

UN5390: Scientific Computing I

Dr. Gowtham

Director of Research Computing, IT
Adj. Asst. Professor, Physics and ECE

EERC B39 · [\(906\) 487-4096](tel:(906)487-4096) · g@mtu.edu · [@sgowtham](https://twitter.com/@sgowtham)

Week #13: 2016/11/29 and 2016/12/01

Cross-listed as BE5390, EE5390 and MA5390

Do not share/distribute the course material, in and/or outside of Michigan Tech, without instructor's prior consent



Recap

What we did last week, and what you were supposed to do



<http://dilbert.com/strip/1998-09-14/>

- * Parallel computing
- * Message Passing Interface
- * OpenMP and MATLAB
- * Parallel computing examples
 - * Hello, World!

Week #11 Before we meet again

- * Review the syllabus, course material, grade through week #11, notations, active participation, free time exercises, tips, opportunities, mathematical results, videos, and training camps
- * Complete assignment #07
- * Make progress in the term project
- * Turn in the project status by Friday 4:59 pm (weeks #11 & #12)
- * Make progress towards Research Marketing III
- * Estimate amount of work necessary to earn desired/possible grade

- * No class during week #12

- Turn in the project status by Friday 4:59 pm

- Get some rest, catch up on life and other courses

- Get started on and make progress in assignment #12

- * 'Attend' SC16, if interested

- @SuperComputing | #HPCMatters | #SuperComputing | #SC16

- * Have a happy and safe Thanksgiving!

Parallel Computing Examples

OpenMP (C) programs



<http://dilbert.com/strip/2000-07-11/>

OpenMP sample programs

In-class activity

To be performed in `colossus.it`

```
cd ${UN5390}  
git pull  
cd CourseWork/Week_11/AdditionalMaterial  
rsync -avhP ./Parallel/ ../../${USER}_11/Parallel/  
  
cd ${UN5390}/CourseWork/Week_11/  
git add ${USER}_11  
git commit -m "PM: OpenMP samples #13"  
git push origin master
```

Array manipulation

Journal of Failed Experiments

Input validation, memory pre-allocation, row/column major language, zero is not really zero, array indexing, etc.

- * Populate two arrays, $A[N]$ and $B[N]$
- * Perform arithmetic manipulation to generate another array, $C[N]$

$A[1]$

$B[1]$

$C[1] = A[1] + B[1]$

$A[2]$

$B[2]$

$C[2] = A[2] + B[2]$

$A[N]$

$B[N]$

$C[N] = A[N] + B[N]$

Array manipulation

Compilation and execution instructions (serial; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="ArrayManipulation"  
VER="s"  
GCC="gcc -Wall -g -pg -lm"  
  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
$(pwd)/${BASENAME}_${VER}.x
```

`ArrayManipulation.s.c` is included in week #11 AdditionalMaterial.

Array manipulation

Compilation and execution instructions (parallel; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel
BASENAME="ArrayManipulation_omp_"
VER="01" # Change this to 02 for version #02
GCC="gcc -Wall -g -lm -fopenmp"

# Compile, and run using 1, 2, 4, 8 threads
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x
for x in $(seq 0 1 3)
do
    OMP_NUM_THREADS=$(echo "2^$x" | bc)
    ${pwd}/${BASENAME}_${VER}.x
done
```

`ArrayManipulation_omp_0*.c` are included in week #11 AdditionalMaterial.

Array manipulation

Pseudo-code; serial version

Define N , $A[N]$, $B[N]$, $C[N]$, i

Array population and initialization

LOOP BEGINS: i starts at 0 and does not exceed N

Populate $A[i] = i$ and $B[i] = i + 1$

Initialize $C[i] = 0$

Set $i = i + 1$

LOOP ENDS

Array manipulation

LOOP BEGINS: i starts at 0 and does not exceed N

Compute $C[i] = A[i] + B[i]$

Set $i = i + 1$

LOOP ENDS



Array manipulation

Pseudo-code; OpenMP #01

```
Define N, A[N], B[N], C[N], i
# POPULATE A[N], B[N]; INITIALIZE C[N] (AS IN SERIAL)

# NAIVE PARALLELIZATION (using #pragma omp parallel for)
LOOP BEGINS: i starts at 0 and does not exceed N
    Compute C[i] = A[i] + B[i]
    Set i = i + 1
LOOP ENDS
```

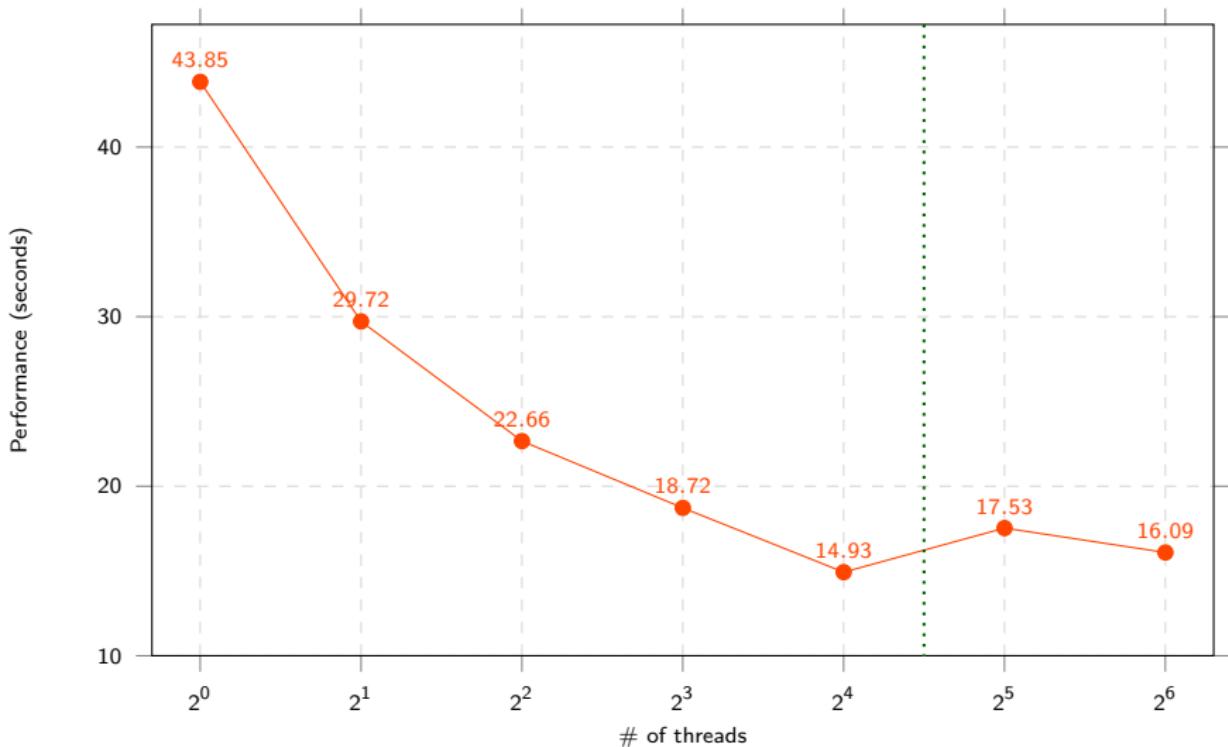
Comparison with MATLAB

Naive parallelization in OpenMP is equivalent to replacing the outermost for loop with **parfor** loop in MATLAB.



Array manipulation

Raw performance; OpenMP #01



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Array manipulation

Pseudo-code; OpenMP #02

```
Define  $N$ ,  $A[N]$ ,  $B[N]$ ,  $C[N]$ ,  $i$ 
# POPULATE  $A[N]$ ,  $B[N]$ ; INITIALIZE  $C[N]$  (AS IN SERIAL)
```

IMPROVED PARALLELIZATION (using `#pragma omp parallel`)

```
Define array_chunk, array_llimit, array_ulimit, NTHREADS
```

```
Compute array_chunk =  $N/NTHREADS$ 
```

```
Compute array_llimit = thread_id  $\times$  array_chunk
```

```
Compute array_ulimit = array_llimit + array_chunk - 1
```

LOOP BEGINS: i ranges from *array_llimit* through *array_ulimit*

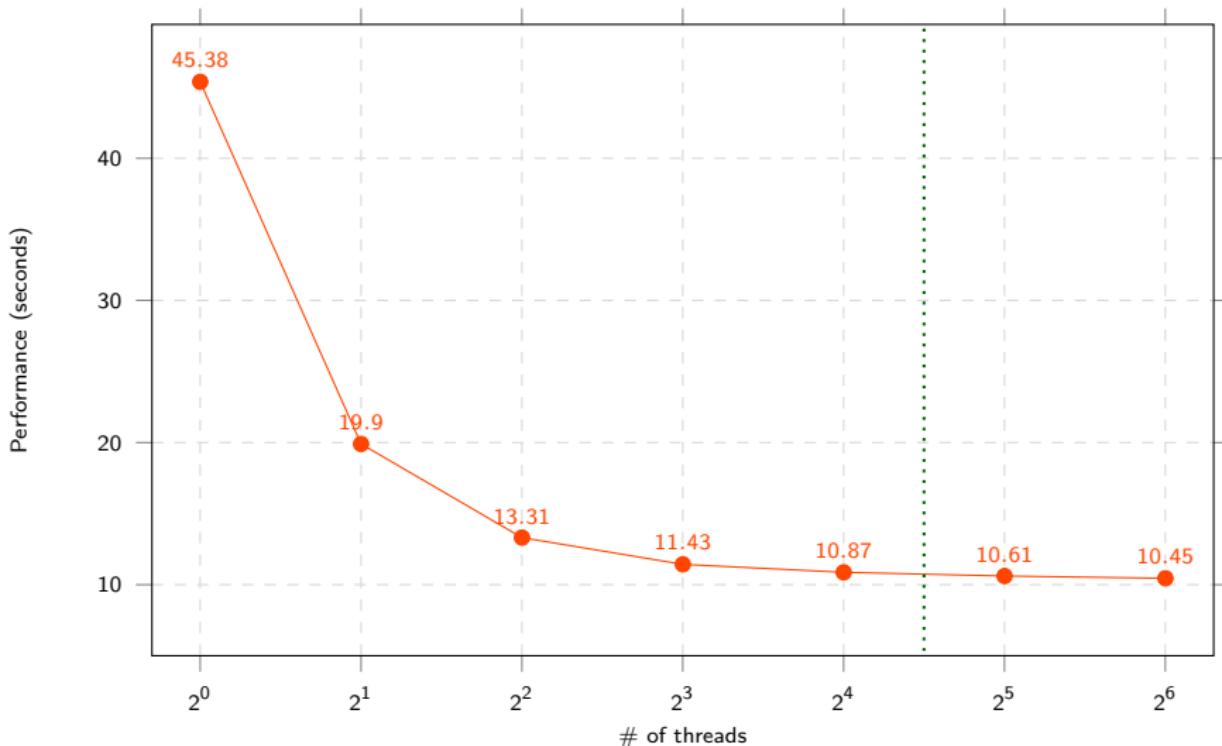
```
    Compute  $C[i] = A[i] + B[i]$ 
```

```
    Set  $i = i + 1$ 
```

LOOP ENDS

Array manipulation

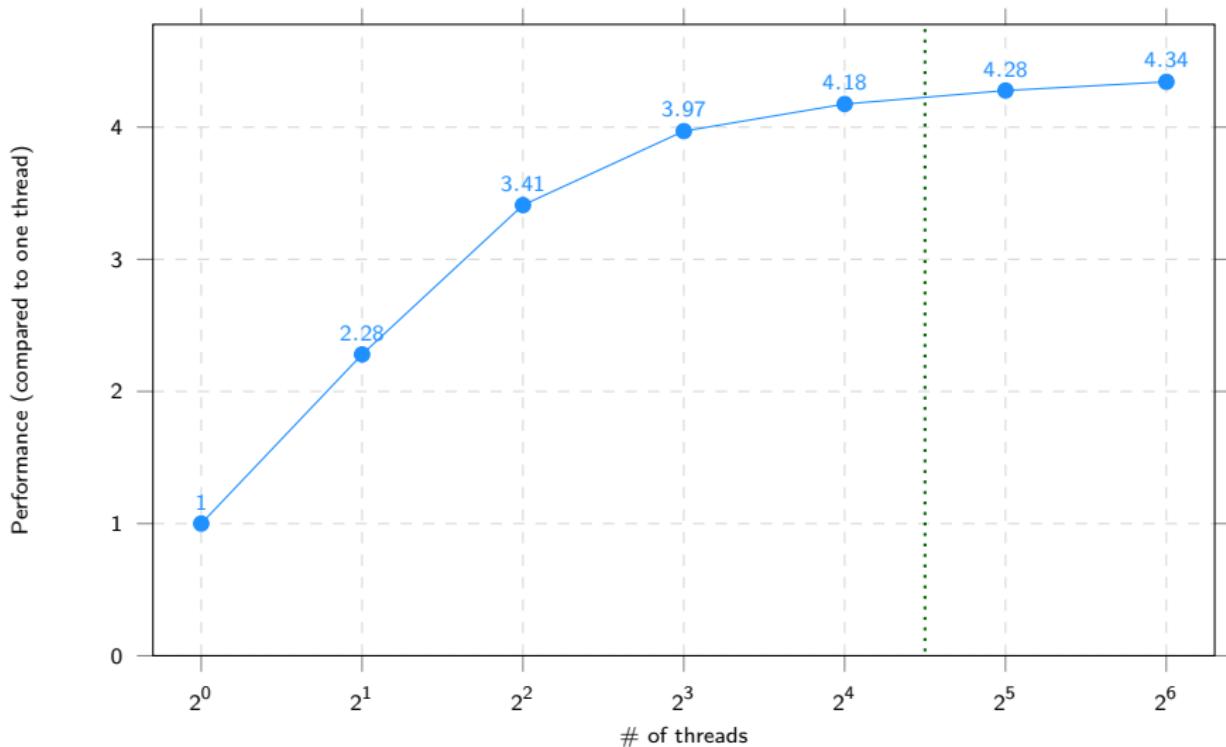
Raw performance; OpenMP #02



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

Array manipulation

Performance scaling; OpenMP #02



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

Array manipulation

My approach/mistakes

- * Should have written the serial version before attempting to write the parallel version
- * Spent a lot of time in formulating the expressions for *array_llimit* and *array_ulimit* based on *thread_id* and *array_chunk*. Working it out for smaller N (e.g., 2^4) with four threads on paper helped visualize division of labor
- * Used less than instead of less than or equal to in the for loop for threads. This resulted in the last element of $C[N]$ for each thread being zero. Explicit printing of $C[N]$ by master thread helped identify and resolve this problem

- * Keep simulation parameters (i.e., the array size) outside of the core program. Check if array size is integer-divisible by the number of usable threads
- * Check if the executable has been called with an appropriate number of arguments. Perform memory pre-allocation prior to populating the arrays
- * If writing to and/or reading from a file, check file/folder permissions

Parent shell script

Input validation, file/folder permissions, and checking if the array size is integer-divisible by the number of usable threads can be performed by the parent shell script.

Update usage instructions for `ArrayManipulation_omp_0*.c` if a parent shell script is performing some of these tasks.

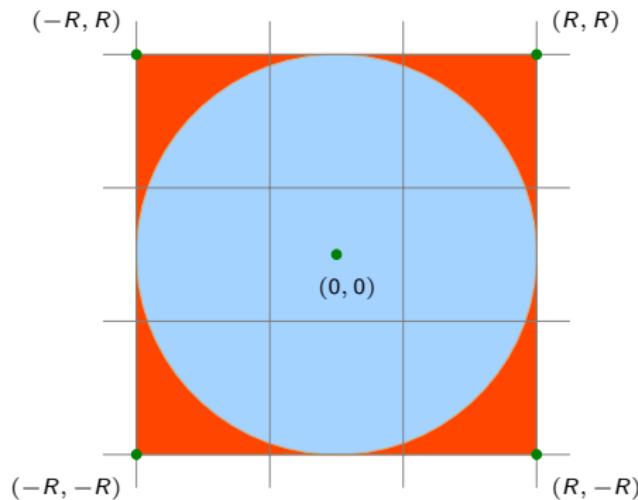


Monte Carlo methods

π evaluation; dart board algorithm

Monte Carlo methods

(Pseudo) Random numbers, seeding the random number generator, statistics, mapping functions, etc.



Compilation and execution instructions (serial; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="MonteCarloPi"  
VER="s"  
GCC="gcc -Wall -g -pg -lm"  
  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
$(pwd)/${BASENAME}_${VER}.x
```

`MonteCarloPi.s.c` is included in week #11 AdditionalMaterial.

Compilation and execution instructions (parallel; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="MonteCarloPi_omp"  
VER="01" # Change this to 02, 03, 04  
GCC="gcc -Wall -g -lm -fopenmp"  
  
# Compile, and run using 1, 2, 4, 8 threads  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
for x in $(seq 0 1 3)  
do  
    OMP_NUM_THREADS=$(echo "2^$x" | bc)  
    $(pwd)}/${BASENAME}_${VER}.x  
done
```

`MonteCarloPi_omp_0*.c` are included in week #11 AdditionalMaterial.

