Benchmark pamäťového systému

Benchmark 1

```
#define REPEATN (32)
  register int *a;
  register int x=0,y=0;
  register int i, j;
  struct timeval t1, t2;
  float elapsedtime;
  a = (int *) malloc (1024 * 1024 * sizeof (int));
  gettimeofday (&t1, 0);
  for (j = 0; j < REPEATN; j++)
    for (i = 0; i < 1024 * 1024; i += 32) {
      //heavily unrolled loop
      a[i+0] = 7;
      a[i+1] = 7;
// ## vynechané riadky - indexy 2 až 30
      a[i+31] = 7;
    }
  gettimeofday (&t2, 0);
  elapsedtime = ((float) ((t2.tv_sec - t1.tv_sec) * 1000000  +
      (t2.tv_usec - t1.tv_usec))) / 1000000.0  ;
  printf ("WRITE: Elapsed time: %.3f, that means %.2f MB/s \n", elapsedtime,
      (float)(sizeof (int)) * REPEATN / elapsedtime);

  gettimeofday (&t1, 0);
  for (j = 0; j < REPEATN; j++)
    for (i = 0; i < 1024 * 1024; i+=32) {
      //another heavily unrolled loop
      x += a[i+0]; y += a[i+1];
      x += a[i+2]; y += a[i+3];
// ## vynechané riadky - indexy 4 až 29
      x += a[i+30]; y += a[i+31];
    }
  gettimeofday (&t2, 0);
  elapsedtime = ((float) ((t2.tv_sec - t1.tv_sec) * 1000000  +
      (t2.tv_usec - t1.tv_usec))) / 1000000.0  ;
  printf ("READ: Elapsed time: %.3f, that means %.2f MB/s\n", elapsedtime,
      (float)(sizeof (int)) * REPEATN / elapsedtime);
```

Benchmark 2

```
#define REPEATN (8*1024)
#define DATASIZE (1024*4)
#define TYPE int
  register TYPE *a, *b;
  register TYPE x=0,y=0;
  register int i, j;
  struct timeval t1, t2;
  float elapsedtime;
  a = (TYPE *) malloc (DATASIZE  * sizeof (TYPE));
  b = (TYPE *) malloc (DATASIZE  * sizeof (TYPE));

  gettimeofday (&t1, 0);
  for (j = 0; j < REPEATN; j++)
  {
    for (i = 0; i < DATASIZE ; i += 4) {
      a[i+0] = a[DATASIZE -i]+b[DATASIZE -i];
      a[i+1] = a[DATASIZE -i]+b[DATASIZE -i];
      a[i+2] = a[DATASIZE -i]+b[DATASIZE -i];
      a[i+3] = a[DATASIZE -i]+b[DATASIZE -i];
    }
    TYPE *c = a; a = b; b = c;
  }
  gettimeofday (&t2, 0);
  elapsedtime = ((float) ((t2.tv_sec - t1.tv_sec) * 1000000  +
      (t2.tv_usec - t1.tv_usec))) / 1000000.0  ;
  printf ("READ+WRITE: Elapsed time: %.3f, that means %.2f MB/s \n", elapsedtime,
      (float)(sizeof (TYPE)) * DATASIZE  / (1024*1024) * REPEATN / elapsedtime);
```

Benchmark siete - sockety

Inicializácia socketov vynechaná, použité boli štandardné UNIXovské volania - socket, bind, open, listen, accept, close

Časť 1 - klient

```
//how much to transfer?
#define TRANSFER 1e8

#define DATASIZE 16384
  total = 0;
  while (total < TRANSFER ) {
    nbytes = send (sock, bordel, DATASIZE , 0);
    total += nbytes;
  }
  return 0;
```

Časť 2 - server

```
  total = 0;
  while (total < TRANSFER ) {
    nbytes = read (sock2, bordel, DATASIZE );
    total += nbytes;
  }
```

benchmark siete - MPI

Nepodstatné časti sú vynechané.

```
  //initialize MPI
  MPI_Init  (&argc, &argv);
  MPI_Comm_rank  (MPI_COMM_WORLD , &rank);
  MPI_Comm_size  (MPI_COMM_WORLD , &size);
  if (rank == 0) master = 1; else master = 0;

  for (; dsize <= 1024*1024*128; dsize *= 2) {
    data = (char *) malloc (dsize);
    gettimeofday (&t1, 0);
    cnt = 0;
    elapsedtime = 0.0;
    while (elapsedtime < 2.0) { //ensure that we test for at least 2 seconds
      cnt++;
      if (master)
        MPI_Send  (data, dsize, MPI_CHAR, 1, 0, MPI_COMM_WORLD );
      else
        MPI_Recv  (data, dsize, MPI_CHAR, 0, 0, MPI_COMM_WORLD , MPI_STATUS_IGNORE  );
      gettimeofday (&t2, 0);
      elapsedtime = ((float) ((t2.tv_sec - t1.tv_sec) * 1000000  +
          (t2.tv_usec - t1.tv_usec))) / 1000000.0  ;
    }
```