .          .
                     .    struct timespec t1;
         .     .     .          .     .     .          .
```

```
                    .     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     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) l;
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) l;
117          .     .     .          .     .     .          .          .
                     .
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          1          0
                       0       bs = n;
124          .     .     .          .     .     .          .          .
                     .   #endif
125          .     .     .          .     .     .          .          .
                     .
126          .     .     .          .     .     .          .          .
                     .       /* load matrix a */
127         11     1     1          5     1     0          1          0
                       0       if (!(a = create_matrix(n, n)))
128          .     .     .          .     .     .          .          .
                     .         goto _exit_main;
129          .     .     .          .     .     .          .          .
                     .
130        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
                         0          e = strtod(nptr, &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 (nptr == endptr) {
138          .     .     .            .     .     .           .          .
                           .            fprintf(stderr, "missing A
               matrix element");
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         nptr = endptr;
150        324     1     1          108     0     0          36          0
                         0          e = strtod(nptr, &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 (nptr == endptr) {
156          .     .     .            .     .     .           .          .
                           .            fprintf(stderr, "missing B
               matrix element");
157          .     .     .            .     .     .           .          .
                           .            goto _exit_main;
```

```
158          .     .     .           .     .     .           .          .
                                  .         }
159        252     1     1          144     0     0          36          9
                                 9         b->array[i] = e;
160          .     .     .           .     .     .           .          .
                                  .       }
161          .     .     .           .     .     .           .          .
                                  .
162          8     1     1           2     0     0           0          0
                                    0       clock_gettime(CLOCK_REALTIME, &t0);
163          .     .     .           .     .     .           .          .
                                  .
164          .     .     .           .     .     .           .          .
                                  .       /* multiply matrixes */
165         13     2     2           6     1     1           1          0
                                    0       if (!(c = matrix_multiply(a, b, bs))
                                        )
166          .     .     .           .     .     .           .          .
                                  .         goto _exit_main;
167          .     .     .           .     .     .           .          .
                                  .
168          8     1     1           2     0     0           0          0
                                    0       clock_gettime(CLOCK_REALTIME, &t1);
169          .     .     .           .     .     .           .          .
                                  .
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   }
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          .     .     .           .     .     .           .           .
                .     double sum;
200          .     .     .           .     .     .           .           .
                .     matrix_t* mr;
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         56     2     2          20     0     0           7           0
                    0     for(i=0; i<n; i++)
```

```
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           .     .     .            .     .     .           .          .
```

```
                    .  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   2,201    5    5        734    1    0        147          0
                 0    while (EOF != (c=fgetc(fp)) && c != '\n')
                    {
251   1,160    2    2        435    0    0        290          4
                 4      str[len++]=c;
252     725    0    0        290    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           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)
          m->rows) < 0) {
305          .    .    .          .    .    .          .          .
                      .      perror("");
306          .    .    .          .    .    .          .          .
                      .      return -1;
307          .    .    .          .    .    .          .          .
                      .    }
308         56    2    2         20    0    0          7          0
                    0    for(i=0; i<n; i++)
309        336    2    2        120    0    0         42          0
                    0      for (j=0; j<n; j++)
310        792    3    3        360    0    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     if (fprintf(fp, "\n") < 0) {
315         .     .     .           .     .     .          .          .
                  .       perror("");
316         .     .     .           .     .     .          .          .
                  .       return -1;
317         .     .     .           .     .     .          .          .
                  .     }
318         1     1     1           0     0     0          0          0
                  0     return 0;
319         6     1     1           2     0     0          0          0
                  0   }
320
321 --------------------------------------------------------------------------------

322 Ir           I1mr ILmr Dr          D1mr DLmr Dw          D1mw
          DLmw
323 --------------------------------------------------------------------------------

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%     +
```

```
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.644
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
--------------------------------------------------------------------------------

147,273,033 2,400 2,382 52,551,899 3,660 2,936 21,032,031
    1,311,345 1,311,292  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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:
    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);
         .    .    .           .    .    .           .         .
                   .
         .    .    .           .    .    .           .         .
```

```
                    .  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
                 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
                 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
                 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
                 0    for(; !feof(stdin); lineno++) {
81         10    1    1          4    0    0          6          0
                 0      a=b=c=NULL;
82          .    .    .          .    .    .          .          .
                 .
83         18    3    3          8    1    0          2          0
                 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          .     .     .           .     .     .           .          .
                .       /* parse dimension */
89          2     1     1           1     0     0           1          1
                  0       nptr = line;
90         10     1     1           3     1     0           1          0
                   0      l = strtol(nptr, &endptr, 10);
91          8     1     1           3     0     0           0          0
                  0       if (errno) {
92          .     .     .           .     .     .           .          .
                  .         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         .     .     .           .     .     .           .          .
                  .     #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) 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         .    .    .           .    .    .          .          .
                    .   #endif
124         .    .    .           .    .    .          .          .
                    .
125         .    .    .           .    .    .          .          .
                    .       /* load matrix a */
126        11    1    1           5    1    0          1          0
                     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
```

```
                matrix element");
138           .     .     .           .     .     .           .           .
                    .           goto _exit_main;
139           .     .     .           .     .     .           .           .
                    .         }
140         343     1     1         196     0     0          49          11
                 11         a->array[i] = e;
141           .     .     .           .     .     .           .           .
                    .       }
142           .     .     .           .     .     .           .           .
                    .
143           .     .     .           .     .     .           .           .
                    .       /* load matrix b */
144          11     1     1           5     0     0           1           0
                  0       if (!(b = create_matrix(n, n)))
145           .     .     .           .     .     .           .           .
                    .         goto _exit_main;
146           .     .     .           .     .     .           .           .
                    .
147         500     2     2         199     0     0          50           0
                  0       for (i=0; i < n*n; i++) {
148          98     1     1          49     0     0          49           0
                  0         nptr = endptr;
149         441     1     1         147     0     0          49           0
                  0         e = strtod(nptr, &endptr);
150         392     1     1         147     0     0           0           0
                  0         if (errno) {
151           .     .     .           .     .     .           .           .
                    .           perror("");
152           .     .     .           .     .     .           .           .
                    .           goto _exit_main;
153           .     .     .           .     .     .           .           .
                    .         }
154         196     1     1          98     0     0           0           0
                  0         if (nptr == endptr) {
155           .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing B
                matrix element");
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
                       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          .     .     .           .     .     .           .           .
                     .     destroy_matrix(b);
190          .     .     .           .     .     .           .           .
                     .     destroy_matrix(c);
191          .     .     .           .     .     .           .           .
                     .     exit(1);
192          6     1     1           2     0     0           0           0
                       0   }
193          .     .     .           .     .     .           .           .
                     .
194          .     .     .           .     .     .           .           .
                     .   matrix_t*
195          .     .     .           .     .     .           .           .
                     .   matrix_multiply(matrix_t* m1, matrix_t*
                 m2, int bs)
196         11     2     2           0     0     0           6           0
                       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
                       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          .     .     .           .     .     .           .           .
                     .   #if 1
212          .     .     .           .     .     .           .           .
                     .     if (1) {
213          .     .     .           .     .     .           .           .
                     .       size_t j;
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     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          .     .     .           .     .     .           .           .
```