π evaluation

Dart board algorithm; pseudo-code; serial version

Define π , π_c , R , $n_s = N$, $n_c = 0$, x , y , ϵ , i

Seed the random number generator

LOOP BEGINS: i starts at 0 and does not exceed n_s

Generate x and y randomly, mapped in $(-R, R)$

IF BEGINS: $x^2 + y^2 \leq R^2$

 Set $n_c = n_c + 1$

IF ENDS

 Set $i = i + 1$

LOOP ENDS

Compute $\pi_c = 4(n_c/n_s)$ and $\epsilon = |\pi - \pi_c|$



π evaluation

Dart board algorithm; pseudo-code; OpenMP #01

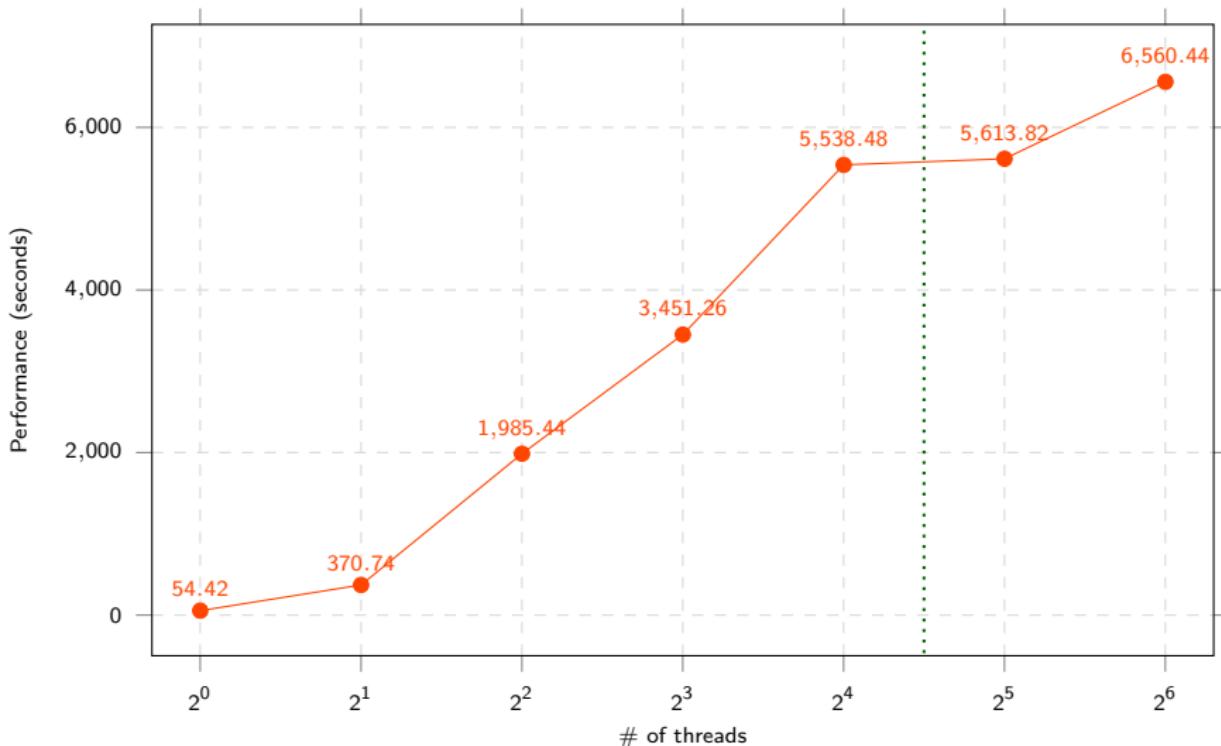
Define π , π_c , R , $n_s = N$, $n_c = 0$, x , y , ϵ , i
Seed the random number generator

```
# NAIVE PARALLELIZATION (using #pragma omp parallel for)
LOOP BEGINS: i starts at 0 and does not exceed ns
    Generate x and y randomly, mapped in (-R, R)
    IF BEGINS: x2 + y2 ≤ R2
        Set nc = nc + 1
    IF ENDS
        Set i = i + 1
    LOOP ENDS
```

Compute $\pi_c = 4(n_c/n_s)$ and $\epsilon = |\pi - \pi_c|$

π evaluation

Dart board algorithm; raw performance; OpenMP #01



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

π evaluation

Dart board algorithm; pseudo-code; OpenMP #02

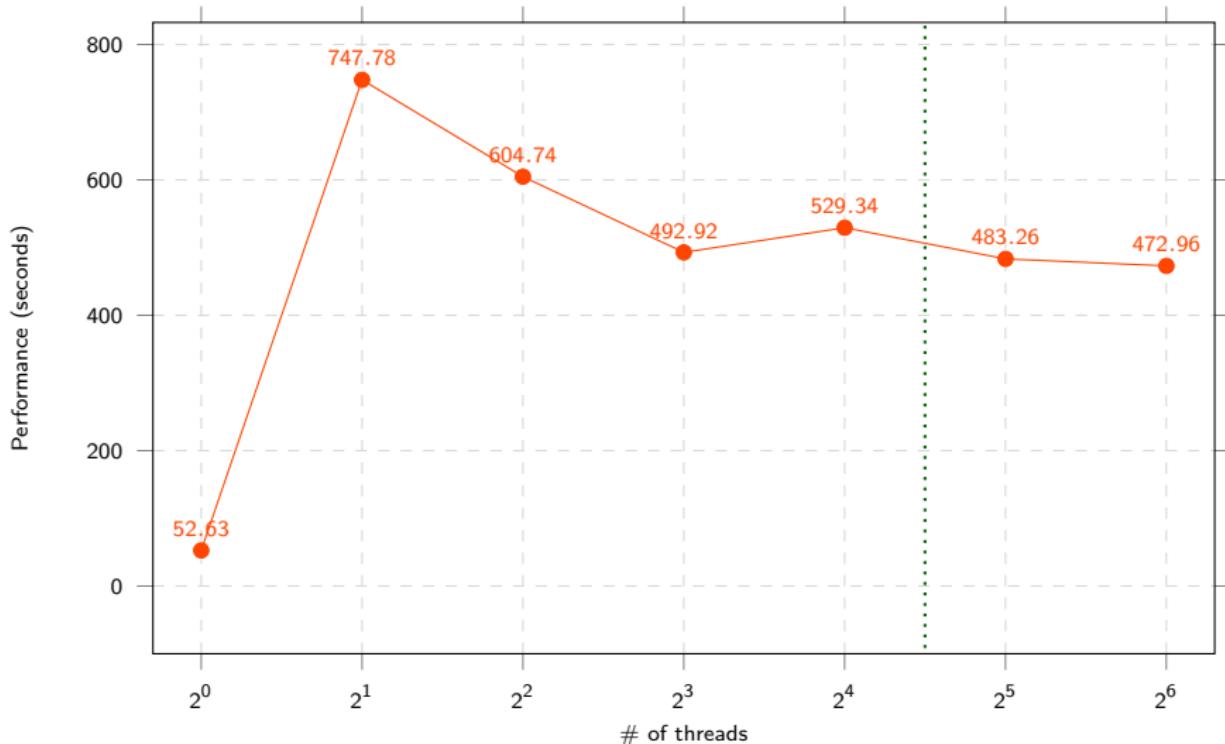
Define π , π_c , R , $n_s = N$, $n_c = 0$, x , y , ϵ , i
Seed the random number generator

```
# IMPROVED PARALLELIZATION (using #pragma omp parallel)
Define dart_chunk, dart_llimit, dart_ulimit, NTHREADS
Compute dart_chunk = N/NTHREADS
Compute dart_llimit = thread_id × dart_chunk
Compute dart_ulimit = dart_llimit + dart_chunk - 1
LOOP BEGINS: i ranges from dart_llimit through dart_ulimit
    # THE USUAL STUFF
LOOP ENDS
```

Compute $\pi_c = 4(n_c/n_s)$ and $\epsilon = |\pi - \pi_c|$

π evaluation

Dart board algorithm; raw performance; OpenMP #02



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

π evaluation

Dart board algorithm; pseudo-code; OpenMP #03

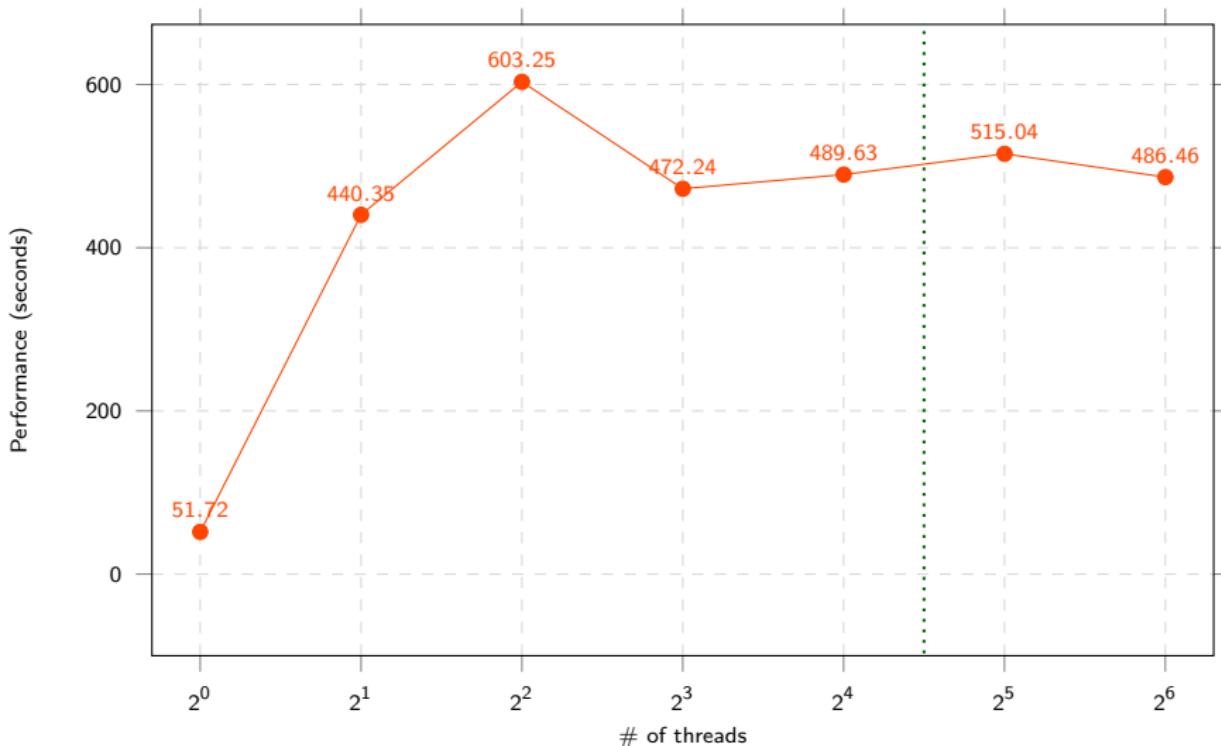
Define π , π_c , R , $n_s = N$, $n_c = 0$, x , y , x_r , y_r , ϵ , i
Seed the random number generator

```
# IMPROVED PARALLELIZATION
# (using #pragma omp parallel for and reduction)
LOOP BEGINS: i starts at 0 but does not exceed n_s
    # THE USUAL STUFF
LOOP ENDS
```

Compute $\pi_c = 4(n_c/n_s)$ and $\epsilon = |\pi - \pi_c|$

π evaluation

Dart board algorithm; raw performance; OpenMP #03



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

VARIABLE DECLARATION AND SEEDING THE RNG

```
# IMPROVED PARALLELIZATION (using #pragma omp sections)
```

```
Define dart_chunk, dart_llimit, dart_ulimit, NTHREADS, C[N]
```

```
Initialize C[N]; compute dart_chunk, dart_llimit, dart_ulimit
```

```
LOOP BEGINS: i ranges from dart_llimit through dart_ulimit
```

```
    # THE USUAL STUFF EXCEPT THE FOLLOWING CHANGE
```

```
    IF BEGINS:  $x^2 + y^2 \leq R^2$ 
```

```
        Set C[i] = 1
```

```
    IF ENDS
```

```
LOOP ENDS
```

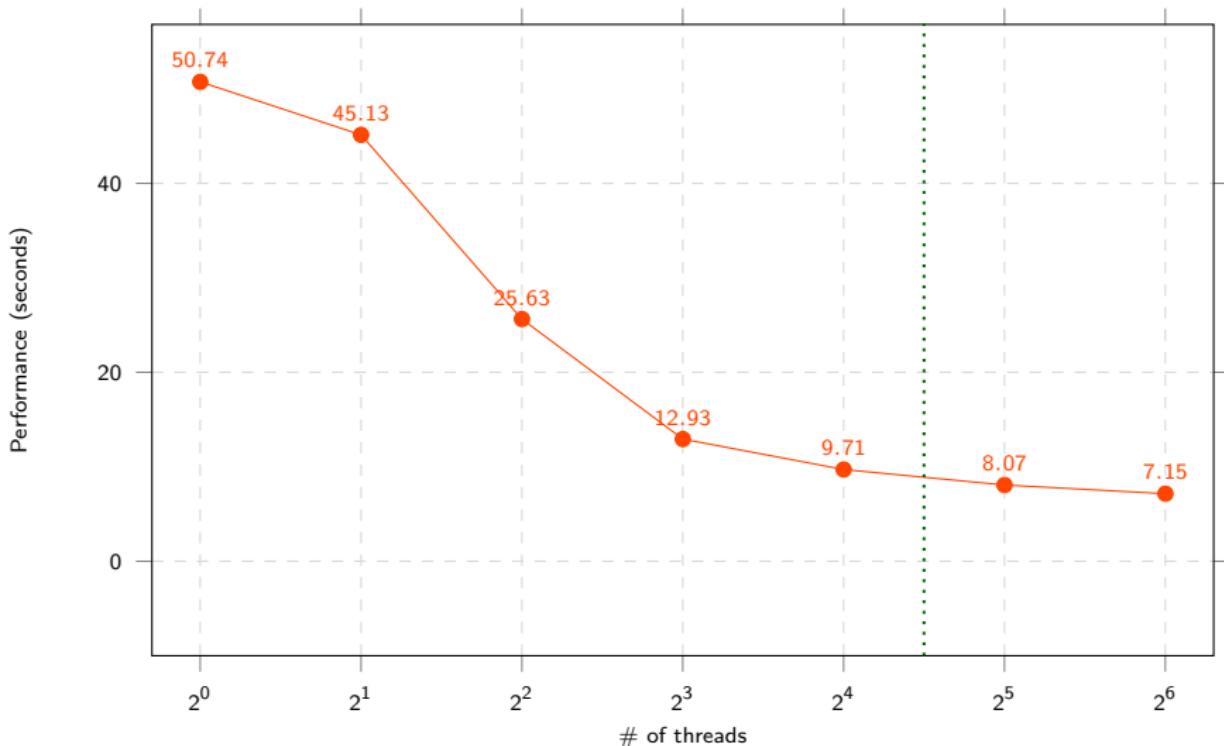
n_c = sum of all elements in *C[N]*

Compute $\pi_c = 4(n_c/n_s)$ and $\epsilon = |\pi - \pi_c|$



π evaluation

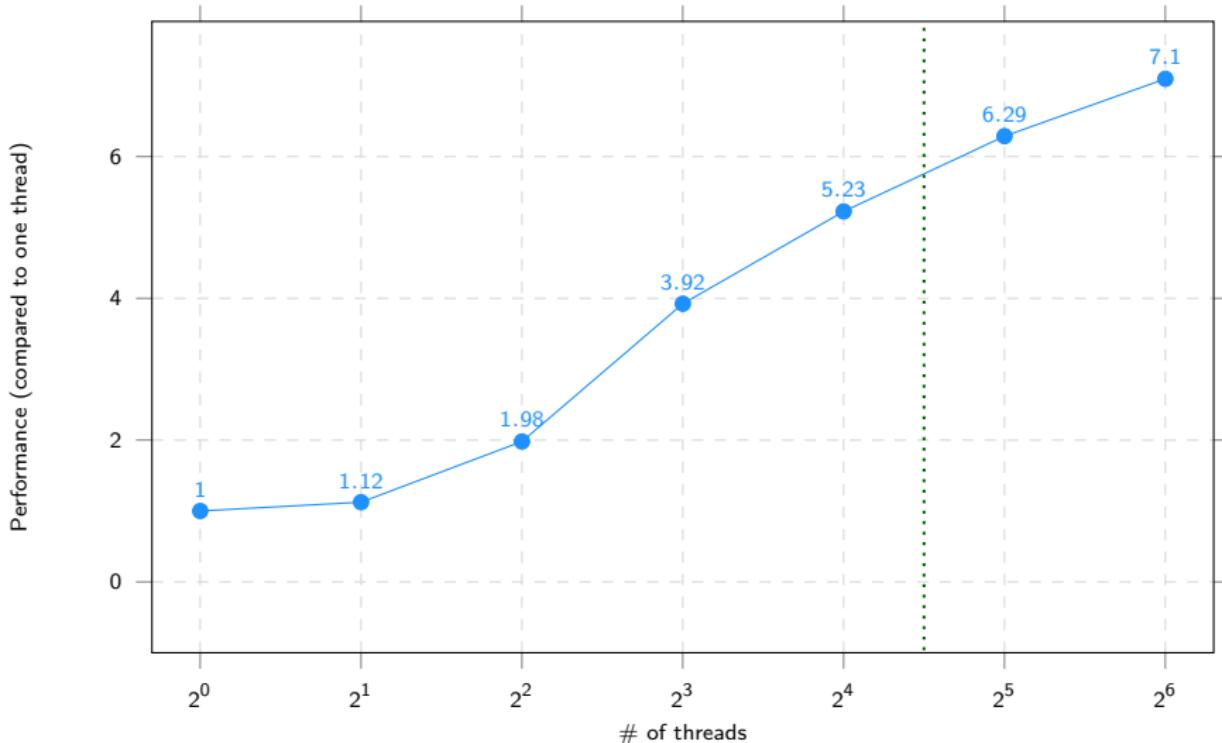
Dart board algorithm; raw performance; OpenMP #04



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



π evaluation Dart board algorithm; performance scaling; OpenMP #04



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

- * Should have written the serial version before attempting to write the parallel version(s)
- * Versions #01 and #02 took more time with an increase in number of threads. Some variables were declared as shared and private. Some necessary variables weren't declared as shared or private. Working it out for smaller N (e.g., 2^5) with four threads helped visualize division of labor but did not resolve the issue
- * Version #03 employed the reduction operation (collective communication) but did not resolve the issue
- * Version #04 implemented `#pragma omp sections` along with an array and resolved the performance issues noted above

- * Keep simulation parameters (i.e., # of darts thrown) outside of the core program. Check if # of darts thrown is integer-divisible by the number of usable threads
- * Check if the executable has been called with an appropriate number of arguments. Perform memory pre-allocation prior to populating the arrays
- * If writing to and/or reading from a file, check file/folder permissions

Parent shell script

Input validation, file/folder permissions, and checking if the # of darts thrown is integer-divisible by the number of usable threads can be performed by the parent shell script.

Update usage instructions for `MonteCarloPi_omp_0*.c` if a parent shell script is performing some of these tasks.

A note of caution

$$x^2 + y^2 \leq R^2$$

$$\sqrt{x^2 + y^2} \leq R$$

Different compilers implement the above conditions differently. There could also be a difference in implementation of $(x * x)$ and $pow(x, 2)$ even though both give the same numerical answer.

Check compiler documentation and run benchmarks to identify (and use) the one that gives optimal result for the available compiler.

Numerical integration

Trapezoidal rule in polynomial approximations.

