Final Year B. Tech., Sem VII 2022-23

High Performance Computing Lab

PRN: 2020BTECS00206

Full Name: SAYALI YOGESH DESAI

Batch: B4

Assignment No. 9

---

1. **Implement Vector-Vector addition using CUDA C. State and justify the speedup using different size of threads and blocks.**

```
#include <stdio.h>

/*
 * Host function to initialize vector elements. This function
 * simply initializes each element to equal its index in the
 * vector.
 */

void initWith(float num, float *a, int N)
{
  for(int i = 0; i < N; ++i)
  {
    a[i] = num;
  }
}

/*
 * Device kernel stores into `result` the sum of each
 * same-indexed value of `a` and `b`.
 */
```

```
__global__
void addVectorsInto(float *result, float *a, float *b, int N)
{
  int index = threadIdx.x + blockIdx.x * blockDim.x;
  int stride = blockDim.x * gridDim.x;

  for(int i = index; i < N; i += stride)
  {
    result[i] = a[i] + b[i];
  }
}

/*
 * Host function to confirm values in `vector`. This function
 * assumes all values are the same `target` value.
 */

void checkElementsAre(float target, float *vector, int N)
{
  for(int i = 0; i < N; i++)
  {
    if(vector[i] != target)
    {
      printf("FAIL: vector[%d] - %0.0f does not equal %0.0f\n", i, vector[i], target);
      exit(1);
    }
  }
  printf("Success! All values calculated correctly.\n");
}

int main()
{
```

```
const int N = 2<<24;
size_t size = N * sizeof(float);

float *a;
float *b;
float *c;

cudaMallocManaged(&a, size);
cudaMallocManaged(&b, size);
cudaMallocManaged(&c, size);

initWith(3, a, N);
initWith(4, b, N);
initWith(0, c, N);

size_t threadsPerBlock;
size_t numberOfBlocks;

/*
 * nsys should register performance changes when execution configuration
 * is updated.
 */

threadsPerBlock = 1;
numberOfBlocks = 1;

cudaError_t addVectorsErr;
cudaError_t asyncErr;

addVectorsInto<<<numberOfBlocks, threadsPerBlock>>>(c, a, b, N);

addVectorsErr = cudaGetLastError();
```

```
  if(addVectorsErr != cudaSuccess) printf("Error: %s\n",
cudaGetErrorString(addVectorsErr));

  asyncErr = cudaDeviceSynchronize();
  if(asyncErr != cudaSuccess) printf("Error: %s\n", cudaGetErrorString(asyncErr));

  checkElementsAre(7, c, N);

  cudaFree(a);
  cudaFree(b);
  cudaFree(c);
}
```

N=2<<16

Serial Execution Time: 17944640

```
 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  --------------------
    93.2        254696095          3  84898698.3      4141  254670295  cudaMallocManaged
     6.6         17955956          1  17955956.0  17955956   17955956  cudaDeviceSynchronize
     0.2           566454          3    188818.0     33156     416960  cudaFree
     0.0            39346          1     39346.0     39346      39346  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                      Name
 -------  ---------------  ---------  ----------  --------  --------  ------------------------------------------
   100.0         17944640          1  17944640.0  17944640  17944640  addVectorsInto(float*, float*, float*, int)
```

Parallel Execution Time:

Number of Blocks	Threads per Block	Time (in ns)	Speedup
8	256	1305172	13.7488
16	256	1309816	13.7001
32	256	1406457	12.7587
8	512	1567085	11.4509
16	512	1391095	12.8996
132	512	1384345	12.9625

Number of blocks: 8, Threads per block: 256, Time Required: 1305172

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.3        242232114          3  80744038.0     3776  242214856  cudaMallocManaged
     0.5          1308578          1   1308578.0  1308578    1308578  cudaDeviceSynchronize
     0.1           254964          3     84988.0    10490     212916  cudaFree
     0.0            35428          1     35428.0    35428      35428  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1305172          1  1305172.0  1305172  1305172  addVectorsInto(float*, float*, float*, int)
```

Number of blocks: 16, Threads per block: 256, Time Required: 1309816

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.4        253585855          3  84528618.3     3258  253574212  cudaMallocManaged
     0.5          1312041          1   1312041.0  1312041    1312041  cudaDeviceSynchronize
     0.1           250346          3     83448.7    10620     208049  cudaFree
     0.0            40880          1     40880.0    40880      40880  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1309816          1  1309816.0  1309816  1309816  addVectorsInto(float*, float*, float*, int)
```