```
                    .     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
                     0    if (c != EOF)
261       7     0     0           2     0     0           2           0
                     0      str[len++]='\n';
262       .     .     .           .     .     .           .           .
                    .
263      12     1     1           4     0     0           4           0
                     0    str[len++]='\0';
264       2     0     0           2     0     0           0           0
                     0    return str;
265      12     1     1           4     0     0           0           0
                     0  }
266       .     .     .           .     .     .           .           .
                    .
267       .     .     .           .     .     .           .           .
                    .
```

```
268  .     .     .           .     .     .           .           .
                    .  matrix_t*
269  .     .     .           .     .     .           .           .
                    .  create_matrix(size_t rows, size_t cols)
270  30    1     1           0     0     0           15          0
                   0  {
271  .     .     .           .     .     .           .           .
                    .    matrix_t * m;
272  .     .     .           .     .     .           .           .
                    .
273  30    2     2           9     1     0           3           0
                   0    if (!(m = malloc(sizeof(matrix_t)))) {
274  .     .     .           .     .     .           .           .
                    .      perror("");
275  .     .     .           .     .     .           .           .
                    .      return NULL;
276  .     .     .           .     .     .           .           .
                    .    }
277  .     .     .           .     .     .           .           .
                    .
278  9     1     1           6     0     0           3           0
                  0    m->rows = rows;
279  9     1     1           6     0     0           3           0
                  0    m->cols = cols;
280  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  {
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  {
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          .     .     .           .     .     .           .           .
                      .      return -1;
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          .     .     .           .     .     .           .           .
                      .          perror("");
311          .     .     .           .     .     .           .           .
                      .          return -1;
312          .     .     .           .     .     .           .           .
                      .        }
313         10     1     1           4     0     0           0           0
                    0    if (fprintf(fp, "\n") < 0) {
314          .     .     .           .     .     .           .           .
                      .      perror("");
315          .     .     .           .     .     .           .           .
                      .      return -1;
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 ------------------------------------------------------------------------------
```

```
Ir           I1mr ILmr Dr          D1mr DLmr Dw          D1mw
         DLmw
--------------------------------------------------------------------------------

146,827,393  120  119 52,440,658   54   48 20,973,736 1,310,762
     1,310,760  events annotated
```

```
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
 PID: 597
==647==
--647-- Warning: Cannot auto-detect cache config, using
    defaults.
--647--          Run with -v to see.
==647==
==647== I   refs:      147,337,400
==647== I1  misses:          2,405
==647== LLi misses:          2,387
==647== I1  miss rate:        0.00%
==647== LLi miss rate:        0.00%
==647==
==647== D   refs:       73,608,465  (52,568,673 rd   +
    21,039,792 wr)
==647== D1  misses:      1,315,252  (     3,828 rd   +
    1,311,424 wr)
==647== LLd misses:      1,314,257  (     2,949 rd   +
    1,311,308 wr)
==647== D1  miss rate:         1.8% (       0.0%     +
    6.2%  )
==647== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==647==
==647== LL refs:         1,317,657  (     6,233 rd   +
    1,311,424 wr)
==647== LL misses:       1,316,644  (     5,336 rd   +
    1,311,308 wr)
==647== LL miss rate:          0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 2-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.647
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
--------------------------------------------------------------------------------

147,337,400 2,405 2,387 52,568,673 3,828 2,949 21,039,792
    1,311,424 1,311,308  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
         .    .    .          .    .    .          .          .
                   .
         .    .    .          .    .    .          .          .
                   .  int print_matrix(FILE* fp, matrix_t* m)
            ;
         .    .    .          .    .    .          .          .
                   .
         .    .    .          .    .    .          .          .
                   .  int
         .    .    .          .    .    .          .          .
                   .  main(int argc, char** argv)
        10    2    2          0    0    0          5          0
                   0  {
         .    .    .          .    .    .          .          .
                   .    /* matrixes */
         .    .    .          .    .    .          .          .
                   .    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           1           0
                      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           1           0
                      0     size_t lineno = 1;
73          .     .     .           .     .     .           .           .
                    .     struct timespec t0;
74          .     .     .           .     .     .           .           .
                    .     struct timespec t1;
75          .     .     .           .     .     .           .           .
                    .     double dt;
76          .     .     .           .     .     .           .           .
                    .     size_t i;
77          .     .     .           .     .     .           .           .
                    .
78         25     4     4           9     0     0           1           0
                      0     for(; !feof(stdin); lineno++) {
79         10     1     1           4     0     0           6           0
                      0       a=b=c=NULL;
80          .     .     .           .     .     .           .           .
                    .
81         18     3     3           8     1     0           2           0
                      0       line = read_line(stdin);
82          .     .     .           .     .     .           .           .
                    .
83          6     0     0           2     0     0           0           0
                      0       if (!line) goto _exit_main;
84          9     0     0           4     0     0           0           0
                      0       if (line[0] == 0) break;
85          .     .     .           .     .     .           .           .
                    .
86          .     .     .           .     .     .           .           .
                    .       /* parse dimension */
87          2     1     1           1     0     0           1           1
                      0       nptr = line;
88         10     1     1           3     1     0           1           0
                      0       l = strtol(nptr, &endptr, 10);
89          8     1     1           3     0     0           0           0
```

```
                        0       if (errno) {
90          .     .     .           .     .     .           .           .
                        .         perror("");
91          .     .     .           .     .     .           .           .
                        .         goto _exit_main;
92          .     .     .           .     .     .           .           .
                        .       }
93          4     2     2           2     0     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
                        0       if (l < 1) {
98          .     .     .           .     .     .           .           .
                        .         fprintf(stderr, "invalid
                dimension");
99          .     .     .           .     .     .           .           .
                        .         goto _exit_main;
100         .     .     .           .     .     .           .           .
                        .       }
101         2     1     1           1     0     0           1           0
                        0       n = (size_t) l;
102         .     .     .           .     .     .           .           .
                        .   #if 0
103         .     .     .           .     .     .           .           .
                        .       /* parse block size */
104         .     .     .           .     .     .           .           .
                        .       nptr = endptr;
105         .     .     .           .     .     .           .           .
                        .       l = strtol(nptr, &endptr, 10);
106         .     .     .           .     .     .           .           .
                        .       if (errno) {
107         .     .     .           .     .     .           .           .
                        .         perror("");
108         .     .     .           .     .     .           .           .
                        .         goto _exit_main;
109         .     .     .           .     .     .           .           .
                        .       }
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
                      0       bs = n;
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
                    0       for (i=0; i < n*n; i++) {
128       128    0    0         64    0    0         64          0
                    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
```