Composite formula for trapezoidal rule obtained by splitting the interval $[x_0, x_n]$ into n segments of equal width h

$$\int_{x_0}^{x_n} f(x) dx = h \left[\frac{f(x_0) + f(x_n)}{2} + \sum_{i=1}^{n-1} f(x_0 + i * h) \right]$$

One of the many integrals yielding π

$$\pi_c = 4 \int_0^1 \sqrt{1 - x^2}$$

Compilation and execution instructions (serial; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="IntegrationPi"  
VER="s"  
GCC="gcc -Wall -g -pg -lm"  
  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
$(pwd)/${BASENAME}_${VER}.x
```

`IntegrationPi.s.c` is included in week #11 AdditionalMaterial.

Compilation and execution instructions (parallel; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="IntegrationPi_omp_"  
VER="01" # Change this to 02  
GCC="gcc -Wall -g -lm -fopenmp"  
  
# Compile, and run using 1, 2, 4, 8 threads  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
for x in $(seq 0 1 3)  
do  
    OMP_NUM_THREADS=$(echo "2^$x" | bc)  
    $(pwd)}/${BASENAME}_${VER}.x  
done
```

`IntegrationPi_omp_0*.c` are included in week #11 AdditionalMaterial.

π evaluation

Trapezoidal rule; pseudo code; serial version

Define π , π_c , $a = 0$, $b = 1$, h , N , ϵ , i

Compute $h = (b - a)/N$

LOOP BEGINS: i starts at 1 and does not exceed $N - 1$

$\pi_c = \pi_c + f_x(a + i \times h)$

Set $i = i + 1$

LOOP ENDS

Compute $\pi_c = \pi_c + 0.50 \times [f_x(a) + f_x(b)]$

Compute $\pi_c = 4.00 \times \pi_c$

Compute $\epsilon = |\pi - \pi_c|$

function $f_x(x)$ {

 Compute and return $y = \sqrt{1 - x^2}$

}



π evaluation

Trapezoidal rule; pseudo code; OpenMP #01

Define π , π_c , $a = 0$, $b = 1$, h , N , ϵ , i
Compute $h = (b - a)/N$

NAIVE PARALLELIZATION (using `#pragma omp parallel for`)

LOOP BEGINS: i starts at 1 and does not exceed $N - 1$

$\pi_c = \pi_c + f_x(a + i \times h)$

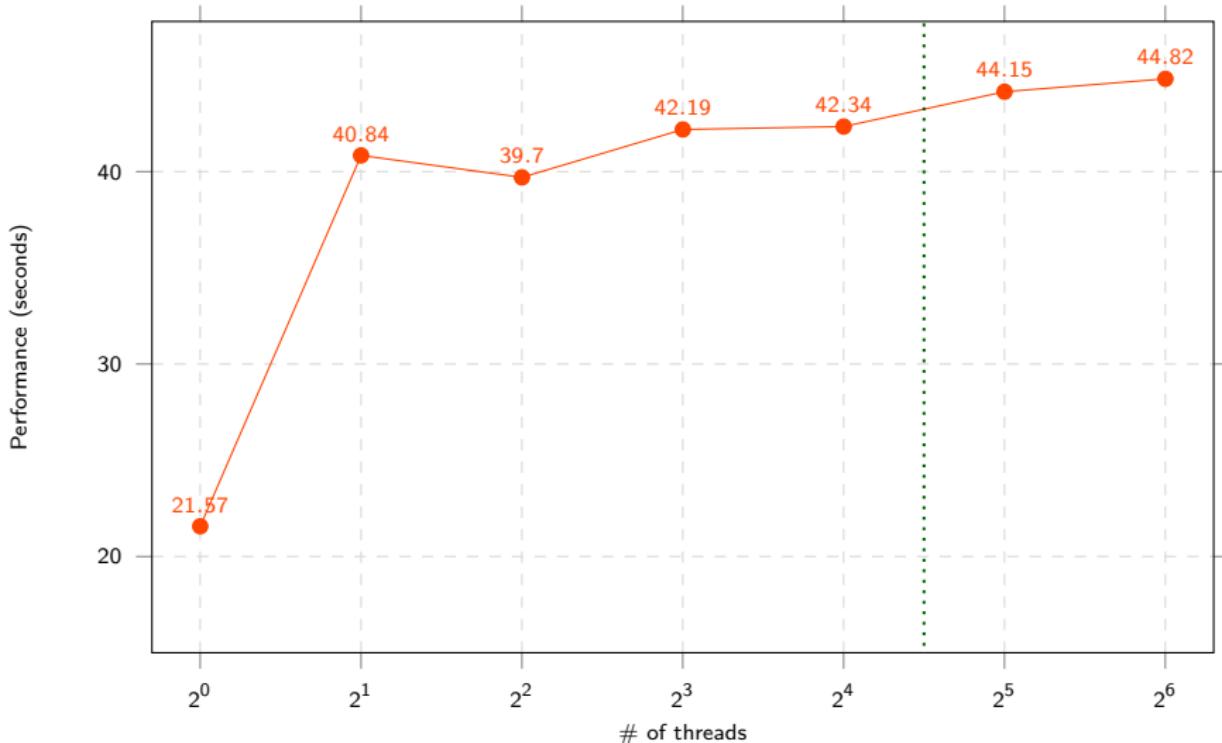
Set $i = i + 1$

LOOP ENDS

COMPUTE π_c AND ϵ

FUNCTION $f_x(x)$

π evaluation Trapezoidal rule; raw performance; OpenMP #01



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



In addition to wasting precious CPU time,

π_c values are incorrect for $NTHREAD > 1$

NTHREAD	π_c
2	1.889572898852592
4	0.942186265377353
8	0.627084118251614
16	0.368390622330348
32	0.198695634590972
64	0.100347566640417

Define π , π_c , $a = 0$, $b = 1$, h , N , ϵ , i
Compute $h = (b - a)/N$

IMPROVED PARALLELIZATION

(using `#pragma omp parallel for` and reduction)

LOOP BEGINS: i starts at 1 but does not exceed $N - 1$

$$\pi_c = \pi_c + f_x(a + i \times h)$$

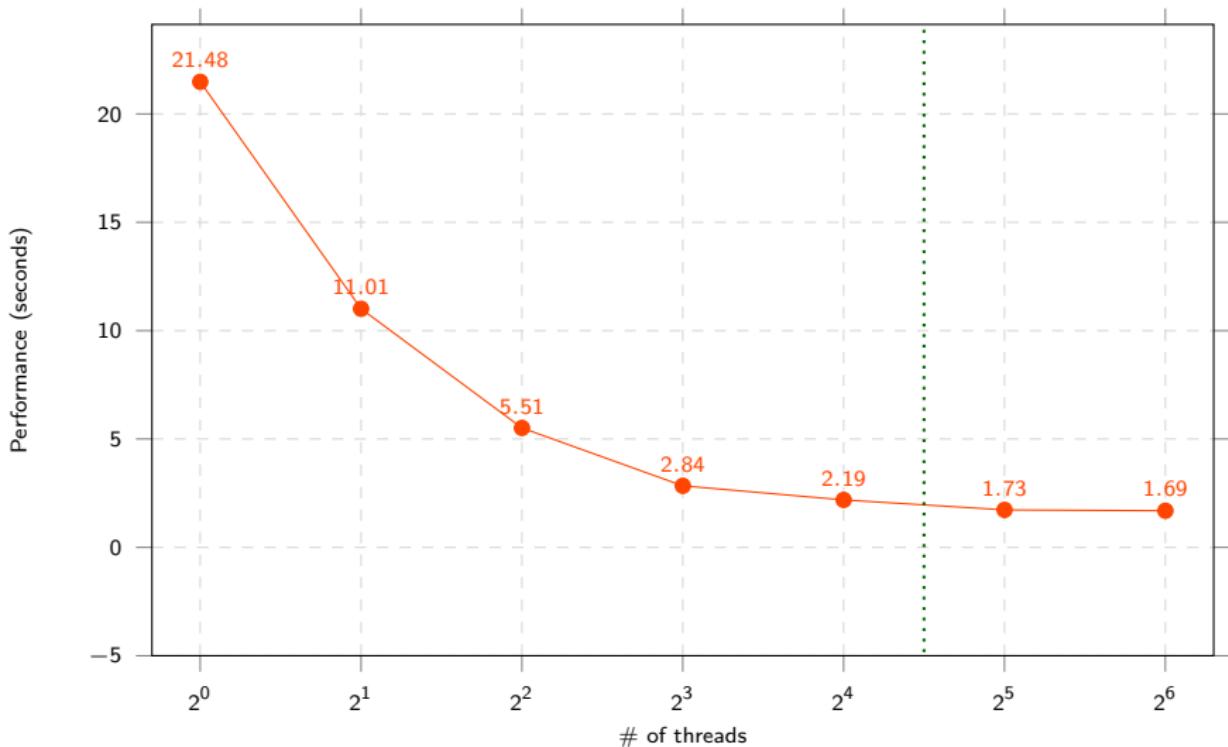
Set $i = i + 1$

LOOP ENDS

COMPUTE π_c AND ϵ

FUNCTION $f_x(x)$

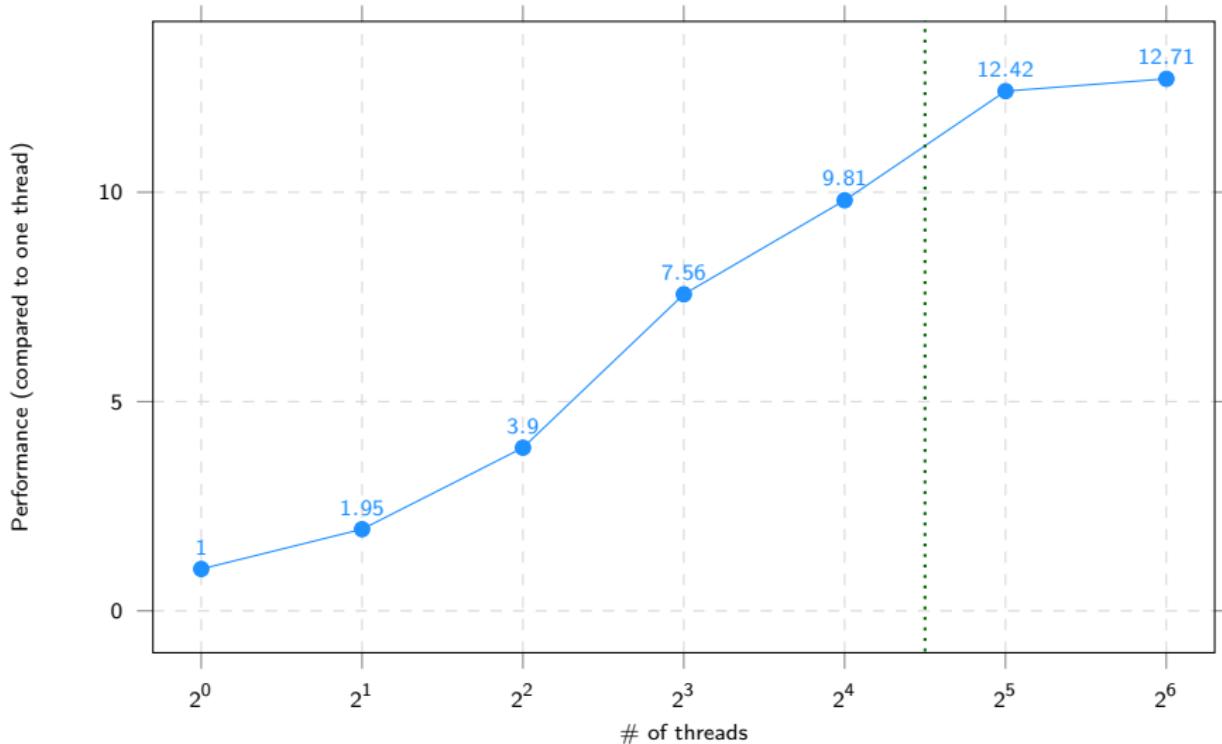
π evaluation Trapezoidal rule; raw performance; OpenMP #02



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



π evaluation Trapezoidal rule; performance scaling; OpenMP #02



$N = 2^{30}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

- * Serial version was written before attempting to write the parallel version(s)
- * Version #01 not only consumed more time for $NTHREADS > 1$ compared to $NTHREADS = 1$ but the value of π_c was incorrect as well. Had $NTHREADS > 1$ taken less time compared to $NTHREADS = 1$, there was some chance I would have assumed the correctness of results
- * Version #02 employed the reduction operation (collective communication), and resolved accuracy and performance issues

- * Keep simulation parameters (i.e., # of intervals, limits of integration, integrand and integration scheme) outside of the core program.
Check if # of intervals is appropriately integer-divisible by the number of usable threads
- * Check if the executable has been called with an appropriate number of arguments. Check if h or $f_x(x)$ is too small/big at some point
- * If writing to and/or reading from a file, check file/folder permissions

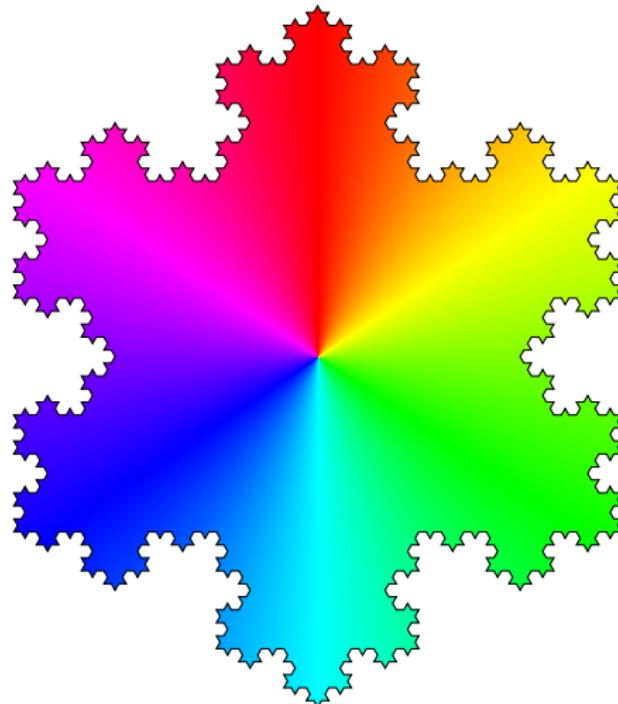
Parent shell script

Input validation, file/folder permissions, and checking if the # of intervals is appropriately integer-divisible by the number of usable threads can be performed by the parent shell script.

Update usage instructions for `IntegrationPi_omp_0*.c` if a parent shell script is performing some of these tasks.

Before we meet again

- * Review the syllabus, course material, grade through week #13, notations, active participation, free time exercises, tips, opportunities, mathematical results, videos, and training camps
- * Practice compiling and executing included OpenMP programs
- * Make progress in assignment #12
- * Make progress in the term project
- * Make progress towards Research Marketing III



End of Tuesday lecture.

Parallel Computing Examples

OpenMP (C) programs



<http://dilbert.com/strip/2000-07-11/>

OpenMP sample programs

In-class activity

To be performed in `colossus.it`

```
cd ${UN5390}  
git pull  
cd CourseWork/Week_11/AdditionalMaterial  
rsync -avhP ./Parallel/ ../${USER}_11/Parallel/  
  
cd ${UN5390}/CourseWork/Week_11/  
git add ${USER}_11  
git commit -m "PM: OpenMP samples #13"  
git push origin master
```

Matrix multiplication

Journal of Failed Experiments

Input validation, memory pre-allocation, row/column major language, zeros cost space and time, zero is not really zero, order and nature of matrices involved, etc.

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \dots & \dots & \dots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \begin{pmatrix} b_{11} & \dots & b_{1q} \\ b_{21} & \dots & b_{2q} \\ \dots & \dots & \dots \\ b_{p1} & \dots & b_{pq} \end{pmatrix} = \begin{pmatrix} c_{11} & \dots & c_{1q} \\ c_{21} & \dots & c_{2q} \\ \dots & \dots & \dots \\ c_{m1} & \dots & c_{mq} \end{pmatrix}$$

$$a_{ij} = i \times (m - i - 1) \times j \times (n - j - 1)$$

$$b_{ij} = i + j + 2$$

Matrix multiplication

Compilation and execution instructions (serial; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="MatrixMultiplication"  
VER="s"  
GCC="gcc -Wall -g -pg -lm"  
  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
$(pwd)/${BASENAME}_${VER}.x
```

`MatrixMultiplication.s.c` is included in week #11 AdditionalMaterial.

Matrix multiplication

Compilation and execution instructions (parallel; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel
BASENAME="MatrixMultiplication_omp_"
VER="01" # Change this to 02 for version #02
GCC="gcc -Wall -g -lm -fopenmp"

# Compile, and run using 1, 2, 4, 8 threads
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x
for x in $(seq 0 1 3)
do
    OMP_NUM_THREADS=$(echo "2^$x" | bc)
    ${pwd}/${BASENAME}_${VER}.x
done
```

`MatrixMultiplication_omp_0*.c` are included in week #11 AdditionalMaterial.