Number of blocks: 32, Threads per block: 256, Time Required: 1406457

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.4        257983525          3  85994508.3     3841  257967279  cudaMallocManaged
     0.5          1410940          1   1410940.0  1410940    1410940  cudaDeviceSynchronize
     0.1           243629          3     81209.7     9975     202158  cudaFree
     0.0            29226          1     29226.0    29226      29226  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1406457          1  1406457.0  1406457  1406457  addVectorsInto(float*, float*, float*, int)
```

Number of blocks: 8, Threads per block: 512, Time Required: 1567085

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.3        258789829          3  86263276.3     6193  258766685  cudaMallocManaged
     0.6          1587219          1   1587219.0  1587219    1587219  cudaDeviceSynchronize
     0.1           325685          3    108561.7    13740     275675  cudaFree
     0.0            42591          1     42591.0    42591      42591  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1567085          1  1567085.0  1567085  1567085  addVectorsInto(float*, float*, float*, int)
```

Number of blocks: 16, Threads per block: 512, Time Required: 1391095

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  --------------------
    99.4        302167922          3  100722640.7     7493  302136720  cudaMallocManaged
     0.4          1351968          1    1351968.0  1351968    1351968  cudaDeviceSynchronize
     0.2           470686          3     156895.3    20743     398779  cudaFree
     0.0           135244          1     135244.0   135244     135244  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                     Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1391095          1  1391095.0  1391095  1391095  addVectorsInto(float*, float*, float*, int)
```

Number of blocks: 32, Threads per block: 512, Time Required: 1384345

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  --------------------
    99.3        249471482          3  83157160.7     3935  249453873  cudaMallocManaged
     0.6          1388012          1   1388012.0  1388012    1388012  cudaDeviceSynchronize
     0.1           234438          3     78146.0    10113     191420  cudaFree
     0.0            30378          1     30378.0    30378      30378  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                     Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          1384345          1  1384345.0  1384345  1384345  addVectorsInto(float*, float*, float*, int)
```