```
                    0       if (!(b = create_matrix(n, n)))
143       .     .     .           .     .     .           .           .
                    .         goto _exit_main;
144       .     .     .           .     .     .           .           .
                    .
145     650     2     2         259     0     0          65           0
                    0       for (i=0; i < n*n; i++) {
146     128     1     1          64     0     0          64           0
                    0         nptr = endptr;
147     576     1     1         192     0     0          64           0
                    0         e = strtod(nptr, &endptr);
148     512     1     1         192     0     0           0           0
                    0         if (errno) {
149       .     .     .           .     .     .           .           .
                    .           perror("");
150       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
151       .     .     .           .     .     .           .           .
                    .         }
152     256     1     1         128     0     0           0           0
                    0         if (nptr == endptr) {
153       .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing B
            matrix element");
154       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
155       .     .     .           .     .     .           .           .
                    .         }
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       .     .     .           .     .     .           .           .
                    .       /* multiply matrixes */
162      13     2     2           6     1     1           1           0
                    0       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    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*
```

```
                m2, int bs)
194         11     2     2           0     0     0           6           0
                          0  {
195          .     .     .           .     .     .           .           .
                           .     size_t n, en, i, j, k, kk, jj;
196          .     .     .           .     .     .           .           .
                           .     double sum;
197          .     .     .           .     .     .           .           .
                           .     matrix_t* mr;
198          .     .     .           .     .     .           .           .
                           .
199          3     1     1           2     0     0           1           0
                           0     n = m1->rows;
200          .     .     .           .     .     .           .           .
                           .
201         11     1     1           5     0     0           1           0
                           0     if(!(mr = create_matrix(n,n))) return
            NULL;
202          .     .     .           .     .     .           .           .
                           .
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     0     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 1,310,720
    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          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  .    .    .          .    .    .          .          .
                  .      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
                  0  {
290  9    0    0          3    0    0          0          0
                  0    if (!m) return;
291 24    1    1         12    0    0          0          0
                  0    free(m->array);
292 24    1    1          9    0    0          0          0
                  0    free(m);
293 18    0    0          6    0    0          0          0
                  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
```

6.3. 4WSA:

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

```
==651== Copyright (C) 2002-2017, and GNU GPL'd, by Nicholas Nethercote et al.
==651== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==651== Command: /tmp/02-mmult
==651== Parent PID: 597
==651==
--651-- Warning: Cannot auto-detect cache config, using defaults.
--651--          Run with -v to see.
==651==
==651== I   refs:      147,097,895
==651== I1  misses:          2,354
==651== LLi misses:          2,339
==651== I1  miss rate:        0.00%
==651== LLi miss rate:        0.00%
==651==
==651== D   refs:       73,517,815  (52,506,511 rd   + 21,011,304 wr)
==651== D1  misses:      1,314,783  (     3,494 rd   + 1,311,289 wr)
==651== LLd misses:      1,314,133  (     2,894 rd   + 1,311,239 wr)
==651== D1  miss rate:         1.8% (       0.0%     +     6.2%  )
==651== LLd miss rate:         1.8% (       0.0%     +     6.2%  )
==651==
==651== LL refs:         1,317,137  (     5,848 rd   + 1,311,289 wr)
==651== LL misses:       1,316,472  (     5,233 rd   + 1,311,239 wr)
==651== LL miss rate:          0.6% (       0.0%     +     6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.651
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
```

```
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  {
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     3     3           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          .     .     .           .     .     .           .           .
                        .       /* 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 .    .    .          .    .    .          .          .
                .        perror("");
112 .    .    .          .    .    .          .          .
                .        goto _exit_main;
113 .    .    .          .    .    .          .          .
                .      }
114 -- line 75 ----------------------------------------
115 -- line 83 ----------------------------------------
116 .    .    .          .    .    .          .          .
                .      }
117 .    .    .          .    .    .          .          .
                .      bs = (size_t) l;
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 .     .     .           .     .     .           .           .
                  .         goto _exit_main;
159 .     .     .           .     .     .           .           .
                  .       }
160 7     1     1           4     0     0           1           0
                  0       b->array[i] = e;
161 .     .     .           .     .     .           .           .
                  .     }
162 .     .     .           .     .     .           .           .
                .
163 8     1     1           2     0     0           0           0
                  0     clock_gettime(CLOCK_REALTIME, &t0);
164 .     .     .           .     .     .           .           .
                .
165 .     .     .           .     .     .           .           .
                  .     /* multiply matrixes */
166 13    2     2           6     1     1           1           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", (
    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 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 .     .     .           .     .     .           .           .
                .   #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 |          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     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
    |                       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           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          16     2     2           5     0     0           2           0
                      0     for(i=0; i<n; i++)
310          16     2     2           5     0     0           2           0
                      0       for (j=0; j<n; j++)
311          22     3     3          10     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
322 --------------------------------------------------------------------------------

323 Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------

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
```

```
==654== Command: /tmp/02-mmult
==654== Parent PID: 597
==654==
--654-- Warning: Cannot auto-detect cache config, using defaults.
--654--          Run with -v to see.
==654==
==654== I   refs:      147,109,372
==654== I1  misses:          2,401
==654== LLi misses:          2,383
==654== I1  miss rate:        0.00%
==654== LLi miss rate:        0.00%
==654==
==654== D   refs:       73,522,043  (52,509,336 rd   + 21,012,707 wr)
==654== D1  misses:      1,314,795  (     3,498 rd   + 1,311,297 wr)
==654== LLd misses:      1,314,145  (     2,898 rd   + 1,311,247 wr)
==654== D1  miss rate:         1.8% (       0.0%     +       6.2%  )
==654== LLd miss rate:         1.8% (       0.0%     +       6.2%  )
==654==
==654== LL refs:         1,317,196  (     5,899 rd   + 1,311,297 wr)
==654== LL misses:       1,316,528  (     5,281 rd   + 1,311,247 wr)
==654== LL miss rate:          0.6% (       0.0%     +       6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.654
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
--------------------------------------------------------------------------------