Matrix multiplication

Pseudo-code; serial version

Define $i, j, k, NRA, NCA, NRB, NCB$

Define $A[NRA][NCA], B[NRB][NCB], C[NRA][NCB]$

Populate $A[NRA][NCA], B[NRB][NCB], C[NRA][NCB]$

LOOP BEGINS: i starts at 0 and does not exceed NRA

 LOOP BEGINS: j starts at 0 and does not exceed NCB

 LOOP BEGINS: k starts at 0 and does not exceed NCA

 Compute $C[i][j] = C[i][j] + (A[i][k] \times B[k][j])$

 Set $k = k + 1$

 LOOP ENDS

 Set $j = j + 1$

 LOOP ENDS

 Set $i = i + 1$

LOOP ENDS

Matrix multiplication

Pseudo-code; OpenMP #01

Define $i, j, k, NRA, NCA, NRB, NCB$

Define/Populate $A[NRA][NCA], B[NRB][NCB], C[NRA][NCB]$

NAIVE PARALLELIZATION (using `#pragma omp parallel for`)

LOOP BEGINS: i starts at 0 and does not exceed NRA

 LOOP BEGINS: j starts at 0 and does not exceed NCB

 LOOP BEGINS: k starts at 0 and does not exceed NCA

 Compute $C[i][j] = C[i][j] + (A[i][k] \times B[k][j])$

 Set $k = k + 1$

 LOOP ENDS

 Set $j = j + 1$

 LOOP ENDS

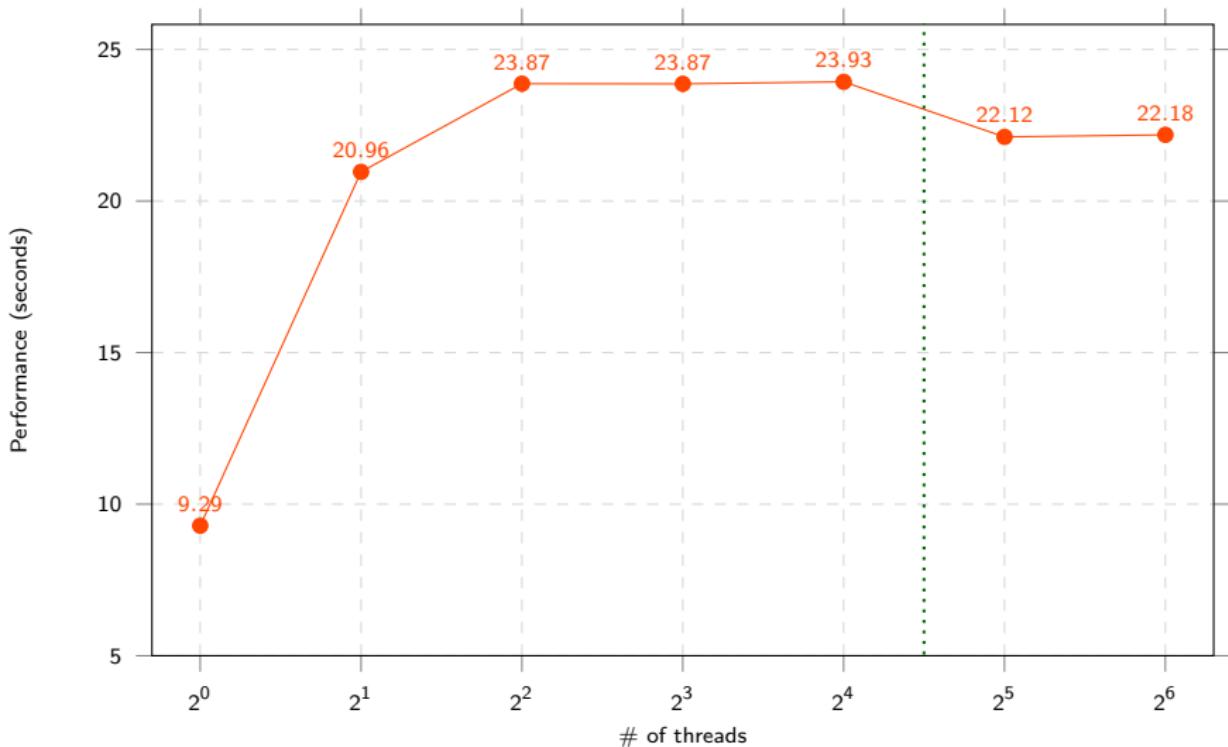
 Set $i = i + 1$

LOOP ENDS



Matrix multiplication

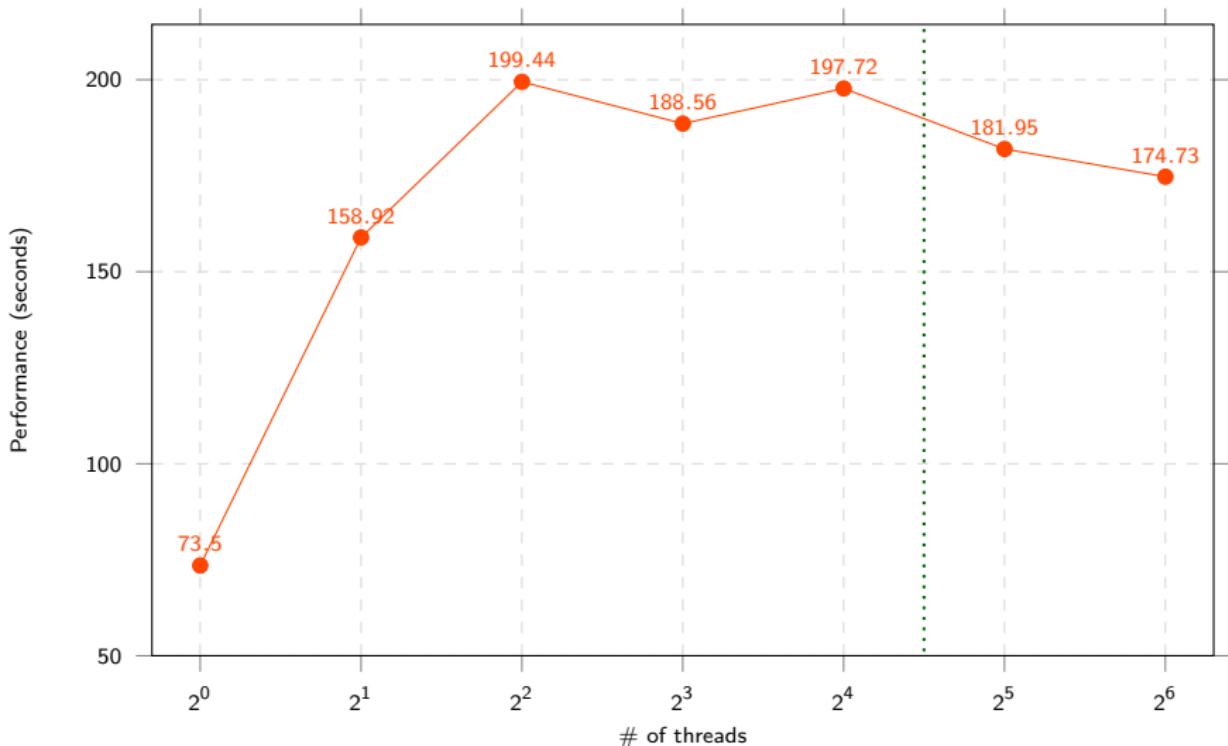
Raw performance; OpenMP #01



$N = 2^{10}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

Matrix multiplication

Raw performance; OpenMP #01

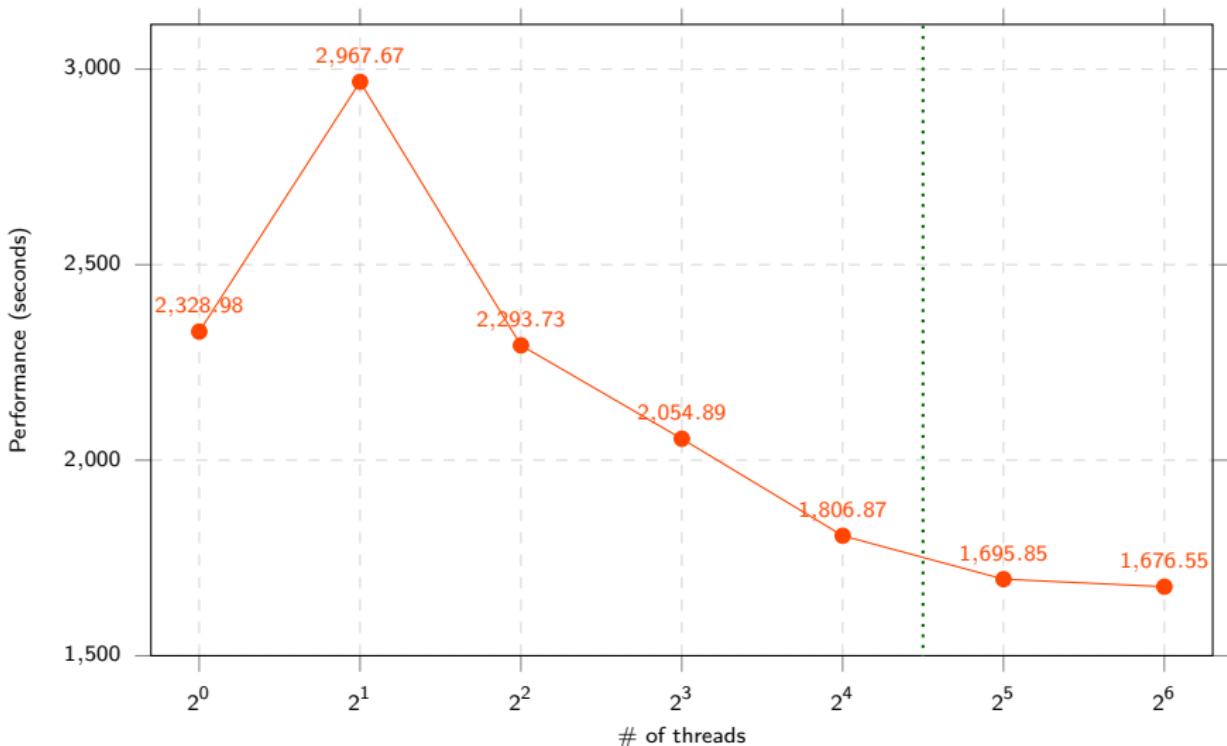


$N = 2^{11}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

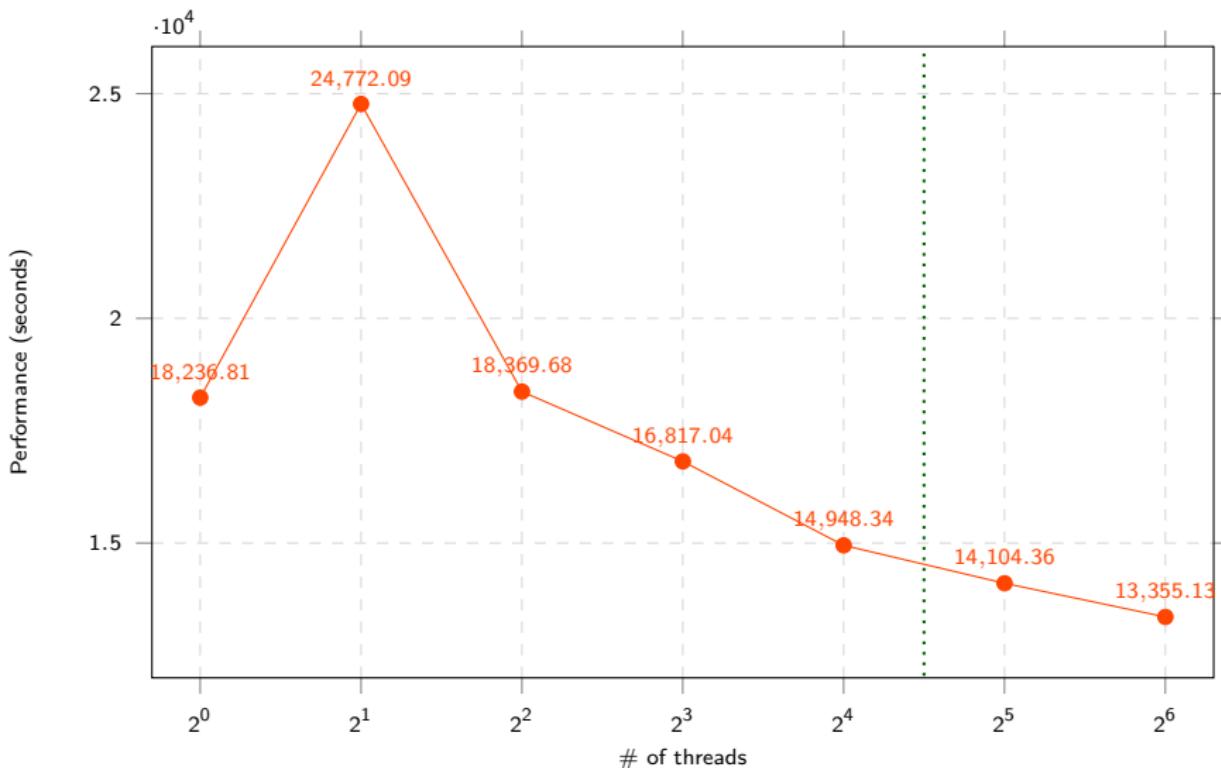
Raw performance; OpenMP #01



$N = 2^{12}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

Matrix multiplication

Raw performance; OpenMP #01



$N = 2^{13}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Pseudo-code; OpenMP #02

Define $i, j, k, NRA, NCA, NRB, NCB$

Define/Populate $A[NRA][NCA]$, $B[NRB][NCB]$, $C[NRA][NCB]$

IMPROVED PARALLELIZATION

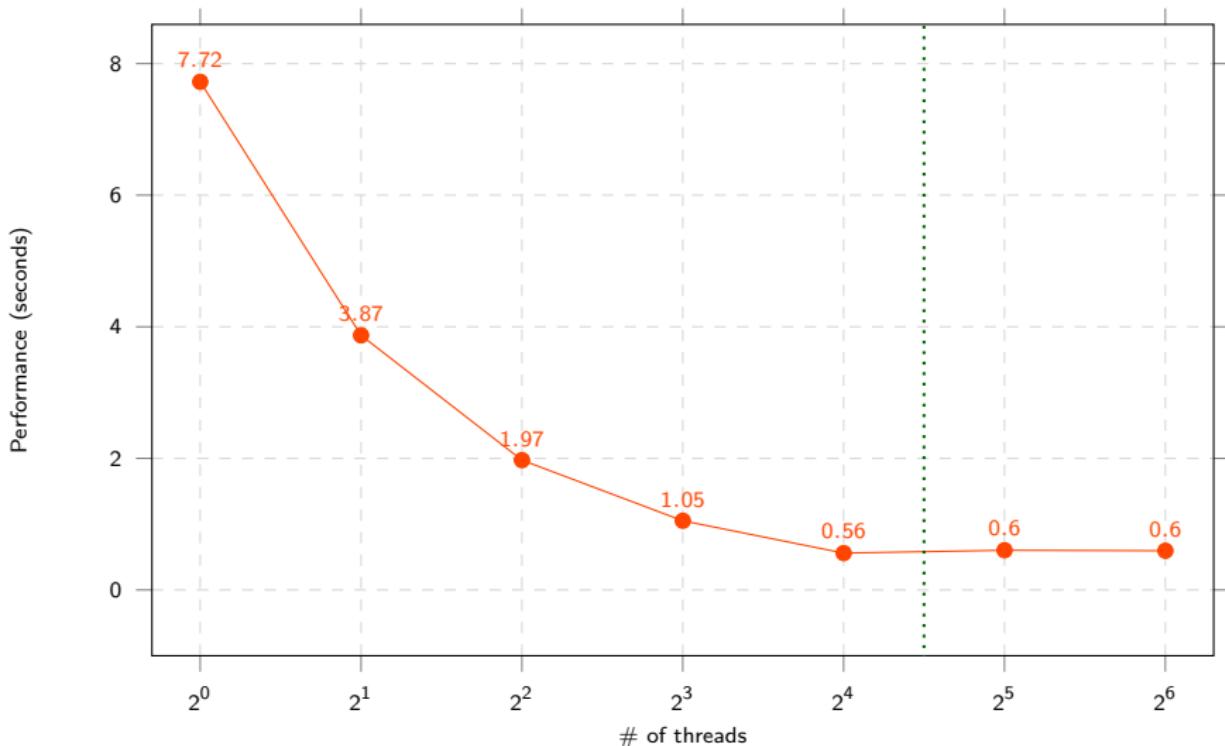
Using #pragma omp parallel for and explicit

identification of shared and private variables

THE NESTED LOOPS

Matrix multiplication

Raw performance; OpenMP #02

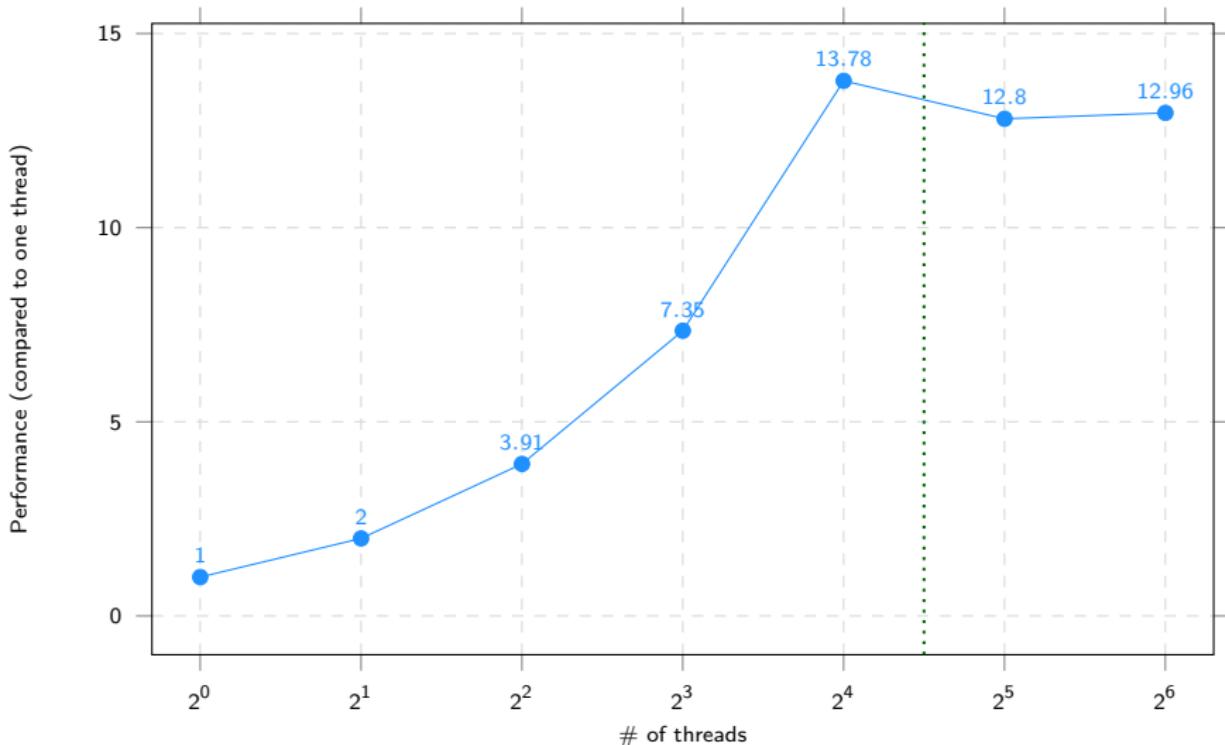


$N = 2^{10}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Performance scaling; OpenMP #02