N= 2<<24

Serial Execution Time: 2117915980

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average      Minimum     Maximum           Name
 -------  ---------------  ---------  ------------  ----------  ----------  ---------------------
    88.6       2175923366          1  2175923366.0  2175923366  2175923366  cudaDeviceSynchronize
    10.1        249249035          3    83083011.7       18262   249160342  cudaMallocManaged
     0.9         21165881          3     7055293.7     6105795     8534822  cudaFree
     0.4         10211381          3     3403793.7        7661    10050536  cudaMemPrefetchAsync
     0.0            38289          1       38289.0       38289       38289  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average      Minimum     Maximum                      Name
 -------  ---------------  ---------  ------------  ----------  ----------  ----------------------------------------
   100.0       2117915980          1  2117915980.0  2117915980  2117915980  addVectorsInto(float*, float*, float*, int)
```

Parallel Execution Time:

Number of Blocks	Threads per Block	Time (in ns)	Speedup
8	256	10659168	198.6943
16	256	5827432	363.4389
32	256	3360477	630.2426
8	512	6078953	348.4014
16	512	3496636	605.7010
132	512	2148469	985.7791

Threads per block: 256, Number of blocks: 8, Time Required: 10659168

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    71.5        253195240          3  84398413.3     17526  253123280  cudaMallocManaged
    12.7         45030822          1  45030822.0  45030822   45030822  cudaDeviceSynchronize
     9.4         33370308          3  11123436.0    165062   22466204  cudaMemPrefetchAsync
     6.3         22349575          3   7449858.3   6700811    8705098  cudaFree
     0.0            36668          1     36668.0     36668      36668  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                     Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------------------
   100.0         10659168          1  10659168.0  10659168  10659168  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 256, Number of blocks: 16, Time Required: 5827432

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    71.6        239709717          3  79903239.0     35708  239568662  cudaMallocManaged
    11.2         37333789          1  37333789.0  37333789   37333789  cudaDeviceSynchronize
    10.8         36235467          3  12078489.0    135113   22478737  cudaMemPrefetchAsync
     6.4         21345459          3   7115153.0   6286294    8561414  cudaFree
     0.0            38040          1     38040.0     38040      38040  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                     Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          5827432          1  5827432.0  5827432  5827432  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 256, Number of blocks: 32, Time Required: 3360477

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    73.9        262692662          3  87564220.7     19089  262609446  cudaMallocManaged
    10.7         37900928          3  12633642.7    158259   22605477  cudaMemPrefetchAsync
     9.4         33240782          1  33240782.0  33240782   33240782  cudaDeviceSynchronize
     6.0         21479652          3   7159884.0   6268936    8659407  cudaFree
     0.0            41117          1     41117.0     41117      41117  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                     Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          3360477          1  3360477.0  3360477  3360477  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 512, Number of blocks: 8, Time Required: 6078953

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    79.2        366247859          3  122082619.7     24337  366138814  cudaMallocManaged
     9.0         41369623          1   41369623.0  41369623   41369623  cudaDeviceSynchronize
     7.0         32442406          3   10814135.3    149806   22495493  cudaMemPrefetchAsync
     4.8         22047012          3    7349004.0   6603656    8798783  cudaFree
     0.0            39697          1      39697.0     39697      39697  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          6078953          1  6078953.0  6078953  6078953  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 512, Number of blocks: 16, Time Required: 3496636

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    72.7        251162089          3  83720696.3     35468  250999154  cudaMallocManaged
    17.7         61022632          1  61022632.0  61022632   61022632  cudaDeviceSynchronize
     6.7         23073097          3   7691032.3   6258226    8622708  cudaFree
     2.9         10126663          3   3375554.3      7856    9972133  cudaMemPrefetchAsync
     0.0            36285          1     36285.0     36285      36285  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          3496636          1  3496636.0  3496636  3496636  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 512, Number of blocks: 32, Time Required: 2148469

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    73.7        258462091          3  86154030.3     19090  258371602  cudaMallocManaged
    10.7         37459080          1  37459080.0  37459080   37459080  cudaDeviceSynchronize
     9.3         32494474          3  10831491.3    147396   22518420  cudaMemPrefetchAsync
     6.4         22437186          3   7479062.0   6554815    8612458  cudaFree
     0.0            40541          1     40541.0     40541      40541  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------
   100.0          2148469          1  2148469.0  2148469  2148469  addVectorsInto(float*, float*, float*, int)
```

N= 2<<10

Serial Execution Time: 266554

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  --------------------
    99.8        427305624          3  142435208.0     7724  427266355  cudaMallocManaged
     0.1           456341          3     152113.7    20364     404617  cudaMemPrefetchAsync
     0.1           267087          1     267087.0   267087     267087  cudaDeviceSynchronize
     0.0           153975          3      51325.0    17898      99035  cudaFree
     0.0            39164          1      39164.0    39164      39164  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                    Name
 -------  ---------------  ---------  --------  -------  -------  ----------------------------------------
   100.0           266554          1  266554.0   266554   266554  addVectorsInto(float*, float*, float*, int)
```

Parallel Execution Time:

Number of Blocks	Threads per Block	Time (in ns)	Speedup
8	256	5440	48.9988
16	256	5216	51.1031
32	256	5344	49.8791
8	512	5664	47.0610
16	512	5472	48.7123
132	512	5665	47.0527

Threads per block: 256, Number of blocks: 8, Time Required: 5440

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.8        288048459          3  96016153.0     5528  288016841  cudaMallocManaged
     0.1           413841          3    137947.0    18716     370145  cudaMemPrefetchAsync
     0.0           131855          3     43951.7    15804      88695  cudaFree
     0.0            36206          1     36206.0    36206      36206  cudaLaunchKernel
     0.0             7962          1      7962.0     7962       7962  cudaDeviceSynchronize


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5440          1   5440.0     5440     5440  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 256, Number of blocks: 16, Time Required: 5216

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.7        274279608          3  91426536.0     3638  274261154  cudaMallocManaged
     0.1           374808          1    374808.0   374808     374808  cudaDeviceSynchronize
     0.1           355617          3    118539.0    18989     239004  cudaMemPrefetchAsync
     0.0           108074          3     36024.7    11639      73272  cudaFree
     0.0            62709          1     62709.0    62709      62709  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5216          1   5216.0     5216     5216  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 256, Number of blocks: 32, Time Required: 5344

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.8        254890908          3  84963636.0     5617  254856180  cudaMallocManaged
     0.2           433121          3    144373.7    20363     387395  cudaMemPrefetchAsync
     0.1           141081          3     47027.0    15056      94988  cudaFree
     0.0            40716          1     40716.0    40716      40716  cudaLaunchKernel
     0.0             8131          1      8131.0     8131       8131  cudaDeviceSynchronize


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5344          1   5344.0     5344     5344  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 512, Number of blocks: 8, Time Required: 5664

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  --------------------
    99.6        256999308          3  85666436.0     4304  256965354  cudaMallocManaged
     0.1           377884          1    377884.0   377884     377884  cudaDeviceSynchronize
     0.1           373425          3    124475.0    19140     250564  cudaMemPrefetchAsync
     0.0           114766          3     38255.3    11556      75299  cudaFree
     0.0            65032          1     65032.0    65032      65032  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5664          1   5664.0     5664     5664  addVectorsInto(float*, float*, float*, int)
```