147,109,372 2,401 2,383 52,509,336 3,498 2,898 21,012,707 1,311,297 1,311,247  PROGRAM TOTALS

```

```
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  .   .   .       .   .   .       .       .
                .     /* 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             .     .     .           .     .     .           .           .
                              .         perror("");
112             .     .     .           .     .     .           .           .
                              .         goto _exit_main;
113             .     .     .           .     .     .           .           .
                              .       }
114     -- line 75 ----------------------------------------
115     -- line 83 ----------------------------------------
116             .     .     .           .     .     .           .           .
                              .       }
117             .     .     .           .     .     .           .           .
                              .       bs = (size_t) l;
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         50    3    3         19    0    0          5          0
                    0      for (i=0; i < n*n; i++) {
132          8    0    0          4    0    0          4          0
                     0       nptr = endptr;
133         36    1    1         12    0    0          4          0
                    0        e = strtod(nptr, &endptr);
134         32    1    1         12    0    0          0          0
                    0        if (errno) {
135          .    .    .          .    .    .          .          .
                    .          perror("");
136          .    .    .          .    .    .          .          .
                    .          goto _exit_main;
137          .    .    .          .    .    .          .          .
                    .        }
138         16    2    2          8    0    0          0          0
                    0        if (nptr == endptr) {
139          .    .    .          .    .    .          .          .
                    .          fprintf(stderr, "missing A
    matrix element");
140          .    .    .          .    .    .          .          .
                    .          goto _exit_main;
141          .    .    .          .    .    .          .          .
                    .        }
142         28    1    1         16    0    0          4          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         50    2    2         19    0    0          5          0
                    0      for (i=0; i < n*n; i++) {
150          8    1    1          4    0    0          4          0
                     0       nptr = endptr;
151         36    1    1         12    0    0          4          0
                    0        e = strtod(nptr, &endptr);
```

```
152          32     1     1           12     0     0           0           0
                           0         if (errno) {
153           .     .     .            .     .     .           .           .
                            .          perror("");
154           .     .     .            .     .     .           .           .
                            .          goto _exit_main;
155           .     .     .            .     .     .           .           .
                            .        }
156          16     1     1            8     0     0           0           0
                           0         if (nptr == endptr) {
157           .     .     .            .     .     .           .           .
                            .          fprintf(stderr, "missing B
                 matrix element");
158           .     .     .            .     .     .           .           .
                            .          goto _exit_main;
159           .     .     .            .     .     .           .           .
                            .        }
160          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           .     .     .            .     .     .           .           .
                            .       /* multiply matrixes */
166          13     2     2            6     1     1           1           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", (
    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 |          .     .     .           .     .     .           .           .
    |                        .   #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
```

```
                    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         .     .     .          .     .     .          .          .
                    .   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       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         .     .     .          .     .     .          .          .
                    .         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           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          24     2     2           8     0     0           3           0
                          0     for(i=0; i<n; i++)
310          48     2     2          16     0     0           6           0
                          0       for (j=0; j<n; j++)
311          88     3     3          40     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
322 --------------------------------------------------------------------------------

323 Ir           I1mr  ILmr Dr          D1mr  DLmr Dw          D1mw
          DLmw
324 --------------------------------------------------------------------------------

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
       Nethercote et al.
4  ==657== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
       copyright info
5  ==657== Command: /tmp/02-mmult
6  ==657== Parent PID: 597
7  ==657==
8  --657-- Warning: Cannot auto-detect cache config, using
       defaults.
9  --657--          Run with -v to see.
10 ==657==
11 ==657== I   refs:      147,126,801
12 ==657== I1  misses:          2,402
13 ==657== LLi misses:          2,384
14 ==657== I1  miss rate:        0.00%
15 ==657== LLi miss rate:        0.00%
```

```
==657==
==657== D   refs:       73,528,505  (52,513,724 rd   + 21,014,781 wr)
==657== D1  misses:      1,314,800  (     3,503 rd   + 1,311,297 wr)
==657== LLd misses:      1,314,153  (     2,901 rd   + 1,311,252 wr)
==657== D1  miss rate:         1.8% (       0.0%     + 6.2%  )
==657== LLd miss rate:         1.8% (       0.0%     + 6.2%  )
==657==
==657== LL refs:         1,317,202  (     5,905 rd   + 1,311,297 wr)
==657== LL misses:       1,316,537  (     5,285 rd   + 1,311,252 wr)
==657== LL miss rate:          0.6% (       0.0%     + 6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.657
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
--------------------------------------------------------------------------------

147,126,801 2,402 2,384 52,513,724 3,503 2,901 21,014,781
    1,311,297 1,311,252  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------
```

```
-- 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);
           .     .     .           .     .     .           .     .
           .     .     .           .     .     .           .     .  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) */
           1     0     0           0     0     0           1     0      char *line = NULL;
           .     .     .           .     .     .           .     .      /* line parsing auxiliar pointers */
           .     .     .           .     .     .           .     .      char *nptr, *endptr;
           .     .     .           .     .     .           .     .      /* auxiliar variables */
           .     .     .           .     .     .           .     .      long l;
           .     .     .           .     .     .           .     .      double e;
           2     1     1           0     0     0           1     0      size_t lineno = 1;
           .     .     .           .     .     .           .     .      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          .     .     .           .     .     .           .           .
                  .       /* 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 |          .     .     .           .     .     .           .           .
    |                      .         perror("");
112 |          .     .     .           .     .     .           .           .
    |                      .         goto _exit_main;
113 |          .     .     .           .     .     .           .           .
    |                      .       }
114 | -- line 75 ----------------------------------------
115 | -- line 83 ----------------------------------------
116 |          .     .     .           .     .     .           .           .
    |                      .       }
117 |          .     .     .           .     .     .           .           .
    |                      .       bs = (size_t) l;
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 |        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
                     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, &t0);
164             .     .     .           .     .     .           .           .
                      .
165             .     .     .           .     .     .           .           .
                      .       /* multiply matrixes */
166           13    2     2           6     1     1           1           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", (
            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          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  .     .     .          .     .     .          .          .
                  .  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  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           .     .     .           .     .     .           .           .
                                .   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;
```

```
          .    .    .           .    .    .           .          .
                        .         }
         10    1    1           4    0    0           0          0
                       0    if (fprintf(fp, "\n") < 0) {
          .    .    .           .    .    .           .          .
                        .      perror("");
          .    .    .           .    .    .           .          .
                        .      return -1;
          .    .    .           .    .    .           .          .
                        .    }
          1    1    1           0    0    0           0          0
                       0    return 0;
          6    1    1           2    0    0           0          0
                       0  }

--------------------------------------------------------------------------------

Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
         DLmw
--------------------------------------------------------------------------------

146,805,085  120   119  52,430,618   24    18  20,971,960  1,310,727
     1,310,725  events annotated
```

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

```
==660== LL refs:          1,317,213  (     5,909 rd   +     1,311,304 wr)
==660== LL misses:        1,316,548  (     5,289 rd   +     1,311,259 wr)
==660== LL miss rate:           0.6% (       0.0%     +           6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.660
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
--------------------------------------------------------------------------------