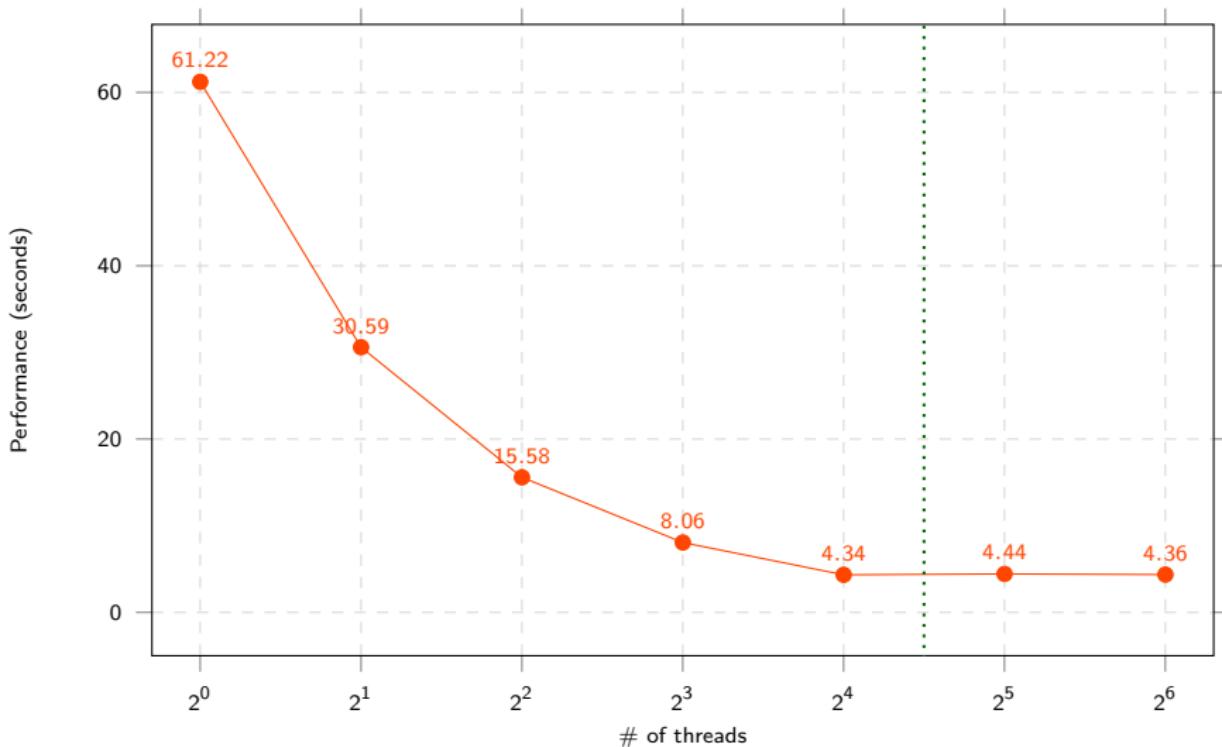


$N = 2^{10}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Raw performance; OpenMP #02

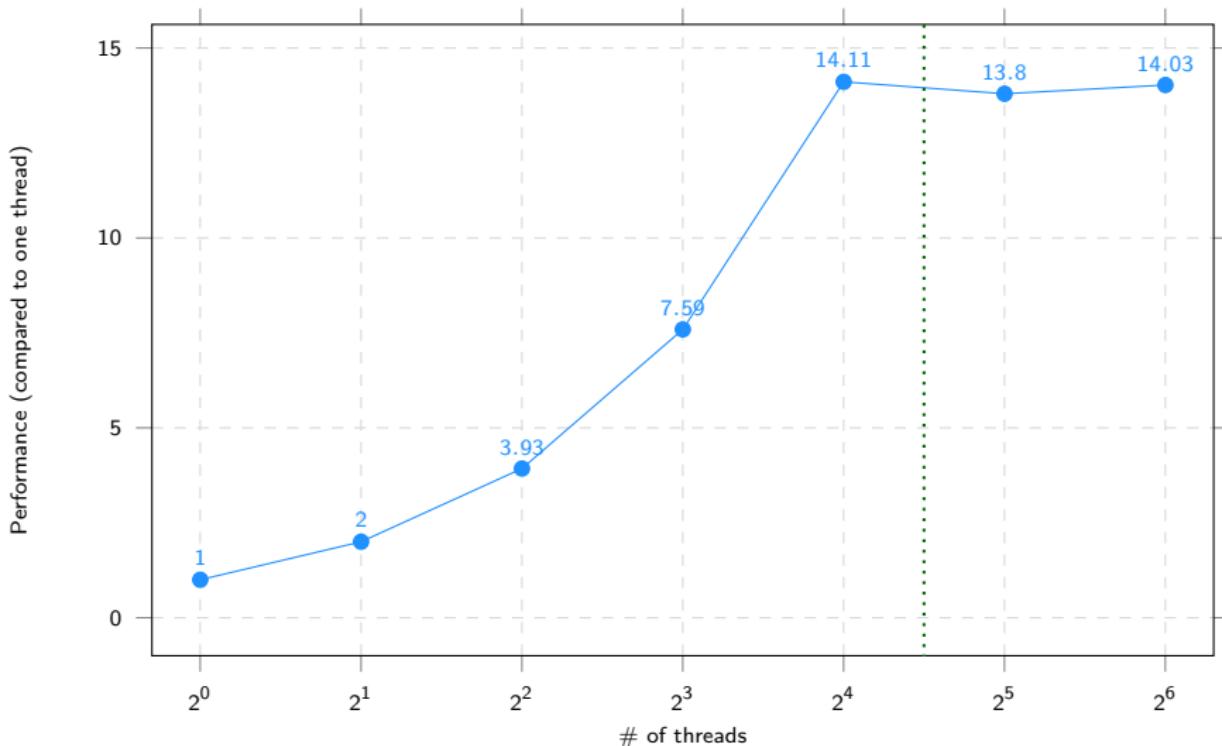


$N = 2^{11}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Performance scaling; OpenMP #02

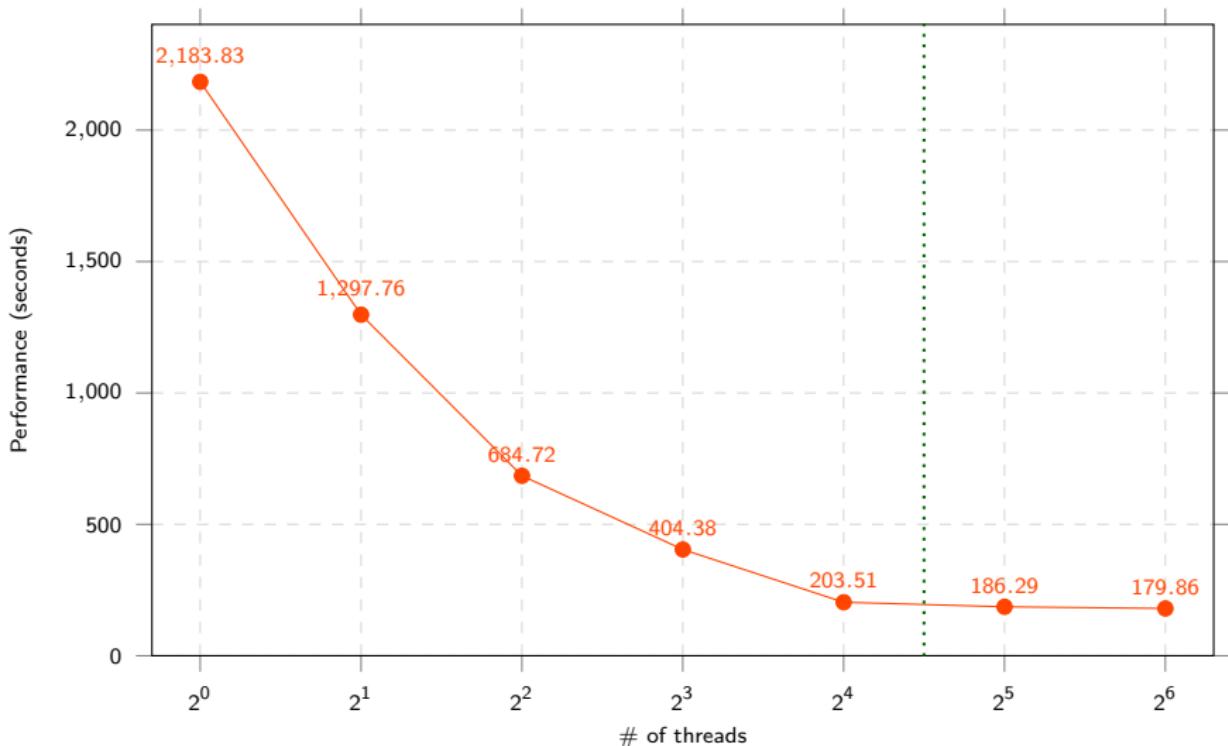


$N = 2^{11}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Raw performance; OpenMP #02

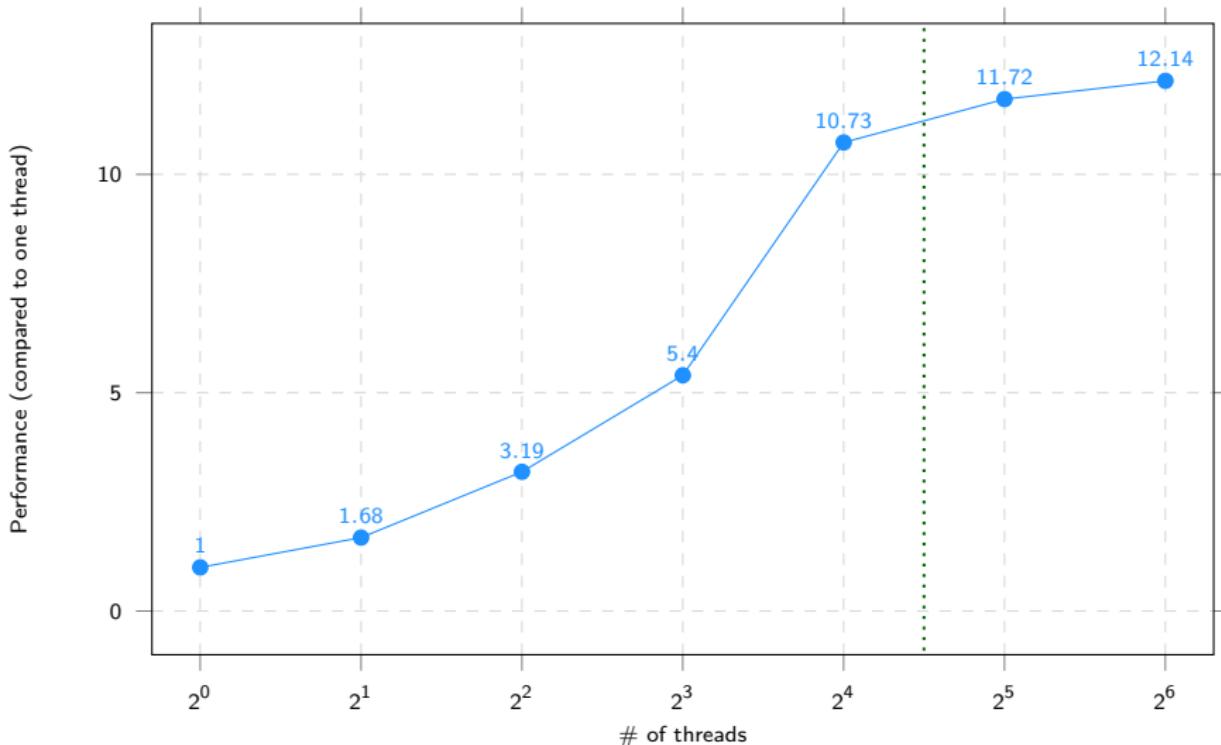


$N = 2^{12}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

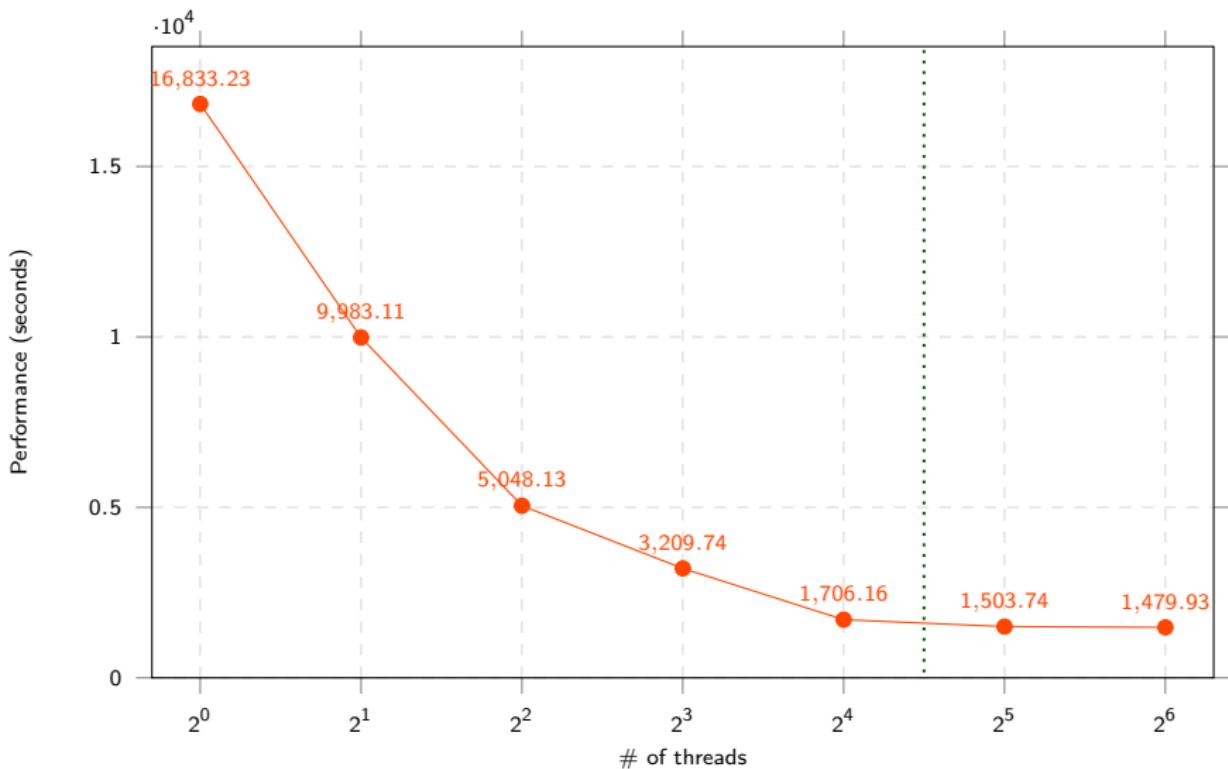
Performance scaling; OpenMP #02



$N = 2^{12}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM

Matrix multiplication

Raw performance; OpenMP #02

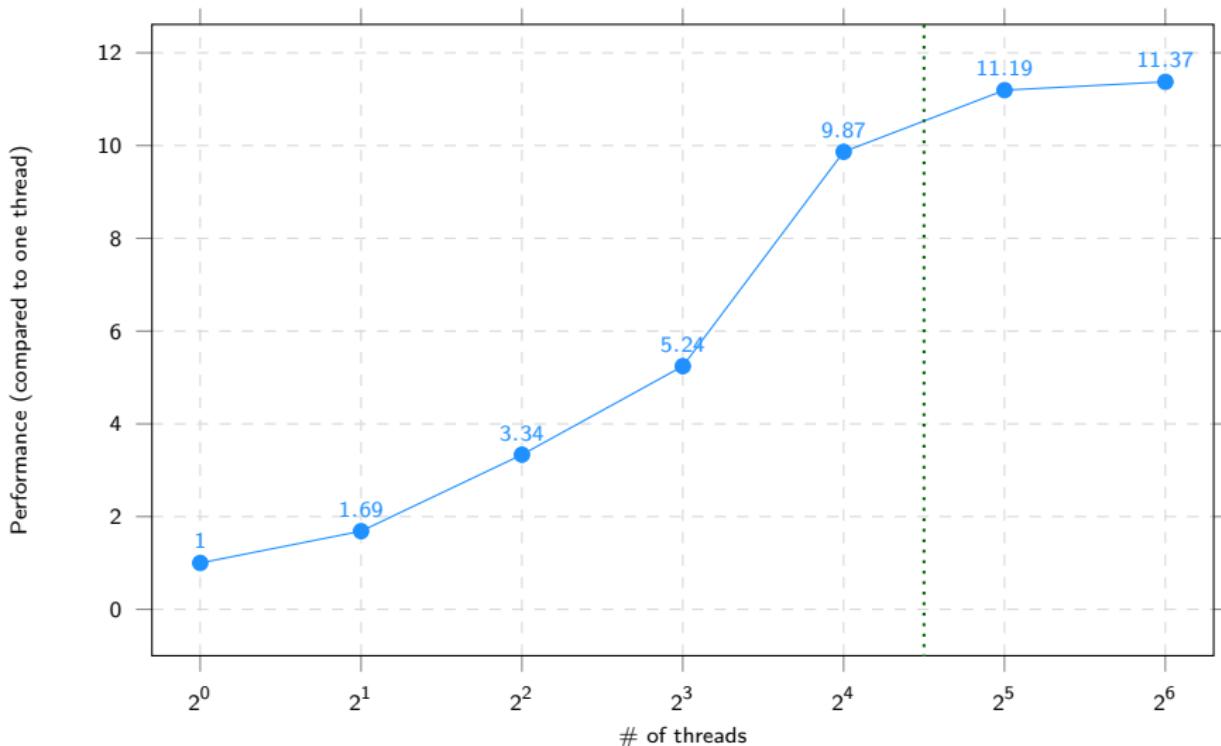


$N = 2^{13}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Performance scaling; OpenMP #02