Threads per block: 512, Number of blocks: 16, Time Required: 5472

```
CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5472          1   5472.0     5472     5472  addVectorsInto(float*, float*, float*, int)


CUDA Memory Operation Statistics (by time):

 Time(%)  Total Time (ns)  Operations  Average  Minimum  Maximum             Operation
 -------  ---------------  ----------  -------  -------  -------  --------------------------------
    56.6             8640           3   2880.0     2880     2880  [CUDA Unified Memory memcpy HtoD]
    43.4             6622           3   2207.3     1632     3135  [CUDA Unified Memory memcpy DtoH]
```

Threads per block: 512, Number of blocks: 32, Time Required: 5665

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  --------------------
    99.7        304684255          3  101561418.3     8513  304635437  cudaMallocManaged
     0.2           566573          3     188857.7    26426     504410  cudaMemPrefetchAsync
     0.1           172369          3      57456.3    21393     112972  cudaFree
     0.0            56456          1      56456.0    56456      56456  cudaLaunchKernel
     0.0            10349          1      10349.0    10349      10349  cudaDeviceSynchronize


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average  Minimum  Maximum                    Name
 -------  ---------------  ---------  -------  -------  -------  ----------------------------------------
   100.0             5665          1   5665.0     5665     5665  addVectorsInto(float*, float*, float*, int)
```

Conclusion:

From above graphs we can conclude that

1. For vector-vector addition problem for larger values of N time for parallel execution decreases exponentially as we increase number of threads
2. For vector-vector addition problem for smaller values of N time first decreases and then increases exponentially as we increase number of threads
3. Parallel execution is preferred for higher values of N

4. Implement N-Body Simulator using CUDA C. State and justify the speedup using different size of threads and blocks.

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "timer.h"
#include "files.h"
#define SOFTENING 1e-9f

/*
 * Each body contains x, y, and z coordinate positions,
 * as well as velocities in the x, y, and z directions.
 */
typedef struct { float x, y, z, vx, vy, vz; } Body;
/*
 * Calculate the gravitational impact of all bodies in the system
 * on all others.
 */
```

```
__global__ void bodyForce(Body *p, float dt, int N) {
 int tid = blockIdx.x * blockDim.x + threadIdx.x;

  if (tid < N) {
    float Fx = 0, Fy = 0, Fz = 0;
    for (int i = 0; i < N; i++) {
      float dx = p[i].x - p[tid].x;
      float dy = p[i].y - p[tid].y;
      float dz = p[i].z - p[tid].z;
      float distSqr = dx*dx + dy*dy + dz*dz + SOFTENING;
      float invDist = rsqrtf(distSqr);
      float invDist3 = invDist * invDist * invDist;
      Fx += dx * invDist3;
      Fy += dy * invDist3;
      Fz += dz * invDist3;
    }
    p[tid].vx += dt*Fx;
    p[tid].vy += dt*Fy;
    p[tid].vz += dt*Fz;
  }
}

int main(const int argc, const char** argv) {

  // The assessment will test against both 2<11 and 2<15.
  // Feel free to pass the command line argument 15 when you generate ./nbody report files
  int nBodies = 2<<11;
```

```
if (argc > 1) nBodies = 2<<atoi(argv[1]);

// The assessment will pass hidden initialized values to check for correctness.
// You should not make changes to these files, or else the assessment will not work.
const char * initialized_values;
const char * solution_values;

if (nBodies == 2<<11) {
  initialized_values = "09-nbody/files/initialized_4096";
  solution_values = "09-nbody/files/solution_4096";
} else { // nBodies == 2<<15
  initialized_values = "09-nbody/files/initialized_65536";
  solution_values = "09-nbody/files/solution_65536";
}

if (argc > 2) initialized_values = argv[2];
if (argc > 3) solution_values = argv[3];

const float dt = 0.01f; // Time step
const int nIters = 10;  // Simulation iterations

int bytes = nBodies * sizeof(Body);
float *buf;

//buf = (float *)malloc(bytes);
cudaMallocManaged(&buf, bytes);

Body *p = (Body*)buf;
```

```
read_values_from_file(initialized_values, buf, bytes);
double totalTime = 0.0;

/*
 * This simulation will run for 10 cycles of time, calculating gravitational
 * interaction amongst bodies, and adjusting their positions to reflect.
 */

for (int iter = 0; iter < nIters; iter++) {
  StartTimer();
/*
 * You will likely wish to refactor the work being done in `bodyForce`,
 * and potentially the work to integrate the positions.
 */

  //int threads_per_block = 128;
  //int number_of_blocks = (nBodies / threads_per_block);
  int threads_per_block = 1;
  int number_of_blocks = 1;

  bodyForce <<< number_of_blocks, threads_per_block >>> ( p, dt, nBodies );
  //bodyForce<<<>>>(p, dt, nBodies); // compute interbody forces
   cudaDeviceSynchronize();
/*
 * This position integration cannot occur until this round of `bodyForce` has completed.
 * Also, the next round of `bodyForce` cannot begin until the integration is complete.
 */
```

```
  for (int i = 0 ; i < nBodies; i++) { // integrate position
    p[i].x += p[i].vx*dt;
    p[i].y += p[i].vy*dt;
    p[i].z += p[i].vz*dt;
  }

  const double tElapsed = GetTimer() / 1000.0;
  totalTime += tElapsed;
 }

 double avgTime = totalTime / (double)(nIters);
 float billionsOfOpsPerSecond = 1e-9 * nBodies * nBodies / avgTime;
 write_values_to_file(solution_values, buf, bytes);
 // You will likely enjoy watching this value grow as you accelerate the application,
 // but beware that a failure to correctly synchronize the device might result in
 // unrealistically high values.
 printf("%0.3f Billion Interactions / second\n", billionsOfOpsPerSecond);

 cudaFree(buf);
}
```