147,153,509 2,400 2,382 52,520,440 3,509 2,907 21,017,959
    1,311,304 1,311,259  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
         .    .    .          .    .    .          .          .
                     .
```

```
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          .     .     .           .     .     .           .           .
                  .       /* 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 |          .     .     .           .     .     .           .           .
    |                        .         perror("");
112 |          .     .     .           .     .     .           .           .
    |                        .         goto _exit_main;
113 |          .     .     .           .     .     .           .           .
    |                        .       }
114 | -- line 75 ----------------------------------------
115 | -- line 83 ----------------------------------------
116 |          .     .     .           .     .     .           .           .
    |                        .       }
117 |          .     .     .           .     .     .           .           .
    |                        .       bs = (size_t) l;
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 |        170     3     3          67     0     0          17           0
    |                        0       for (i=0; i < n*n; i++) {
132 |         32     0     0          16     0     0          16           0
    |                        0         nptr = endptr;
133 |        144     1     1          48     0     0          16           0
    |                        0         e = strtod(nptr, &endptr);
134 |        128     1     1          48     0     0           0           0
    |                        0         if (errno) {
135 |          .     .     .           .     .     .           .           .
    |                        .           perror("");
136 |          .     .     .           .     .     .           .           .
    |                        .           goto _exit_main;
137 |          .     .     .           .     .     .           .           .
```

```
                    .          }
138          64     2     2           32     0     0           0           0
                               0          if (nptr == endptr) {
139           .     .     .            .     .     .           .           .
                    .                    fprintf(stderr, "missing A
                 matrix element");
140           .     .     .            .     .     .           .           .
                    .                    goto _exit_main;
141           .     .     .            .     .     .           .           .
                    .          }
142          112    1     1           64     0     0          16           3
                               3          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          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           .     .     .            .     .     .           .           .
                    .                    perror("");
154           .     .     .            .     .     .           .           .
                    .                    goto _exit_main;
155           .     .     .            .     .     .           .           .
                    .          }
156           64    1     1           32     0     0           0           0
                               0          if (nptr == endptr) {
157           .     .     .            .     .     .           .           .
                    .                    fprintf(stderr, "missing B
                 matrix element");
158           .     .     .            .     .     .           .           .
                    .                    goto _exit_main;
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       .    .    .          .    .    .          .          .
                    .       /* multiply matrixes */
166      13    2    2          6    1    1          1          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", (
```

```
                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          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             .     .     .           .     .     .           .           .
```

```
                        .    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         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           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       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 .    .    .          .    .    .          .          .
                   .      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         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;
```

```
          6    1    1           2    0    0          0          0
                    0  }

--------------------------------------------------------------------------------

Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
         DLmw
--------------------------------------------------------------------------------

146,808,382  120   119  52,432,069   29    23  20,972,236  1,310,735
     1,310,733  events annotated
```

```
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
017, and GNU GPL'd, by Nicholas Nethercote et al.
==663== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
==663== Command: /tmp/02-mmult
==663== Parent PID: 597
==663==
--663-- Warning: Cannot auto-detect cache config, using
    defaults.
--663--          Run with -v to see.
==663==
==663== I   refs:      147,187,051
==663== I1  misses:          2,407
==663== LLi misses:          2,389
==663== I1  miss rate:        0.00%
==663== LLi miss rate:        0.00%
==663==
==663== D   refs:       73,550,901  (52,528,981 rd   +
    21,021,920 wr)
==663== D1  misses:      1,314,830  (     3,517 rd   +
    1,311,313 wr)
==663== LLd misses:      1,314,183  (     2,915 rd   +
    1,311,268 wr)
==663== D1  miss rate:         1.8% (       0.0%     +
    6.2%  )
==663== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==663==
==663== LL refs:         1,317,237  (     5,924 rd   +
    1,311,313 wr)
==663== LL misses:       1,316,572  (     5,304 rd   +
    1,311,268 wr)
==663== LL miss rate:          0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.663
```

```
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
--------------------------------------------------------------------------------
147,187,051 2,407 2,389 52,528,981 3,517 2,915 21,021,920
    1,311,313 1,311,268  PROGRAM TOTALS

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

--------------------------------------------------------------------------------
-- 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);
         .    .    .           .    .    .           .           .
                    .
         .    .    .           .    .    .           .           .
                     .  int print_matrix(FILE* fp, matrix_t* m)
            ;
         .    .    .           .    .    .           .           .
                    .
         .    .    .           .    .    .           .           .
                     .  int
         .    .    .           .    .    .           .           .
                     .  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  .     .    .          .    .    .          .          .
                  .    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     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) l;
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) l;
117       .     .     .         .     .     .         .         .
                    .
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         1         0
                    0       bs = n;
124       .     .     .         .     .     .         .         .
                    .   #endif
125       .     .     .         .     .     .         .         .
                    .
126       .     .     .         .     .     .         .         .
                    .       /* load matrix a */
127      11     1     1         5     1     0         1         0
                    0       if (!(a = create_matrix(n, n)))
128       .     .     .         .     .     .         .         .
                    .         goto _exit_main;
129       .     .     .         .     .     .         .         .
                    .
130     260     3     3       103     0     0        26         0
                    0       for (i=0; i < n*n; i++) {
131      50     0     0        25     0     0        25         0
                    0         nptr = endptr;
132     225     1     1        75     0     0        25         0
                    0         e = strtod(nptr, &endptr);
133     200     1     1        75     0     0         0         0
                    0         if (errno) {
134       .     .     .         .     .     .         .         .
                    .           perror("");
135       .     .     .         .     .     .         .         .
                    .           goto _exit_main;
136       .     .     .         .     .     .         .         .
                    .         }
137     100     2     2        50     0     0         0         0
                    0         if (nptr == endptr) {
138       .     .     .         .     .     .         .         .
                    .           fprintf(stderr, "missing A
          matrix element");
139       .     .     .         .     .     .         .         .
                    .           goto _exit_main;
140       .     .     .         .     .     .         .         .
                    .         }
141     175     1     1       100     0     0        25         5
                    5         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     260     2     2         103     0     0          26           0
                    0       for (i=0; i < n*n; i++) {
149      50     1     1          25     0     0          25           0
                    0         nptr = endptr;
150     225     1     1          75     0     0          25           0
                    0         e = strtod(nptr, &endptr);
151     200     1     1          75     0     0           0           0
                    0         if (errno) {
152       .     .     .           .     .     .           .           .
                    .           perror("");
153       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
154       .     .     .           .     .     .           .           .
                    .         }
155     100     1     1          50     0     0           0           0
                    0         if (nptr == endptr) {
156       .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing B
          matrix element");
157       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
158       .     .     .           .     .     .           .           .
                    .         }
159     175     1     1         100     0     0          25           5
                    5         b->array[i] = e;
160       .     .     .           .     .     .           .           .
                    .       }
161       .     .     .           .     .     .           .           .
                    .
162       8     1     1           2     0     0           0           0
                    0       clock_gettime(CLOCK_REALTIME, &t0);
163       .     .     .           .     .     .           .           .
                    .
164       .     .     .           .     .     .           .           .
                    .       /* multiply matrixes */
165      13     2     2           6     1     1           1           0
                    0       if (!(c = matrix_multiply(a, b, bs))
          )
166       .     .     .           .     .     .           .           .
                    .         goto _exit_main;
167       .     .     .           .     .     .           .           .
                    .
168       8     1     1           2     0     0           0           0
```