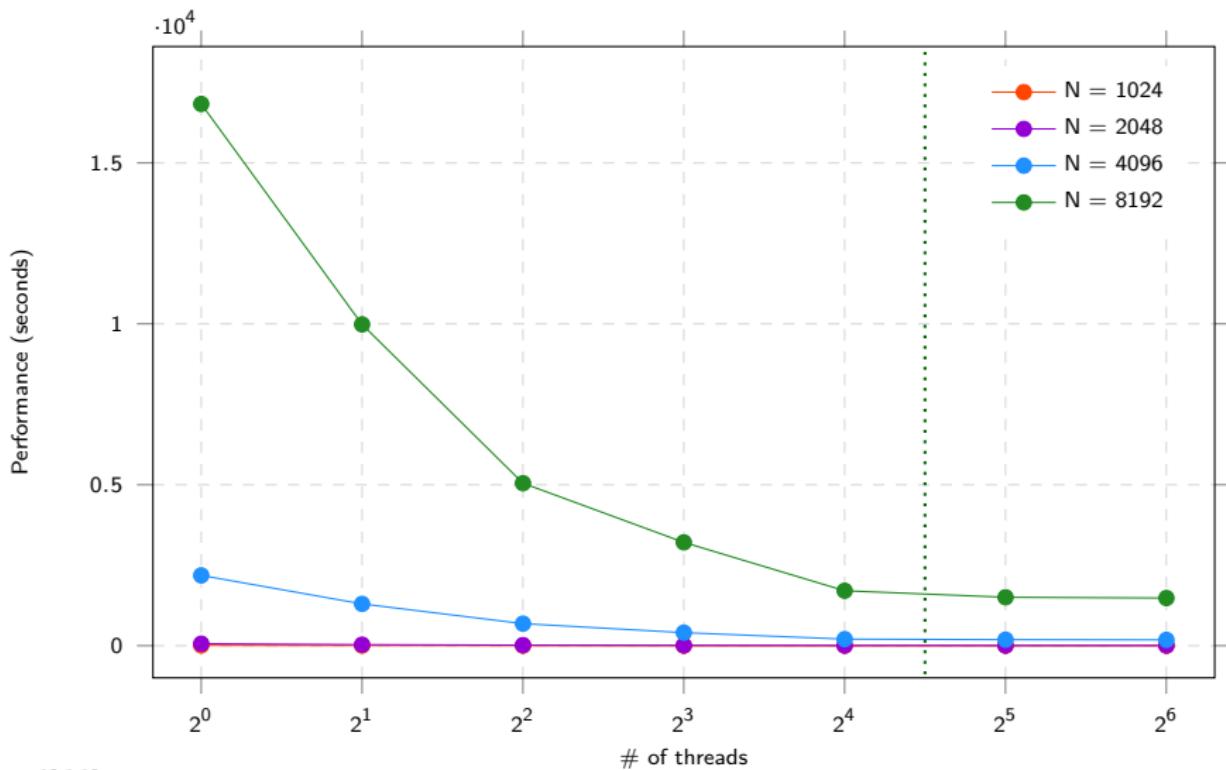


$N = 2^{13}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Raw performance; OpenMP #02

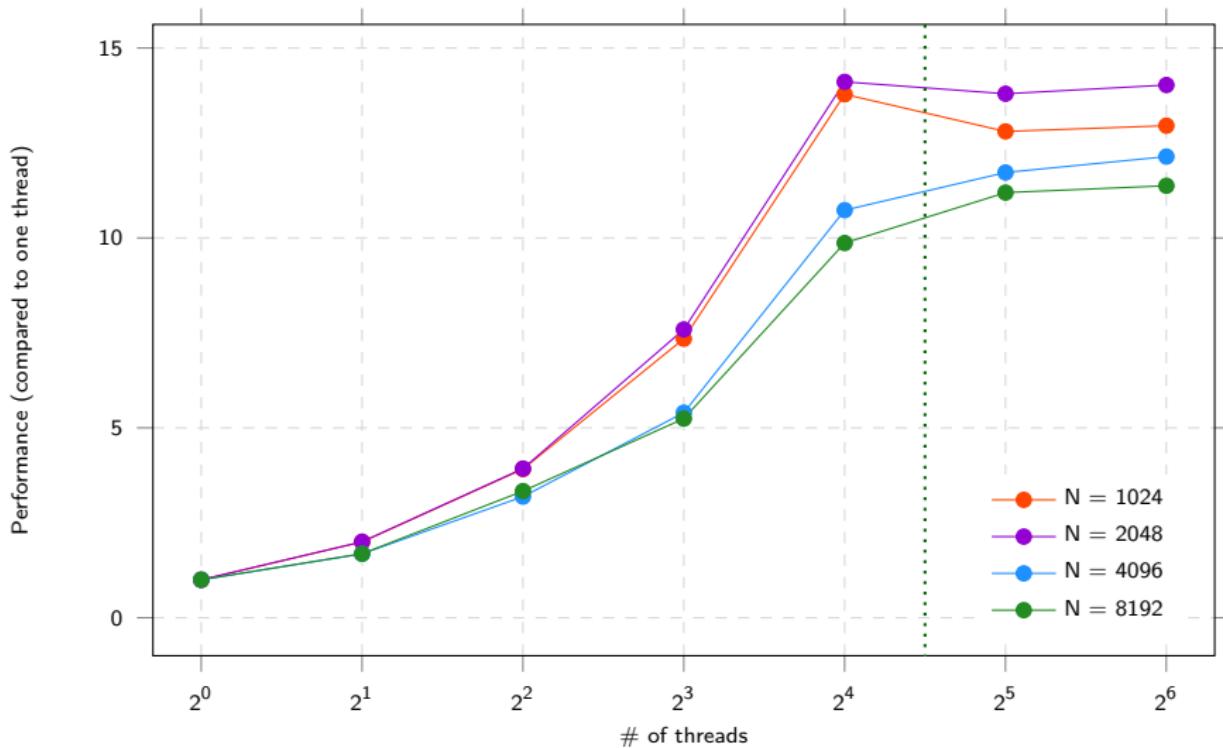


$N = 2^{10:1:13}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

Performance scaling; OpenMP #02



$N = 2^{10:1:13}$; Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Matrix multiplication

My approach/mistakes

- * Both raw and processed data are retained
- * The order of matrices is accepted at the command line
- * Serial version was written before attempting to write the parallel version(s)
- * Version #01 (did not explicitly identify shared and private variables) took approximately the same amount of time for $NTHREADS > 1$ compared to $NTHREADS = 1$ but the results were identical
- * Version #02 (did explicitly identify shared and private variables) was designed to handle (and check) order of the matrices, and resolved the performance issues



- * Check if the order of matrices is appropriately integer-divisible by the number of usable threads
- * Perform memory pre-allocation prior to populating the arrays
- * If writing to and/or reading from a file, check file/folder permissions

Parent shell script

Input validation, file/folder permissions, and checking if the order of matrices is appropriately integer-divisible by the number of usable threads can be performed by the parent shell script.

Update usage instructions for `MatrixMultiplication_omp_0*.c` if a parent shell script is performing some of these tasks.



Additional references

- * Strassen Algorithm
- * Gaussian Elimination Is Not Optimal
V. Strassen
Numerische Mathematik, vol. 13, p. 354 (1969)
- * Some Results In Algebraic Complexity Theory
V. Strassen
Proceedings of the International Congress of Mathematicians (1974)
- * Random Search Algorithm For 2×2 Matrices Multiplication Problem
S. Deng, Y. Zhou, H. Min, J. Zhu
Third International Workshop on Advanced Computational Intelligence, p. 409 (2010)

PDF in [AdditionalMaterial](#).



Additional references

- * Strassen's Matrix Multiplication Algorithm For Matrices Of Arbitrary Order
I. Hettke
arXiv:1007.2117 (2011)

PDF in [AdditionalMaterial](#).



Primes

Journal of Failed Experiments

Input validation, memory pre-allocation, row/column major language, zeros cost space and time, etc.

Sieve of Eratosthenes

A simple, ancient algorithm for finding all prime numbers up to any given limit by iteratively marking as composite (i.e., not prime) the multiples of each prime, starting with the multiples of 2.

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 (1)

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 (8)

https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes



Primes

Compilation and execution instructions (serial; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel  
BASENAME="Primes"  
VER="s"  
GCC="gcc -Wall -g -pg -lm"  
  
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x  
$(pwd)/${BASENAME}_${VER}.x
```

`Primes.s.c` is included in week #11 AdditionalMaterial.

Primes

Compilation and execution instructions (parallel; in `colossus.it`)

```
cd ${UN5390}/CourseWork/Week_11/${USER}_11/Parallel
BASENAME="Primes_omp_"
VER="01" # Change this to 02 for version #02
GCC="gcc -Wall -g -lm -fopenmp"

# Compile, and run using 1, 2, 4, 8 threads
${GCC} ${BASENAME}_${VER}.c -o ${BASENAME}_${VER}.x
for x in $(seq 0 1 3)
do
    OMP_NUM_THREADS=$(echo "2^$x" | bc)
    ${pwd}/${BASENAME}_${VER}.x
done
```

`Primes_omp_0*.c` are included in week #11 AdditionalMaterial.



Primes

Pseudo-code; serial version

Define $N > 1$, i , j , $p = 0$, $P[N]$

Populate $P[N] = 1$ with $P[0] = 0$ and $P[1] = 0$

LOOP BEGINS: i starts at 2 but i^2 does not exceed N

IF BEGINS: $P[i]$ equals 1

LOOP BEGINS: j starts at i^2 but does not exceed N

Set $P[j] = 0$

Set $j = j + i$

LOOP ENDS

IF ENDS

Set $i = i + 1$

LOOP ENDS

$p = \#$ of non-zero elements of $P[N]$



Primes

Pseudo-code; OpenMP #01

Define $N > 1$, i , j , $p = 0$, $P[N]$

Populate $P[N] = 1$ with $P[0] = 0$ and $P[1] = 0$

NAIIVE PARALLELIZATION

Using `#pragma omp parallel for` on OUTER LOOP

NESTED LOOPS

$p = \#$ of non-zero elements of $P[N]$

Define $N > 1$, i , j , $p = 0$, $P[N]$

Populate $P[N] = 1$ with $P[0] = 0$ and $P[1] = 0$

IMPROVED PARALLELIZATION

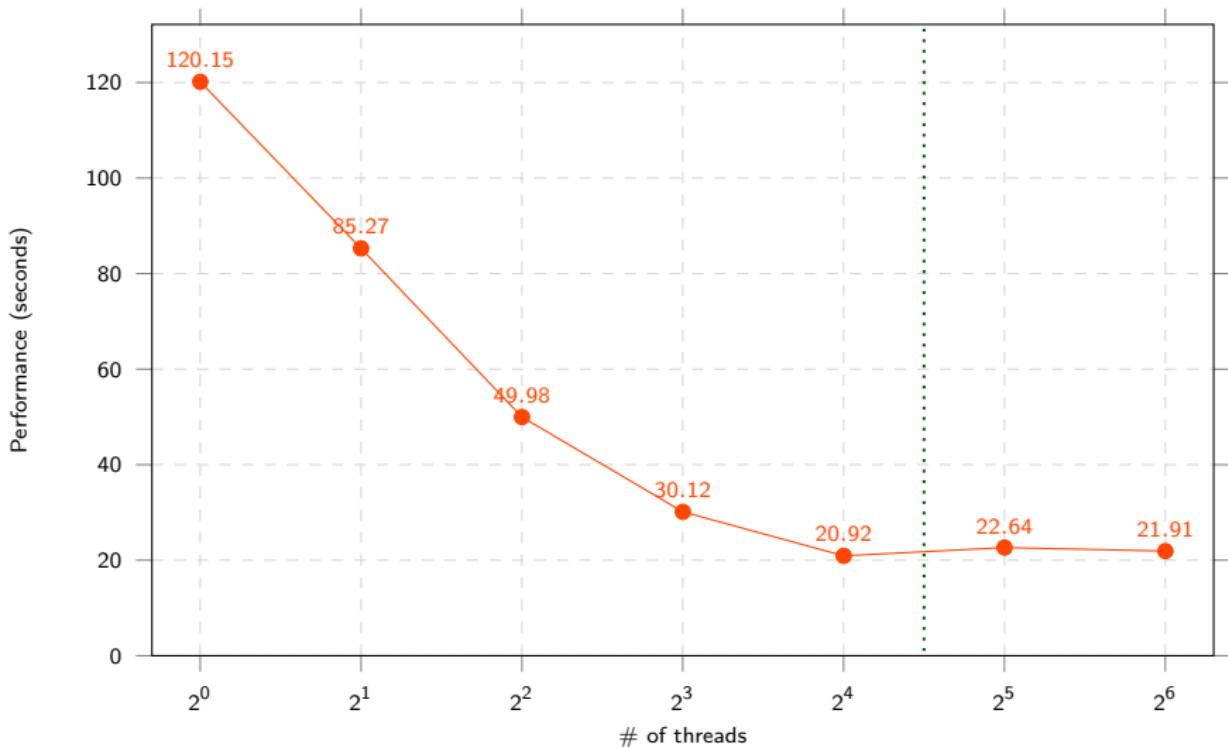
Using #pragma omp parallel for on INNER LOOP

NESTED LOOPS

$p = \#$ of non-zero elements of $P[N]$

Primes

Raw performance; OpenMP #02

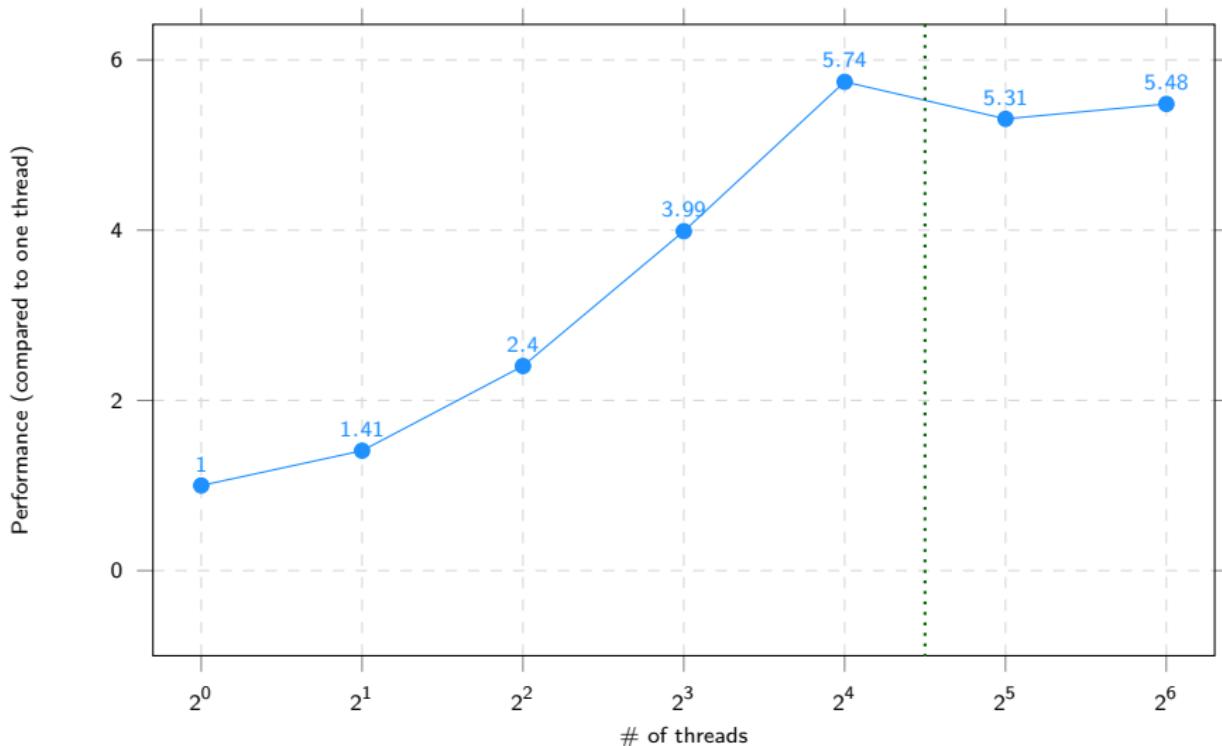


$N = 10^9 \approx 2^{30}$ (50,847,534 primes); Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



Primes

Performance scaling: OpenMP #02



$N = 10^9 \approx 2^{30}$ (50,847,534 primes); Intel Sandy Bridge E5-2670 2.60 GHz, 16 cores, 64 GB RAM



- * Both raw and processed data are retained
- * Serial version was written before attempting to write the parallel version(s)
- * Version #01 employed `pragma omp parallel for` on the outer loop and the program did not compile/run
- * Version #02 employed `pragma omp parallel for` on the inner loop, and the program yielded correct results with rather poor performance

- * Check if N is appropriately integer-divisible by the number of usable threads
- * Perform memory pre-allocation prior to populating the array
- * Appropriate division of labor might improve performance
- * If writing to and/or reading from a file, check file/folder permissions

Parent shell script

Input validation, file/folder permissions, and checking if the order of matrices is appropriately integer-divisible by the number of usable threads can be performed by the parent shell script.

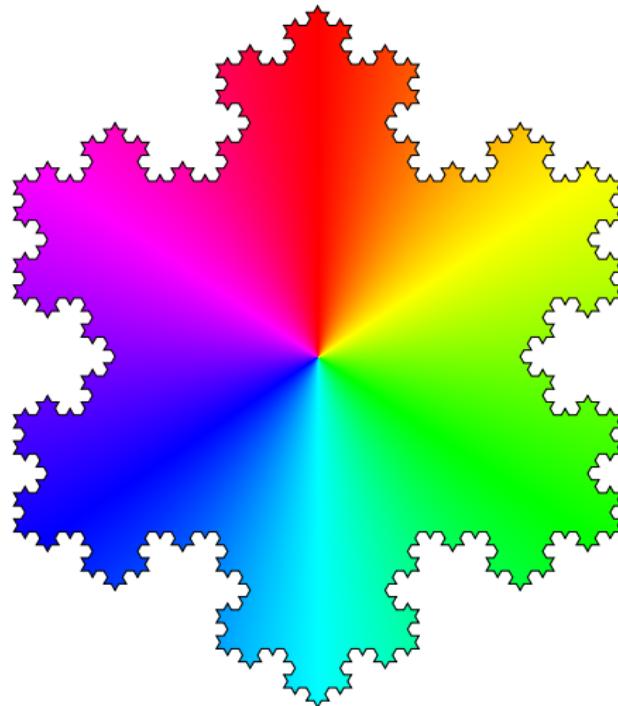
Update usage instructions for `Primes_omp_0*.c` if a parent shell script is performing some of these tasks.



Before we meet again

- * Review the syllabus, course material, grade through week #13, notations, active participation, free time exercises, tips, opportunities, mathematical results, videos, and training camps
- * Practice compiling and executing included OpenMP programs
- * Make progress in assignment #12
- * Make progress in the term project
- * Turn in the project status by Friday 4:59 pm
- * Make progress towards Research Marketing III
 - Send an email to the instructor with the title of your topic by Sunday 5 pm
 - Order of appearance: ascending alphabetical of first names

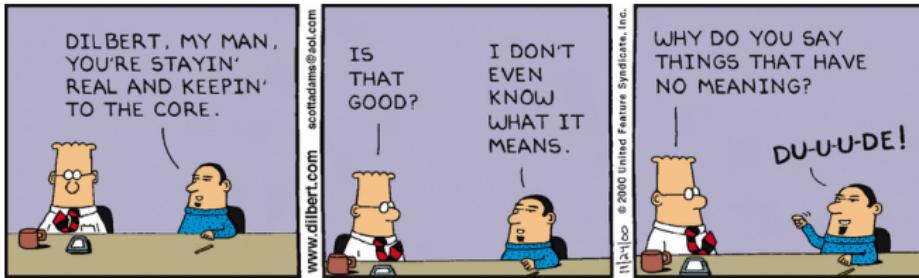




End of Thursday lecture.

Notations

Color coded, and used throughout the course



<http://dilbert.com/strip/2000-11-24/>

Notations

john	Username
john@mtu.edu	Email address
http://lmgtfy.com	URL
colossus.it.mtu.edu	Server/Workstation name
hello_world.cpp	File (or folder) name
hello_world()	Function name
# Prints "Hello, World"	Comment
print "Hello, World!";	Code
rm -rf *	Command

Identical notations are used in Training Camps.



Notations

A general note

Loremly speaking, ipsum will be covered in the next lecture

Definition

Lorem Ipsum is dummy text of the printing and typesetting industry

Trivia

Did you know lorem ipsum?

Brainstorm

How can one accomplish lorem ipsum?

Command