N=2<<11

Serial Time Execution: 7951112

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    96.8        244886108          1  244886108.0  244886108  244886108  cudaMallocManaged
     3.2          7982348         10     798234.8     726906    1121157  cudaDeviceSynchronize
     0.0           109377          1     109377.0     109377     109377  cudaFree
     0.0           102904         10      10290.4       6602      33009  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum              Name
 -------  ---------------  ---------  --------  -------  -------  ----------------------------
   100.0          7951112         10  795111.2   723736  1118516  bodyForce(Body*, float, int)


CUDA Memory Operation Statistics (by time):

 Time(%)  Total Time (ns)  Operations  Average  Minimum  Maximum              Operation
 -------  ---------------  ----------  -------  -------  -------  ---------------------------------
    53.2           255563          46   5555.7     2175    11616  [CUDA Unified Memory memcpy HtoD]
    46.8           225169          60   3752.8     1439    15200  [CUDA Unified Memory memcpy DtoH]
```

Parallel Execution Time:

Number of Blocks	Threads per Block	Time (in ns)	Speedup
2	128	8899145	0.8934
2	1024	20392480	0.3899
32	128	8958927	0.8875
32	1024	18692193	0.4253

Number of blocks: 2, Threads per block: 128, Execution Time: 8899145

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    96.5        250657886          1  250657886.0  250657886  250657886  cudaMallocManaged
     3.4          8932173         10     893217.3     812619    1219339  cudaDeviceSynchronize
     0.0           111204         10      11120.4       6712      33727  cudaLaunchKernel
     0.0           110976          1     110976.0     110976     110976  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum             Name
 -------  ---------------  ---------  --------  -------  -------  ----------------------------
   100.0          8899145         10  889914.5   808952  1216756  bodyForce(Body*, float, int)
```