```
                    0       clock_gettime(CLOCK_REALTIME, &t1);
169 .   .   .           .   .   .           .           .
                  .
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   }
```

```
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 |          .     .     .           .     .     .           .           .
                                .    double sum;
200 |          .     .     .           .     .     .           .           .
                                .    matrix_t* mr;
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 |          .     .     .           .     .     .           .           .
                                .  #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);
```

```
    .     .     .        .     .     .         .         .       }
    .     .     .        .     .     .         .         .       #endif
    .     .     .        .     .     .         .         .
   17     2     2        6     0     0         2         0       for(kk=0; kk<en; kk+=bs)
   17     2     2        6     0     0         2         0         for(jj=0; jj<en; jj+=bs)
   48     1     1       17     0     0         6         0           for(i=0; i<n; i++)
  305     2     2      120     0     0        30         0             for(j=jj; j<jj+bs; j++) {
  275     1     1      150     7     7        25         0               sum = mr->array[i*n+j];
1,525     3     3      600     0     0       150         0               for(k=kk; k<kk+bs; k++)
3,000     2     2    1,625    14    14       125         0                 sum += m1->array[i*n+k] * m2->array[k*n+j];
  275     1     1      150     0     0        25         0               mr->array[i*n+j] = sum;
    .     .     .        .     .     .         .         .             }
    1     0     0        1     0     0         0         0       return mr;
    6     1     1        2     0     0         0         0     }
    .     .     .        .     .     .         .         .
    .     .     .        .     .     .         .         .     char*
    .     .     .        .     .     .         .         .     read_line(FILE *fp)
   18     3     3        0     0     0         8         0     {
    .     .     .        .     .     .         .         .     #define DEF_LINE_SZ 1024
    .     .     .        .     .     .         .         .
    .     .     .        .     .     .         .         .       int c;
    6     0     0        0     0     0         4         0       size_t len = 0, tam = DEF_LINE_SZ;
    .     .     .        .     .     .         .         .       char* str;
    .     .     .        .     .     .         .         .
   14     1     1        6     1     0         2         1       str = malloc(tam);
    6     0     0        2     0     0         0         0       if (!str) {
    .     .     .        .     .     .         .         .
```

```
                    .           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       .     .     .           .     .     .           .           .
                    .           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           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)
            m->rows) < 0) {
305          .    .    .          .    .    .          .          .
                    .      perror("");
306          .    .    .          .    .    .          .          .
                    .      return -1;
307          .    .    .          .    .    .          .          .
                    .    }
308         48    2    2         17    0    0          6          0
                   0    for(i=0; i<n; i++)
309        240    2    2         85    0    0         30          0
                  0      for (j=0; j<n; j++)
310        550    3    3        250    0    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    if (fprintf(fp, "\n") < 0) {
315          .    .    .          .    .    .          .          .
                    .      perror("");
316          .    .    .          .    .    .          .          .
                    .      return -1;
317          .    .    .          .    .    .          .          .
                    .    }
318          1    1    1          0    0    0          0          0
                   0    return 0;
319          6    1    1          2    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    36   30 20,972,616 1,310,741
      1,310,739  events annotated
```

```
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
e et al.
==666== Using Valgrind-3.15.0 and LibVEX; rerun with -h for
    copyright info
==666== Command: /tmp/02-mmult
==666== Parent PID: 597
==666==
--666-- Warning: Cannot auto-detect cache config, using
    defaults.
--666--          Run with -v to see.
==666==
==666== I   refs:      147,225,941
==666== I1  misses:          2,409
==666== LLi misses:          2,389
==666== I1  miss rate:        0.00%
==666== LLi miss rate:        0.00%
==666==
==666== D   refs:       73,565,863  (52,539,344 rd   +
    21,026,519 wr)
==666== D1  misses:      1,314,850  (     3,527 rd   +
    1,311,323 wr)
==666== LLd misses:      1,314,203  (     2,925 rd   +
    1,311,278 wr)
==666== D1  miss rate:         1.8% (       0.0%     +
    6.2%  )
==666== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==666==
==666== LL refs:         1,317,259  (     5,936 rd   +
    1,311,323 wr)
==666== LL misses:       1,316,592  (     5,314 rd   +
    1,311,278 wr)
==666== LL miss rate:          0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.666
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
41 ----------------------------------------------------------------------------

42 147,225,941 2,409 2,389 52,539,344 3,527 2,925 21,026,519
       1,311,323 1,311,278  PROGRAM TOTALS
43
44 ----------------------------------------------------------------------------

45 Ir           I1mr ILmr Dr          D1mr  DLmr Dw          D1mw
         DLmw       file:function
46 ----------------------------------------------------------------------------