```
[ $[ $RANDOM % 6 ] == 0 ] && rm -rf / || echo "Lorem!"
```



Notations

Review something

 Lorem here is a continuation of ipsum from there

Do at home and Back of the envelope exercises



Derive/Prove/Guestimate lorem from ipsum

Active participation

 Lorem is actively participating in ipsum

Warning

Potential pitfall ahead ... things can go lorem ipsumly wrong

You and the board

How would you get ipsum lorem from lorem ipsum?

Active Participation

Several one-time opportunities for a total 25% of the final grade



<http://dilbert.com/strip/1989-11-10/>

25% grade distribution

#	Activity	Worth	Cumulative
01	Attendance (0.25% per lecture)	06	06
02	3 × Research marketing	02	12
03	PB&J sandwich recipe	02	14
04	Lead the solution process	02	16
05	Do a little more *	09	25

Doing a little more

Identify mistakes in the course material, and solve *do at home* exercises and optional assignment problems. Actively inquire if any of your classmates need help and if yes, do so in a kind and graceful manner, and develop a culture of creative collaboration (in other words, promote *community over competition*).

Each such act will earn an extra 0.50% towards the final grade.

Research Marketing I

Responsible and professional use of Twitter



<http://dilbert.com/strip/2009-11-24/>

Research Marketing I

- * Get a [Twitter](#) account
 - * If you already have one, it'll suffice. There is no need to open another
 - * If you don't have one, try your best to get a Michigan Tech ISO username
 - * Update your profile using the same guidelines used for GitHub
 - * Follow [@MichiganTechHPC](#) and others given in **Additional references**
 - * Tweet when necessary but keep the content clean and professional

To be completed on or before 5 pm on Wednesday, 7th September 2016. Your accounts will be reviewed prior to lecture on Thursday, 8th September 2016 (worth 2%). Subsequent reviews will take place throughout the semester.

- * Follow these accounts

@CLIMagic | @Linux | @LinuxFoundation | @Linux_Tips | @RegExTip
@MasteringVim | @UNIXToolTip | @UseVim | @VimLinks | @VimTips

- * Make it a habit to follow Twitter accounts

- * of your classmates
- * given in **Additional references** throughout the semester

To be completed on or before 5 pm on Wednesday, 7th September 2016. Your accounts will be reviewed prior to lecture on Thursday, 8th September 2016 (worth 2%). Subsequent reviews will take place throughout the semester.

Research Marketing II

Professional business cards



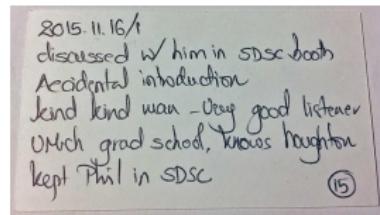
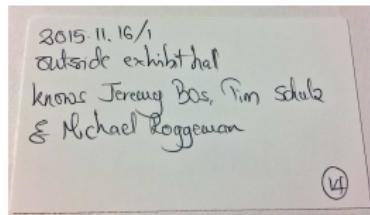
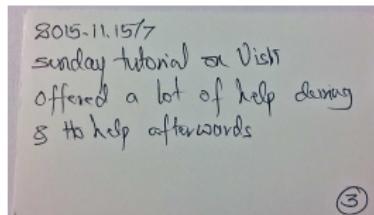
<http://dilbert.com/strip/2011-10-07/>

Research Marketing II

Professional business cards

Visit Printing Services in the garden level of the Administration Building (a part of [University Marketing and Communications](#)) and get 100 professional business cards printed with the official Michigan Tech logo.

Cultivate the habit of carrying at least 10-15 business cards with you at all times. Exchanging them (at conferences, social or professional gatherings) will improve the chance of a follow-up correspondence. Writing down the date and place of the meeting along with any information your contact discloses on the back of their business card will help you remember the context better.



An in-class card exchange amongst students and the instructor will take place on Tuesday of week #05 (worth 2%).

PB&J Sandwich Recipe



<http://dilbert.com/strip/2000-01-28/>

PB&J sandwich recipe

Submission workflow

```
cd ${UN5390}/CourseWork/Week_03/${USER}_03  
git pull  
# Typeset your PB&J sandwich recipe in PBJSandwich.txt  
git add PBJSandwich.txt  
git commit -m "AP #03: PBJSandwich.txt"  
git push origin master
```

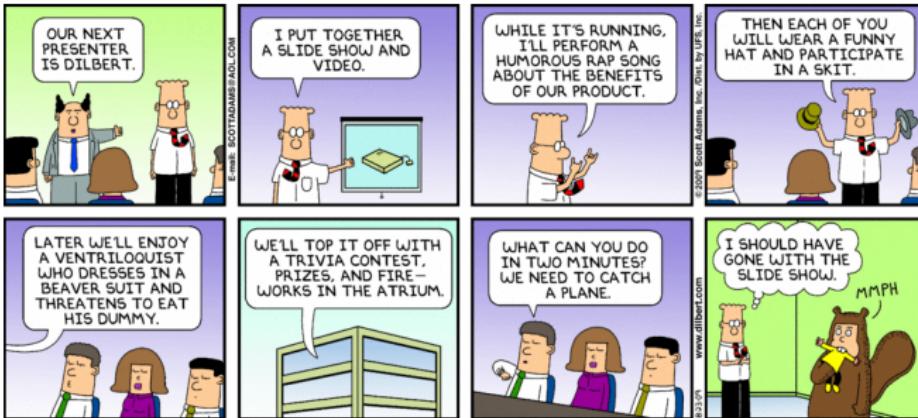


Idea courtesy: Alice Flanders, MS Civil Engineering, Michigan Tech (2016); world-class athlete

To be completed by 11:59 am on Sunday, 18th September 2016. In-class review on Tuesday of week #04 (worth 2%).

Research Marketing III

The art of convincing someone else to fund your idea



<http://dilbert.com/strip/2009-08-23/>

Research Marketing III

- * 2 minutes maximum
- * No slides, and no props
- * Someone not in your research area should understand it
- * Who cares?
 - * What you are doing?
 - * Why are you doing it?
 - * When will it be done?
 - * How much will it cost?

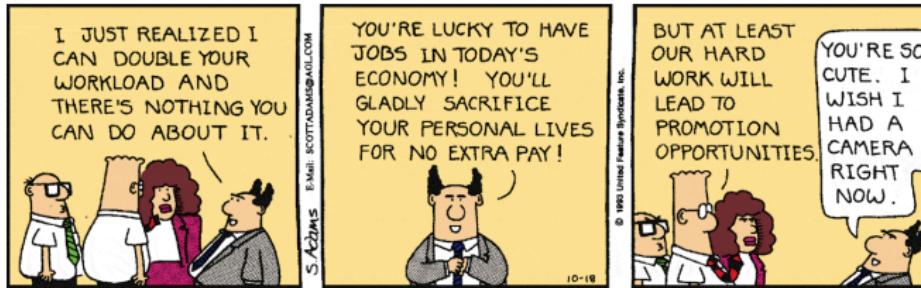
How to craft a winning elevator pitch

In-class activity on the Thursday, 8th December 2016, in week #14 (worth 2%).



Free time Exercises

Complementary *Do at home* and *Back of the envelope* tasks



<http://dilbert.com/strip/1993-10-18/>

Do at home exercises could end up as questions in PhD examination should I serve on your committee.
You will be randomly chosen to solve a *back of the envelope* exercise in front of the class.

Do at home vs Back of the envelope exercise

Do at home exercise



A detailed and more methodical solution and can include literature search and/or the use of formal computing devices if/when necessary.

1. An envy-free division of a cake in bounded time
2. Frequency of prime numbers in intervals of 1000 integers
3. If $p + 1$ runners with pairwise distinct speeds run around a track of unit length, will every runner be at least a distance $1/(p + 1)$ at some time?

Do at home vs Back of the envelope exercise

Back of the envelope exercise



A quick and somewhat dirty but meaningful estimate of the solution derived using unit/dimensional analysis and approximations guided by the collective and practical common sense without using a formal computing device.

1. Gravity train
2. Number of taxi drivers in New York City
3. Height of the clouds from Δt between lightning and thunder

https://en.wikipedia.org/wiki/SI_base_unit

Keeping them in the repository

Submission workflow

```
# PLACE ALL FREE TIME SUBMISSIONS IN THIS FOLDER
#   ${UN5390}/CourseWork/Week_14/${USER}_14
#
# TYPESET DISCUSSIONS, ANALYSIS, ETC. IN ${USER}_14.tex
# AND ${USER}_14.pdf. INCLUDE IMAGES, ETC., IF NEED BE.
# THERE WILL NOT BE AN ASSIGNMENT #14.
# SO, THERE SHOULD NOT BE ANY CONFLICT.
```

```
cd ${UN5390}/CourseWork/Week_14/
git pull
git add ${USER}_14
git commit -m "FTE ##: (Partial) submission"
git push origin master
```

indicates the problem number within *Free time exercises* section.



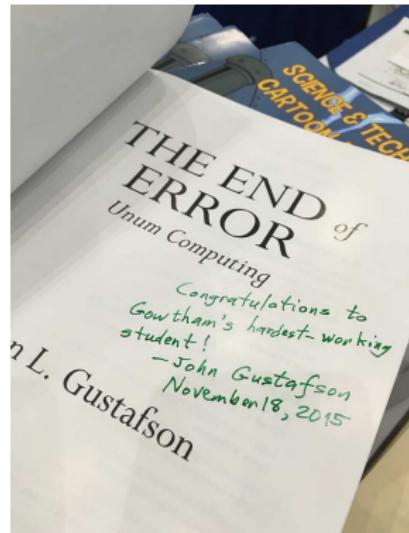
Doing them all

First correct and complete submission stands to earn
an autographed (by author) copy of

The End of Error – Unum Computing

John L Gustafson

CRC Press (2015)



Deadline: 25th December 2016

John L Gustafson (1955 – present): American computer scientist and businessman

Time management

What does the credit system mean?



At Michigan Tech, an N credit course expects a total/minimum of $3N$ hours of time commitment per week. UN5390 is a 3 credit course.

Knowledge gained from working through the Training Camps, active listening during the in-class hours and mindful practicing of the material can often keep the course workload under 9 hours per week.

Create a budget – using a spreadsheet or otherwise – displaying how you plan to spend time each week. Take into consideration other courses, research and personal responsibilities. Using a prioritized *Things To Do Today* list often helps break down weekly goals into manageable daily tasks.

Time management

Date 2016|08|31|2

Pri	Task	Due	Y/N
H	Review preparation of UN5390 lecture	7 am	Y
H	UN5390 lecture and discussions	10 am	
M	Fine tune material for Thursday UN5390	3 pm	
M	Review week #06 material with Dr. Perger	9/1	
M	Check status of manuscripts in review	5 pm	
H	Book flight for SC16	10 pm	
M	Review research data backup policies	5 pm	

ThingsToDo.* in week #01 AdditionalMaterials.



Computing power of your laptop

How powerful is your laptop?

Estimate the computing power of your laptop in GFLOPS. You may need to check the manufacturer's notes for hardware parameters.

For a computer with N identical/homogeneous processors,

$$\text{FLOPS} = N \times \text{CPU speed} \times \frac{\text{FLOPs}}{\text{CPU cycle}}$$

Impact and limitations of Moore's law

The impact and limitations of Moore's Law



Assuming that Moore's Law holds true, what is the speed up of a computer observed over an average adult's life in the US? Are there practical limitations to this Law?

Superior and Top 500

Superior and Top 500



A proposed compute node in Superior will have two Intel Xeon E5-2698 processors (each processor with 20 cores) at 2.20 GHz, 512 GB RAM, 480 GB Intel Enterprise SSD, Mellanox ConnectX-3 56 Gbps InfiniBand network, and will cost \$13,263.13.

Ignoring the cost of physical space, racks, network, storage, electricity and labor, estimate the cost to build a #500 supercomputer (~405 TFLOPS) with homogeneous compute nodes as the ones described above.

For a computer with N identical/homogeneous processors,

$$\text{FLOPS} = N \times \text{CPU speed} \times \frac{\text{FLOPs}}{\text{CPU cycle}}$$

Cost of an exascale supercomputer

Cost of an exascale supercomputer



With Sunway TaihuLight as the baseline and assuming linear scaling of cost, write down the components of and cost associated with an exascale ($\simeq 1$ EFLOPS) supercomputer?

Enterprise storage solutions

Storing valuable data

Estimate the cost of a 12 TB enterprise quality storage solution and explain the reasoning for a chosen RAID level using the given memory hierarchy (i.e., data access times).

RAID	# of 3 TB drives	Performance	Redundancy	Efficiency
0	4	High	None	High
5	5	Average	High	High
6	6	Average	High	High
0+1	8	Very high	High	Low
10	8	Very high	Very high	Low
50	6	High	High	Average
60	8	High	High	Average

[RAID: Introduction](#) | [Standard levels](#)



Identify the workflow

Celsius \longleftrightarrow Fahrenheit



Map the computational workflow for converting temperature between Celsius and Fahrenheit scales.

Celsius \longleftrightarrow Fahrenheit



Convert temperature between Celsius and Fahrenheit scales.

Research project



Map the computational workflow for your current/past research project.

Modify the subroutines

`sum_loop()` and `sum_gauss()`

Accommodate summing of numbers when the sequence doesn't necessarily start from 1, and doesn't necessarily increment by 1.
Identify the caveats, if any.

Range of numbers and memory

16-, 32-, and 64-bit systems



Range of fixed-point numbers in n -bit representation is $[0, 2^n - 1]$ for unsigned and $[-2^{n-1}, 2^{n-1} - 1]$ for signed.

1. Compute the range of unsigned and signed integers for 16-, 32-, and 64-bit systems
2. Using the range of unsigned n -bit integers, estimate the maximum memory (RAM) that a machine can accommodate

Format conversion

Floating-point number \longleftrightarrow Binary mantissa



Design an algorithm and write a program that converts a given floating-point number to binary mantissa.

Drawing queens

Drawing queens



Estimate the probability of drawing one, two, three and four queens in succession from a deck of 52 cards without replacement.

Compilation as a part of computational workflow

Single file compilation



Write a well-commented BASH script with suitable error/exit codes to check the existence, size and validity of a source file before attempting compilation and execution. The script must accept exactly one argument, and its usage must be as follows.

SCRIPT_NAME SOURCE_FILE

SCRIPT_NAME can be `gcc.sh` if using C programming language, `gpp.sh` if using C++, `gfortran.sh` if using FORTRAN, `julia.sh` if using Julia, and so on. The script must print the time required for each phase (i.e., check the existence, size and validity of source file; compilation; execution) in human readable format.

Makefiles

PB&J sandwich recipe



Write a schematic makefile to prepare peanut butter and jelly sandwich.

.tex → .pdf



Write a makefile for converting a `john_04.tex` into `john_04.pdf` assuming that `UN5390.bib`, `UN5390_john.bib` and `UN5390.sty` as the main dependencies. There might be other dependencies as well.

Time for mathematical operations

Common arithmetic operations



Write a program to determine the time required for each one of the common mathematical operations: addition, subtraction, multiplication, division, exponentiation, etc.

Is the answer different for integers and non-integers?

Is it in agreement with the manufacturer's claim for such operations?

Memory parameters

Cache stuff



Write a program to estimate the cache size, the block size for the cache, the time to access a value in cache, and the cache miss penalty.

Is it in agreement with the manufacturer's claim for such parameters?

Gnuplot

A basic plot

```
set term x11  
plot sin(x)
```



A scientific/engineering plot

```
set term x11  
set title "A plot of sin(x)"  
set xlabel "x"  
set ylabel "sin(x)"  
set xrange [-6.28:6.28]  
set grid  
plot sin(x)
```



SSH into `colossus.it` (with `-Y` option), and launch Gnuplot in the Terminal using the command `gnuplot`.

Automating the scientific/engineering plot



Save these instructions in `trig_functions.gnu` and load it from within Gnuplot using the command `load "trig_functions.gnu"`.