Number of blocks: 2, Threads per block: 1024, Execution Time: 20392480

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    92.6        260060741          1  260060741.0  260060741  260060741  cudaMallocManaged
     7.3         20438083         10    2043808.3    1783428    3936978  cudaDeviceSynchronize
     0.1           191800         10      19180.0      10666      34932  cudaLaunchKernel
     0.0           113511          1     113511.0     113511     113511  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum             Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------
   100.0         20392480         10  2039248.0  1780751  3927867  bodyForce(Body*, float, int)
```

Number of blocks: 32, Threads per block: 128, Execution Time: 8958927

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    96.8        286120581          1  286120581.0  286120581  286120581  cudaMallocManaged
     3.0          8876571         10     887657.1     814539    1069596  cudaDeviceSynchronize
     0.1           351827         10      35182.7      12872     127660  cudaLaunchKernel
     0.1           258560          1     258560.0     258560     258560  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum             Name
 -------  ---------------  ---------  --------  -------  -------  ----------------------------
   100.0          8958927         10  895892.7   849304  1138966  bodyForce(Body*, float, int)
```

Number of blocks: 32, Threads per block: 1024, Execution Time: 18692193

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    92.7        241462644          1  241462644.0  241462644  241462644  cudaMallocManaged
     7.2         18727069         10    1872706.9    1790195    2501846  cudaDeviceSynchronize
     0.1           133325         10      13332.5       8815      35724  cudaLaunchKernel
     0.0           123531          1     123531.0     123531     123531  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum             Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------
   100.0         18692193         10  1869219.3  1785937  2498186  bodyForce(Body*, float, int)
```

N=2<<15

Serial Execution Time: 109233415

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    69.9        255879749          1  255879749.0  255879749  255879749  cudaMallocManaged
    29.9        109301730         10   10930173.0   10846567   11419808  cudaDeviceSynchronize
     0.1           467823          1     467823.0     467823     467823  cudaFree
     0.1           291833         10      29183.3      22028      43391  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum             Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------
   100.0        109233415         10  10923341.5  10841446  11413722  bodyForce(Body*, float, int)
```

Parallel Execution Time:

Number of Blocks	Threads per Block	Time (in ns)	Speedup
2	128	120612160	0.9056
2	1024	229499125	0.0888
32	128	116887945	0.9345
32	1024	264315658	0.4132

Number of blocks: 2, Threads per block: 128, Execution Time: 120612160

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------
    68.7        265590539          1  265590539.0  265590539  265590539  cudaMallocManaged
    31.2        120673622         10   12067362.2   11771548   12628154  cudaDeviceSynchronize
     0.1           329526          1     329526.0     329526     329526  cudaFree
     0.1           260968         10      26096.8      22295      36503  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum              Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------
   100.0        120612160         10  12061216.0  11764844  12621883  bodyForce(Body*, float, int)
```

Number of blocks: 2, Threads per block: 1024, Execution Time: 229499125

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------
    53.3        262847173          1  262847173.0  262847173  262847173  cudaMallocManaged
    46.5        229594645         10   22959464.5   19688324   27692528  cudaDeviceSynchronize
     0.1           437803         10      43780.3      38819      51465  cudaLaunchKernel
     0.1           342831          1     342831.0     342831     342831  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum              Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------
   100.0        229499125         10  22949912.5  19680193  27680088  bodyForce(Body*, float, int)
```

Number of blocks: 32, Threads per block: 128, Execution Time: 116887945

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------
    78.2        423221770          1  423221770.0  423221770  423221770  cudaMallocManaged
    21.6        116942222         10   11694222.2   11591034   12261782  cudaDeviceSynchronize
     0.1           401549         10      40154.9      21260      98015  cudaLaunchKernel
     0.1           311617          1     311617.0     311617     311617  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum              Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------
   100.0        116887945         10  11688794.5  11585674  12261083  bodyForce(Body*, float, int)
```

Number of blocks: 32, Threads per block: 1024, Execution Time: 264315658

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  --------------------
    50.7        263706704         10   26370670.4   25506033   27620883  cudaDeviceSynchronize
    49.1        255184018          1  255184018.0  255184018  255184018  cudaMallocManaged
     0.1           511319         10      51131.9      34002     107947  cudaLaunchKernel
     0.1           361911          1     361911.0     361911     361911  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum              Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------
   100.0        264315658         10  26431565.8  25953208  27610805  bodyForce(Body*, float, int)
```

Conclusion:

1. From above graph we can conclude that
2. For nbody problem parallel execution time first decreases exponentially and then increases as we increase number of threads