47 146,810,542   29   29 52,433,589    31   31 20,972,246
       1,310,728 1,310,728  /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          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          .     .     .           .     .     .           .          .
                    .    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    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) l;
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) l;
117         .    .    .          .    .    .          .          .
                 .
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          1          0
                      0       bs = n;
124         .    .    .          .    .    .          .          .
                    .   #endif
125         .    .    .          .    .    .          .          .
                    .
126         .    .    .          .    .    .          .          .
                    .       /* load matrix a */
127        11    1    1          5    1    0          1          0
                     0      if (!(a = create_matrix(n, n)))
128         .    .    .          .    .    .          .          .
                    .         goto _exit_main;
129         .    .    .          .    .    .          .          .
                    .
130       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
                    0        e = strtod(nptr, &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 (nptr == endptr) {
138         .    .    .          .    .    .          .          .
                    .          fprintf(stderr, "missing A
          matrix element");
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        nptr = endptr;
150        324     1     1         108     0     0          36          0
                   0         e = strtod(nptr, &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 (nptr == endptr) {
156          .     .     .           .     .     .           .          .
                     .           fprintf(stderr, "missing B
           matrix element");
157          .     .     .           .     .     .           .          .
                     .           goto _exit_main;
158          .     .     .           .     .     .           .          .
                     .         }
159        252     1     1         144     0     0          36          9
                   9         b->array[i] = e;
160          .     .     .           .     .     .           .          .
                     .       }
161          .     .     .           .     .     .           .          .
                     .
162          8     1     1           2     0     0           0          0
                     0       clock_gettime(CLOCK_REALTIME, &t0);
163          .     .     .           .     .     .           .          .
                     .
164          .     .     .           .     .     .           .          .
                     .       /* multiply matrixes */
165         13     2     2           6     1     1           1          0
                    0       if (!(c = matrix_multiply(a, b, bs))
          )
166          .     .     .           .     .     .           .          .
                     .         goto _exit_main;
167          .     .     .           .     .     .           .          .
                     .
168          8     1     1           2     0     0           0          0
                     0       clock_gettime(CLOCK_REALTIME, &t1);
169          .     .     .           .     .     .           .          .
                     .
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     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          .     .     .           .     .     .           .           .
                       .     double sum;
200          .     .     .           .     .     .           .           .
                       .     matrix_t* mr;
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         56     2     2          20     0     0           7           0
                       0     for(i=0; i<n; i++)
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         .     .     .            .     .     .           .          .
                    .  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     2,201     5     5          734     1     0         147          0
                  0    while (EOF != (c=fgetc(fp)) && c != '\n')
        {
251     1,160     2     2          435     0     0         290          4
```

```
                    4       str[len++]=c;
252         725     0    0          290     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           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)
```

```
             m->rows) < 0) {
305          .    .    .           .    .    .          .          .
                        .      perror("");
306          .    .    .           .    .    .          .          .
                        .      return -1;
307          .    .    .           .    .    .          .          .
                        .    }
308         56    2    2          20    0    0          7          0
                       0    for(i=0; i<n; i++)
309        336    2    2         120    0    0         42          0
                      0      for (j=0; j<n; j++)
310        792    3    3         360    0    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    if (fprintf(fp, "\n") < 0) {
315          .    .    .           .    .    .          .          .
                        .      perror("");
316          .    .    .           .    .    .          .          .
                        .      return -1;
317          .    .    .           .    .    .          .          .
                        .    }
318          1    1    1           0    0    0          0          0
                         0    return 0;
319          6    1    1           2    0    0          0          0
                         0  }
320
321 --------------------------------------------------------------------------------
322 Ir           I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
          DLmw
323 --------------------------------------------------------------------------------
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
```

```
==669== I1  misses:           2,407
==669== LLi misses:           2,389
==669== I1  miss rate:         0.00%
==669== LLi miss rate:         0.00%
==669==
==669== D   refs:       73,583,942  (52,551,909 rd   +
    21,032,033 wr)
==669== D1  misses:      1,314,875  (     3,538 rd   +
    1,311,337 wr)
==669== LLd misses:      1,314,228  (     2,936 rd   +
    1,311,292 wr)
==669== D1  miss rate:         1.8% (       0.0%     +
    6.2%  )
==669== LLd miss rate:         1.8% (       0.0%     +
    6.2%  )
==669==
==669== LL refs:         1,317,282  (     5,945 rd   +
    1,311,337 wr)
==669== LL misses:       1,316,617  (     5,325 rd   +
    1,311,292 wr)
==669== LL miss rate:          0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.669
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
--------------------------------------------------------------------------------

147,273,019 2,407 2,389 52,551,909 3,538 2,936 21,032,033
    1,311,337 1,311,292  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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:
```

```
    matrix_multiply
47
48  --------------------------------------------------------------------------------

49  -- User-annotated source: /root/CARPETA/tp2-2020-2q-src/main.c
50  --------------------------------------------------------------------------------

51  Ir          I1mr  ILmr  Dr          D1mr  DLmr  Dw          D1mw
    DLmw
52
53  -- line 18 ----------------------------------------
54           .     .     .           .     .     .           .           .
                     .  matrix_t* create_matrix(size_t rows,
               size_t cols);
55           .     .     .           .     .     .           .           .
                     .
56           .     .     .           .     .     .           .           .
                     .  void destroy_matrix(matrix_t* m);
57           .     .     .           .     .     .           .           .
                     .
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
                     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
                     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
                        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
                        0     for(; !feof(stdin); lineno++) {
81         10     1     1           4     0     0           6           0
                        0       a=b=c=NULL;
82          .     .     .           .     .     .           .           .
                        .
83         18     3     3           8     1     0           2           0
                        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          .     .     .           .     .     .           .           .
                        .       /* parse dimension */
89          2     1     1           1     0     0           1           1
                        0       nptr = line;
90         10     1     1           3     1     0           1           0
                        0       l = strtol(nptr, &endptr, 10);
91          8     1     1           3     0     0           0           0
                        0       if (errno) {
92          .     .     .           .     .     .           .           .
                        .         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         .   .   .           .   .   .         .         .
                    .   #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) 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         .   .   .           .   .   .         .         .
                    .   #endif
124         .   .   .           .   .   .         .         .
                    .
125         .   .   .           .   .   .         .         .
                    .       /* load matrix a */
126      11   1   1             5   1   0         1         0
                  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
            matrix element");
138       .    .    .           .    .    .          .          .
                    .          goto _exit_main;
139       .    .    .           .    .    .          .          .
                  .        }
140     343    1    1         196    0    0         49         11
                 11        a->array[i] = e;
141       .    .    .           .    .    .          .          .
                  .      }
142       .    .    .           .    .    .          .          .
                  .
143       .    .    .           .    .    .          .          .
                  .      /* load matrix b */
144      11    1    1           5    0    0          1          0
                   0     if (!(b = create_matrix(n, n)))
145       .    .    .           .    .    .          .          .
                    .        goto _exit_main;
146       .    .    .           .    .    .          .          .
                  .
147     500    2    2         199    0    0         50          0
                  0       for (i=0; i < n*n; i++) {
148      98    1    1          49    0    0         49          0
                   0        nptr = endptr;
149     441    1    1         147    0    0         49          0
                  0        e = strtod(nptr, &endptr);
150     392    1    1         147    0    0          0          0
                  0        if (errno) {
151       .    .    .           .    .    .          .          .
                    .          perror("");
152       .    .    .           .    .    .          .          .
                    .          goto _exit_main;
153       .    .    .           .    .    .          .          .
                  .        }
```

