# Introduction to Parallel Processing

Lecturer: Dr. Guy Tel-Zur

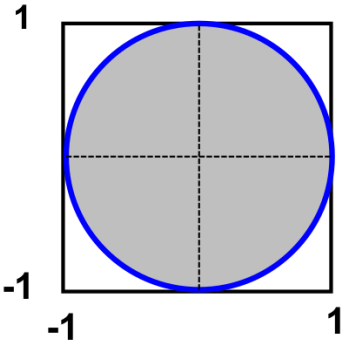
home assignment #1

Topic: Embarrassingly Parallel Computations with MPI
Goal: Estimate $\pi$ by Monte-Carlo computations

Estimate $\pi$ using the following algorithm:

**Throw darts at a square:**
- **Sample x & y randomly on (-1,1)**
- **If $x^2 + y^2 < 1$, tally a hit**

**$\pi \sim 4 * [\#\ hits] / [\#\ tries]$**




- Use MPI to parallelize your code.

- Execute your code on the **virtual machine**

A) Chose the number of tries such that $\pi$ will be accurate to the 4 digit after the decimal point.
B) Time your code using **`MPI_Wtime()`**.
C) Make use of the **`rand()`** function to generate uniform pseudo-random numbers and use the **`srand()`** function to make the seed unique for each MPI task.
D) Repeat the execution for 1,2,4,6, and 8 MPI tasks.
   1) Plot the speedup vs. number of tasks.
   2) Plot the efficiency vs. number of tasks.
E) How many darts per second your best execution had reached? Compare your result with the results in the figure below.

What to submit:
1) Your code.
2) Instructions how to compile and execute your code.
3) The plots that are requested in item D above.
4) Screen capture of your code tracing and profiling using **Jumpshot** and **Scalasca** respectively as was demonstrated in the lab.
5) Your performance result (item E above).
6) Conclusions!

Due:
In 2 weeks.

Good luck!

# Monte Carlo Darts Game (2)

| Year/Place | Machine | Darts / sec |
|---|---|---|
| 1981 LANL | CDC-7600 | 0.18 M |
| 1981 LANL | Cray-1 | 0.40 M |
| 1982 Mich | HP-11C | 1 |
| 1982 Mich | Apple II+ | 34 |
| 1982 Mich | Amdahl 470V/8 | 0.17 M |
| 1982 KAPL | Cyber-205, scalar | 0.74 M |
| 1982 KAPL | Cyber-205, vector | 9.83 M |
| 1999 Mich | 233 M PC | 0.20 M |
| 1999 Mich | 100 M PC | 0.07 M |
| 1999 Mich | 200 M Pentium, Matlab | 446 |
| 2002 Mich | 900 M P3, Matlab | 0.35 M |
| 2002 Mich | 900 M P3, Matlab, vec | 1.25 M |
| 2002 LANL | 1.2 G P3 | 11 M |
| 2005 LANL | 1.0 G P3 | 19 M |
| 2005 LANL | 2.0 G AMD Opteron | 24 M |
| 2005 LANL | 1.7 G PowerPC G4 | 32 M |
| 2005 LANL | 1.2 G Alpha EV68 | 101 M |
| 2005 LANL | 2.6 G PowerPC G5 | 140 M |

| Year/Place | Machine | Darts / sec |
|---|---|---|
| 2010 LANL | 2.6 G i7 2-core, Matlab | 0.8 M |
| 2010 LANL | 2.6 G i7 2-core | 124 M |
| 2010 LANL | 2.6 G i7 2-core *** | 410 M |
| 2010 LANL | 3.0 G 2 Xeon 4-core, 1 thread *** | 189 M |
| 2010 LANL | 3.0 G 2 Xeon 4-core, 8-thread *** | 1460 M |
| 2011 Mich | Linux cluster, MPI, 32 cpu | 2000 M |
| 2013 LANL | 3.0 G i7 2-core 2-HT | 142 M |
| 2013 LANL | 3.0 G i7 2-core 2-HT, 1 thread *** | 518 M |
| 2013 LANL | 3.0 G i7 2-core 2-HT, 2 threads *** | 920 M |
| 2013 LANL | 3.0 G i7 2-core 2-HT, 4 threads *** | 1025 M |
| 2014 LANL | 2.4 G 2 i7 4-core, 2-HT, 1 threads *** | 194 M |
| 2014 LANL | 2.4 G 2 i7 4-core, 2-HT, 8 threads *** | 1448 M |
| 2014 LANL | 2.4 G 2 i7 4-core, 2-HT, 16 threads *** | 2037 M |
| 2014 LANL | 2.7 G Xeon 12-core, 2-HT, 12 thrd *** | 2670 M |
| 2014 LANL | 2.7 G Xeon 12-core, 2-HT, 24 thrd *** | 4000 M |
| 2016 LANL | 2.7 G Xeon 12-core, 2-HT, 24 thrd *** | 5800 M |

*** = hand-tuned, highly optimized
M = MHz, clock speed
G = GHz, clock speed
HT = hyperthreads / core
Fortran, a few Matlab

Note that CPUs, architecture, and compilers all change over time, so that CPU clock speed is not always a good measure of the performance of an application code. This particular comparison is sensitive to 64-bit integer operations (CPU & compiler) and is not necessarily a good predictor of overall Monte Carlo code performance.