```
154       196     1     1          98     0     0           0          0
                  0         if (nptr == endptr) {
155         .     .     .           .     .     .           .          .
                    .           fprintf(stderr, "missing B
              matrix element");
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
                     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          .     .     .           .     .     .           .           .
                     .     destroy_matrix(b);
190          .     .     .           .     .     .           .           .
                     .     destroy_matrix(c);
191          .     .     .           .     .     .           .           .
                     .     exit(1);
192          6     1     1           2     0     0           0           0
                     0   }
193          .     .     .           .     .     .           .           .
                     .
194          .     .     .           .     .     .           .           .
                     .   matrix_t*
195          .     .     .           .     .     .           .           .
                     .   matrix_multiply(matrix_t* m1, matrix_t*
              m2, int bs)
196         11     2     2           0     0     0           6           0
                     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
                     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          .     .     .           .     .     .           .           .
                  .   #if 1
212          .     .     .           .     .     .           .           .
                  .     if (1) {
213          .     .     .           .     .     .           .           .
                  .       size_t j;
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     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          .     .     .           .     .     .           .           .
                           .     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
                    0    if (c != EOF)
261          7    0    0          2    0    0          2          0
                    0      str[len++]='\n';
262          .    .    .          .    .    .          .          .
                    .
263         12    1    1          4    0    0          4          0
                    0    str[len++]='\0';
264          2    0    0          2    0    0          0          0
                    0    return str;
265         12    1    1          4    0    0          0          0
                    0  }
266          .    .    .          .    .    .          .          .
                    .
267          .    .    .          .    .    .          .          .
                    .
268          .    .    .          .    .    .          .          .
                    .  matrix_t*
269          .    .    .          .    .    .          .          .
                    .  create_matrix(size_t rows, size_t cols)
270         30    1    1          0    0    0         15          0
                    0  {
271          .    .    .          .    .    .          .          .
                    .    matrix_t * m;
272          .    .    .          .    .    .          .          .
                    .
273         30    2    2          9    1    0          3          0
                    0    if (!(m = malloc(sizeof(matrix_t)))) {
274          .    .    .          .    .    .          .          .
                    .      perror("");
275          .    .    .          .    .    .          .          .
                    .      return NULL;
276          .    .    .          .    .    .          .          .
                    .    }
277          .    .    .          .    .    .          .          .
                    .
278          9    1    1          6    0    0          3          0
                    0    m->rows = rows;
279          9    1    1          6    0    0          3          0
                    0    m->cols = cols;
280         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  {
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  {
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             .     .     .           .     .     .           .           .
                                  .        return -1;
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           .    .    .           .    .    .           .          .
                      .             perror("");
311           .    .    .           .    .    .           .          .
                      .             return -1;
312           .    .    .           .    .    .           .          .
                      .           }
313          10    1    1           4    0    0           0          0
                     0     if (fprintf(fp, "\n") < 0) {
314           .    .    .           .    .    .           .          .
                      .        perror("");
315           .    .    .           .    .    .           .          .
                      .        return -1;
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  ==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%  )
==672==
==672== LL refs:          1,317,315  (     5,957 rd   +
    1,311,358 wr)
==672== LL misses:        1,316,646  (     5,338 rd   +
    1,311,308 wr)
==672== LL miss rate:           0.6% (       0.0%     +
    6.2%  )
--------------------------------------------------------------------------------

I1 cache:         32768 B, 32 B, 4-way associative
D1 cache:         32768 B, 32 B, 4-way associative
LL cache:         524288 B, 32 B, 8-way associative
Command:          /tmp/02-mmult
Data file:        cachegrind.out.672
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
--------------------------------------------------------------------------------

147,337,351 2,407 2,389 52,568,671 3,550 2,949 21,039,789
    1,311,358 1,311,308  PROGRAM TOTALS

--------------------------------------------------------------------------------

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

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

--------------------------------------------------------------------------------

-- 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);
```

```
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          .     .     .           .     .     .           .           .
                  .     /* 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           1           0
                    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           1           0
                    0     size_t lineno = 1;
73          .     .     .           .     .     .           .           .
                  .     struct timespec t0;
74          .     .     .           .     .     .           .           .
                  .     struct timespec t1;
75          .     .     .           .     .     .           .           .
                  .     double dt;
76          .     .     .           .     .     .           .           .
                  .     size_t i;
77          .     .     .           .     .     .           .           .
                  .
78         25     4     4           9     0     0           1           0
                    0     for(; !feof(stdin); lineno++) {
```

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

```
105           .     .     .           .     .     .           .           .
                      .       l = strtol(nptr, &endptr, 10);
106           .     .     .           .     .     .           .           .
                      .       if (errno) {
107           .     .     .           .     .     .           .           .
                      .         perror("");
108           .     .     .           .     .     .           .           .
                      .         goto _exit_main;
109           .     .     .           .     .     .           .           .
                      .       }
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
                        0       bs = n;
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
                     0      for (i=0; i < n*n; i++) {
128         128     0     0          64     0     0          64           0
                     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
                  0       if (!(b = create_matrix(n, n)))
143       .     .     .           .     .     .           .           .
                    .         goto _exit_main;
144       .     .     .           .     .     .           .           .
                    .
145     650     2     2         259     0     0          65           0
                  0       for (i=0; i < n*n; i++) {
146     128     1     1          64     0     0          64           0
                  0         nptr = endptr;
147     576     1     1         192     0     0          64           0
                  0         e = strtod(nptr, &endptr);
148     512     1     1         192     0     0           0           0
                  0         if (errno) {
149       .     .     .           .     .     .           .           .
                    .           perror("");
150       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
151       .     .     .           .     .     .           .           .
                    .         }
152     256     1     1         128     0     0           0           0
                  0         if (nptr == endptr) {
153       .     .     .           .     .     .           .           .
                    .           fprintf(stderr, "missing B
              matrix element");
154       .     .     .           .     .     .           .           .
                    .           goto _exit_main;
155       .     .     .           .     .     .           .           .
                    .         }
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          .     .     .           .     .     .           .          .
                .       /* multiply matrixes */
162         13     2     2           6     1     1           1          0
                 0       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     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*
    m2, int bs)
194 11    2     2           0     0     0           6           0
                  0   {
195 .     .     .           .     .     .           .           .
                .     size_t n, en, i, j, k, kk, jj;
196 .     .     .           .     .     .           .           .
                .     double sum;
197 .     .     .           .     .     .           .           .
                .     matrix_t* mr;
198 .     .     .           .     .     .           .           .
                .
199 3     1     1           2     0     0           1           0
                  0     n = m1->rows;
200 .     .     .           .     .     .           .           .
                .
201 11    1     1           5     0     0           1           0
                  0     if(!(mr = create_matrix(n,n))) return
    NULL;
202 .     .     .           .     .     .           .           .
                .
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     0     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  1,310,720
     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           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        .    .    .          .    .    .          .          .
                      .      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
                          0  {
290           9     0     0           3     0     0           0          0
                          0     if (!m) return;
291          24     1     1          12     0     0           0          0
                          0     free(m->array);
292          24     1     1           9     0     0           0          0
                          0     free(m);
293          18     0     0           6     0     0           0          0
                          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.