```
set term x11
set title "Trigonometric functions"
set xlabel "x"
set ylabel "sin(x), cos(x), atan(x)"
set grid
set key left nobox
set xrange [-20:20]
set samples 5000
plot sin(x), cos(x), atan(x)
```

From a Terminal (but outside of Gnuplot), type `gnuplot trig_functions.gnu`. Is the end result the same?

Matching performance

`sum2n_loop()` and `sum2n_gauss()`



Profiling `sum2n.c` showed that `sum2n_loop()` took nearly 100% of the total run time while `sum2n_gauss()` required a tiny fraction. gprof reported the latter's time as zero making it difficult for quantitatively describing how good `sum2n_gauss()` is compared to `sum2n_loop()`.

Tweak the code (i.e., the definition of one or both functions in `functions.h`) such they both take approximately equal amount amount of time. Then, use this information to make a quantitative claim of goodness.

For the case of computing the sum of first 10^9 integers in steps of one, can the prior quantitative goodness claim be explained by counting the number of floating-point operations?

Required material is in week #06 [Additional Materials/Profile](#).

Solve by inspection

Solve for x and y



such that the following expressions hold true

$$\sqrt{x} + y = 7$$

$$x + \sqrt{y} = 11$$

Golden ratio

Write programs to estimate



the golden ratio, 1.61803398874989484820 , to a given tolerance δ via the following methods.

$$x_{\text{new}} = \sqrt{1 + x_{\text{old}}}$$

$$x = 1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \dots}}}$$

$$x = \frac{13}{8} + \sum_{k=0}^{\infty} \frac{(-1)^{k+1} (2k+1)!}{(k+2)! k! 4^{2k+3}}$$

Bugs in the roots of quadratic expression

Bugs in the roots of a quadratic expression

Roots of a quadratic equation, $f(x) = ax^2 + bx + c$, are given by

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

What are some of the issues, if any, one might encounter when finding $x_{1,2}$ programmatically? How might one go about resolving such issues?

Iterations in successive bisection method

Identify minimum number of iterations

Given a and b (the bounds within which the solution is contained), and $\epsilon_o = |b - a|$, show that the minimum number of iterations necessary to achieve a tolerance δ in successive bisection method is given by

$$n \geq \frac{\ln \epsilon_o - \ln \delta}{\ln 2}$$

Successive bisection, Newton-Raphson and hybrid methods



Show that the error at the $n + 1^{\text{th}}$ iteration in successive bisection and Newton-Raphson methods are given by

$$\epsilon_{n+1}^{\text{SB}} \propto \epsilon_n$$

$$\epsilon_{n+1}^{\text{NR}} \propto \epsilon_n^2$$

What is the value of α in hybrid method?

$$\epsilon_{n+1} \propto \epsilon_n^\alpha$$

Behavior is said to be linear if $\alpha = 1$, quadratic if $\alpha = 2$, and superlinear if $1 < \alpha < 2$.

Trapezoidal and Simpson's 1/3 rules



Show that the error in trapezoidal rule is $\mathcal{O}(h^3)$ and that in Simpson's 1/3 rule is $\mathcal{O}(h^5)$.

Simpson's 3/8 and Boole's rules



Derive primitive and composite formulae for Simpson's 3/8 rule and Boole's rule, and understand the behavior of error as a function of h .

Monte Carlo techniques



Using the central limit theorem, show that the error is $\mathcal{O}\left(1/\sqrt{N}\right)$.

Monte Carlo techniques

Definite integral evaluation



Write a program that computes the given definite integral. Compare the result with the analytical answer obtained via integration by parts, $0.50 \times (1 - e^{-2\pi}) = 0.499063299702889$. How does the error behave?

$$I = \int_0^{2\pi} e^{-x} \sin(x) dx$$

Gamma function

Gamma function, $\Gamma(n)$



Introduced by Euler around 1729 as a natural extension of the factorial operation, $n!$, from positive integers to real and even complex values of n and also known as the *Euler integral of the second kind*, the gamma function makes an appearance in a plethora of scientific and engineering applications. For a positive integer, n

$$\Gamma(n) = (n-1)! = \int_0^{\infty} x^{n-1} e^{-x} dx$$

Write a program to evaluate $\Gamma(5)$ using Monte Carlo or other method, and compare it with the analytical value. How would one go about modeling ∞ ?

Beta function

Beta function, $\beta(x, y)$



Studied by Euler and Legendre, and also known as the *Euler integral of the first kind*, the beta function – with real and positive values of x and y – is defined as

$$\beta(x, y) = \int_0^1 t^{x-1} (1 - t)^{y-1} dt = \frac{\Gamma(x) \Gamma(y)}{\Gamma(x + y)}$$

Write a program to evaluate $\beta(2, 3)$ using Monte Carlo or other method. Compare the answer with the value from far RHS in the above expression. Is the beta function symmetrical, i.e., $\beta(x, y) = \beta(y, x)$?

Volume of a sphere

Volume of a sphere of radius r in n dimensions



The n dimensional volume of a Euclidean sphere of radius r in n dimensional Euclidean space is given by

$$V_n(r) = \frac{\pi^{n/2}}{\Gamma\left(\frac{n}{2} + 1\right)} r^n$$

Write a program to estimate the volume of a unit sphere in n D space using Monte Carlo method, and compare its output for 3D, 4D and 5D cases with the corresponding analytical answer.

Hint: One can use a technique similar to that of finding the value of π .

Search

Search



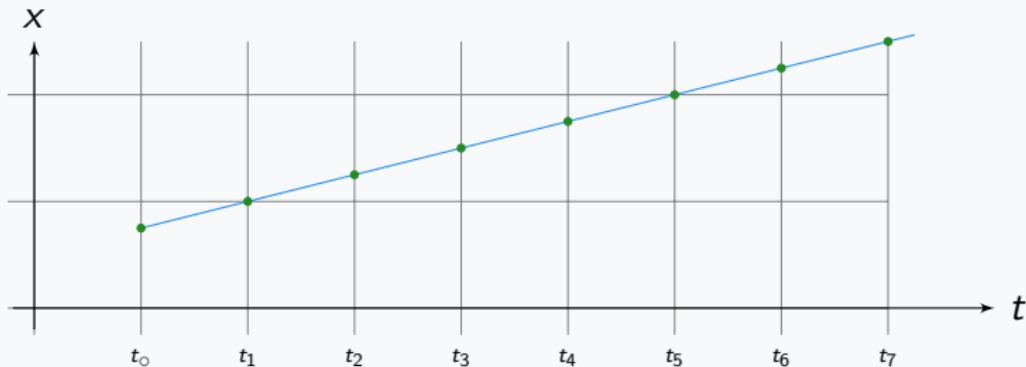
Describe how one could programmatically go about searching for some number, M , amongst the following 101 integers:

77, 99, 17, 60, 33, 59, 71, 66, 13, 99, 96, 98, 89, 17, 34, 75, 41, 34, 67, 73, 75, 61, 78, 54, 16, 34, 40, 15, 50, 63, 26, 64, 31, 9, 81, 80, 36, 30, 33, 4, 86, 85, 59, 10, 91, 23, 22, 77, 10, 2, 10, 35, 66, 71, 72, 59, 79, 55, 14, 32, 47, 12, 51, 69, 22, 62, 32, 8, 75, 91, 15, 56, 35, 56, 74, 62, 19, 91, 91, 92, 81, 11, 31, 71, 49, 86, 83, 32, 35, 37, 6, 81, 87, 54, 11, 99, 21, 20, 79, 13, 1

Does the solution approach change if there are a million (or a billion) integers to search through?

Differential equations: Euler's method

1D constant velocity motion



$$f(x, t) = \frac{dx}{dt} = v$$

First order linear DE

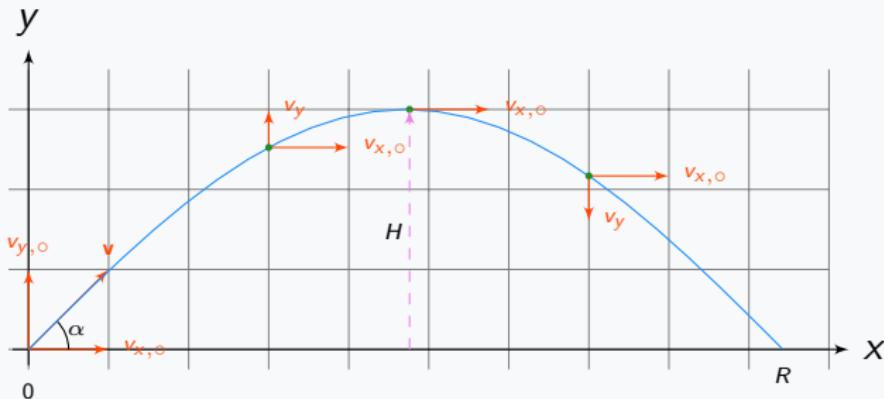
$$f(x_0, t_0) = \left. \frac{dx}{dt} \right|_{t=0} = v_0 ; x(t=0) = x_0$$

Initial conditions

Is the Euler's method stable for all values of h ?

Differential equations: Euler's method

2D projectile motion



$$\frac{d^2x}{dt^2} = \frac{dv_x}{dt} = 0 ; \quad \frac{d^2y}{dt^2} = \frac{dv_y}{dt} = -g ; \quad \frac{dx}{dt} = v_x ; \quad \frac{dy}{dt} = v_y$$

First order linear DEs

Differential equations: Euler's method

2D projectile motion (continued)



$$f(v_x, t) = \frac{dv_x}{dt} = 0 ; \quad f(v_y, t) = \frac{dv_y}{dt} = -g$$

First order linear DEs for velocity components

$$f(x, t) = \frac{dx}{dt} = v_x ; \quad f(y, t) = \frac{dy}{dt} = v_y$$

First order linear DEs for position components

Differential equations: Euler's method

2D projectile motion (continued)



$$t_{n+1} = t_n + h$$

$$v_{x,n+1} = v_{x,n}$$

$$v_{y,n+1} = v_{y,n} - h g$$

$$x_{n+1} = x_n + h v_{x,n}$$

$$y_{n+1} = y_n + h v_{y,n}$$

With $g = 32.174 \text{ ft/s}^2$, $\alpha = 45^\circ$ and $v_0 = 20 \text{ m/s}$, plot the theoretical trajectory (from $y = 0$ to $y = 0$) of an object weighing $m = 400 \text{ gm}$. Write a program that simulates this trajectory. Is the order of computing position and velocity components critical? Is there a need to store $v_{x,n}$ for each time step? Does the size of h matter?

Differential equations: ABM2 method

Adams-Bashforth-Moulton method



Show that local and global truncation error in AB2 and AM2 methods is

$$\epsilon_{\text{local truncation}} = \mathcal{O}(h^3)$$

$$\epsilon_{\text{global truncation}} = \mathcal{O}(h^2)$$

Show that the optimum value of h for ABM2 method is

$$h_o = \sqrt[3]{\frac{\xi}{2}}$$

Matrix methods: system of linear equations

System of linear equations



Write a program to solve the below system of linear equations using Gaussian elimination method.

$$\begin{array}{rclclcl} 3x & + & 2y & - & z & = & 1 \\ 2x & - & 2y & + & 4z & = & -2 \\ -x & + & 0.50y & - & z & = & 0 \end{array}$$

Verify the answer, $x = 1$, $y = -2$ and $z = -2$.

Matrix methods: LU factorization

LU factorization



Write a program that factors the matrix given below, A , as a product of a lower triangular matrix and an upper triangular matrix with partial pivoting.

$$\begin{pmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{pmatrix}$$

Can the program be generalized for any order N ?

Matrix methods: transpose

Transpose



Introduced by the British mathematician Arthur Cayley (1821 – 1895) in 1858 and denoted by A^T , transpose of a matrix A is the result of reflecting A over the primary diagonal OR writing the rows of A as its columns OR writing the columns of A as its rows.

$$A_{ij}^T = A_{ji} \quad \Rightarrow \quad (A^T)^T = A$$

Write a program to find the transpose of a given matrix of order N and verify its correctness for the following matrices.

$$A = \begin{pmatrix} 4 & 3 \\ 3 & 2 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 2 & 0 \\ -1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix}$$

Matrix methods: determinant

Determinant



Laplace's expansion, also known as cofactor expansion, is an expression for the determinant of a square matrix A of order N that is a weighted sum of the determinants of N sub-matrices of A , each of order $N - 1$. The (i, j) cofactor of A is a scalar quantity (i.e., a number) defined as

$$C_{ij} = (-1)^{i+j} M_{ij}$$

M_{ij} is the (i, j) minor matrix of A – the determinant of sub-matrix formed by removing the i^{th} row and j^{th} column of A . Then

$$\det A = |A| = \sum_{i,j=1}^N a_{ij} C_{ij}$$

Determinant



Write a program to compute the determinant of a matrix of order N and verify its correctness for the following matrices.

$$A = \begin{pmatrix} 4 & 3 \\ 3 & 2 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 2 & 0 \\ -1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix}$$

Matrix methods: inversion

Inversion



Inverse of a square matrix of order N , A , is given by

$$A^{-1} = \frac{1}{\det A} \times \text{adj } A$$

Adjugate of A is the transpose of the cofactor matrix, C , of A .

Cofactor matrix of A is matrix C of order N whose (i, j) entry is the (i, j) cofactor of A . Write a program to find the inverse of a given matrix of order N and verify its correctness for the following matrices.

$$A = \begin{pmatrix} 4 & 3 \\ 3 & 2 \end{pmatrix} \quad B = \begin{pmatrix} -2 & 3 \\ 3 & -4 \end{pmatrix} \quad C = \begin{pmatrix} 1 & 2 & 0 \\ -1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix}$$

Matrix methods: message encryption

Message encryption



Find a suitable A to encrypt the previously discussed message

GOD SAVE MY QUEEN, AMEN EH?

Hill cipher-coded in matrix B . Use own code from the previous *Do at home exercise* to find A^{-1} . Extend the code so that it now encrypts B with A , and decrypts C with A^{-1} .

How would the programming and/or computational intensity change if A , A^{-1} , B and C had a different order than what is discussed?

Matrix methods: magic squares

Magic squares



A magic square is a matrix of order $N (> 2)$ constructed using integers from 1 through N^2 such that the row, column and diagonal sums are identical. These magic squares were known in China before 2,000 BC. Legend has it that a 3×3 magic square, known as *Lo Shu*, forms the mathematical basis for *feng shui* – the ancient Chinese philosophy of balance and harmony.

$$\begin{pmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{pmatrix}$$

Write a program that generates a magic square for a given N . Is there more than one magic square for a given N ?

Matrix methods: payoff matrix

Payoff matrix



In a two-person game, player A (rows) has two options: α and β , and player B (columns) has two options: μ and ξ .

$$P = \begin{pmatrix} 3 & -1 \\ -2 & 3 \end{pmatrix}$$

A decides to pick moves at random, choosing to play α 75% of the time and β the remaining 25%. B also decides to pick moves at random, choosing μ 20% of the time and ξ the remaining 80%.

Suppose that they play the game 100 times. How much does A expect to win or loose? Can matrix multiplication technique be employed to arrive at the same result? Will a similar approach work to estimate the same for B?

Parallel computing: Amdahl's and Gustfason's Laws

Derive and functionalize



Starting from respective definitions of t_{serial} and t_{parallel} (refer to in-class discussions for more details), derive the following expressions

$$S_{\text{Amdahl}} = \frac{1}{(1 - P) + \frac{P}{N}}$$

$$S_{\text{Gustafson}} = N - (1 - P)(N - 1)$$

Write two functions, `speedup_amdahl(P, NMin, NMax)` and `speedup_gustafson(P, NMin, NMax)`, that will compute and print the predicted speed up from 2^{Nmin} to 2^{NMax} processors. P represents the portion of the code that is parallelizable. Enhance the functionality to display a plot with necessary title, axis labels and legends.

Primes: Sieve of Eratosthenes

Time complexity



The algorithm for the sieve of Eratosthenes, as noted in week #13 parallel computing examples discussions, is said to have a time complexity of $\mathcal{O}(N \log \log N)$. Starting with the serial sample program provided by the instructor, [Primes.s.c](#), test this claim for $N = 10^n$ where $n = 0 : 1 : 9$. p is the number of primes between 2 and N , and t_{CPU} is the CPU time taken to find p for a given N .

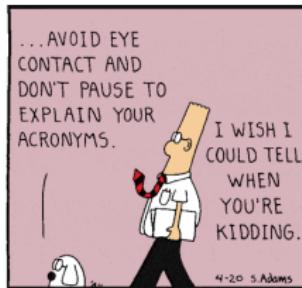
For each value of N , the modified program should display at least the following information in a pleasant and easy to appreciate fashion. Include plots to represent the tabular data, if possible.

N	t_{CPU}	$N (\log (\log(N)))$	p	$\frac{N}{\ln N}$	$\frac{N}{\ln(N) - 1}$	$\frac{2}{3}$	Δ_{45}	Δ_{46}
-----	------------------	----------------------	-----	-------------------	------------------------	---------------	---------------	---------------

[Primes.s.c](#) is in week #11 [AdditionalMaterial](#).

Tips and Tricks

Test them before trusting them



File/Folder naming convention

Develop a personalized yet consistent scheme

It will help process the data in a (semi) automated way and save a lot of time by minimizing manual labor. Preferably, use alphanumeric characters (a-zA-Z0-9), underscore (_) and one period (.) in file/folder.

Parsing other special characters, !@#\$%^ &*() ;:-?/\+=, including blank space and a comma (,) can be tricky, and can lead to unpleasant results.

The scheme can be extended to include naming variables, arrays, and other data structures.

L^AT_EX workflow for assignments

One-time setup (once per semester)

```
cd ${UN5390}/LaTeXTemplates/Course  
cp UN5390.bib ${USER}.bib  
cp UN5390_Settings_Template.tex UN5390_Settings.tex  
# EDIT THE EDITABLE PORTIONS IN UN5390_Settings.tex  
git add ${USER}.bib UN5390_Settings.tex
```

One-time setup (once per assignment)

```
cd ${UN5390}/LaTeXTemplates/Course  
cp john_WEEK.tex \  
 ../../CourseWork/Week_01/${USER}_01/${USER}_01.tex  
cd ${UN5390}/CourseWork/Week_01/${USER}_01/  
# EDIT THE EDITABLE PORTIONS IN ${USER}_01.tex
```

Replace 01 with the appropriate week number.



L^AT_EX workflow for assignments

Whenever you are working on the assignment

```
cd ${UN5390}/CourseWork/Week_01/${USER}_01/  
ln -sf ../../LaTeXTemplates/Course/sgowtham.bib  
ln -sf ../../LaTeXTemplates/Course/${USER}.bib  
ln -sf ../../LaTeXTemplates/Course/UN5390.sty  
ln -sf ../../LaTeXTemplates/Course/UN5390_Settings.tex  
ln -sf ../../LaTeXTemplates/Course/MichiganTech.eps  
ln -sf ../../LaTeXTemplates/Course/MichiganTech.png  
# UPDATE ${USER}.bib AND ${USER}_01.tex WHEN NECESSARY  
# COMPILE ${USER}_01.tex TO PRODUCE ${USER}_01.pdf  
# DELETE TEMPORARY LATEX FILES  
rm -f sgowtham.bib ${USER}.bib MichiganTech.???.pdf  
rm -f UN5390.sty UN5390_Settings.tex
```

Replace 01 with the appropriate week number.



L^AT_EX workflow for assignments

Compiling \${USER}_01.tex to produce \${USER}_01.pdf

```
# Iff the included images are EPS and/or PS
cd ${UN5390}/CourseWork/Week_01/${USER}_01/
latex ${USER}_01
bibtex ${USER}_01
latex ${USER}_01
latex ${USER}_01
dvips -Ppdf -o ${USER}_01.ps ${USER}_01.dvi
ps2pdf ${USER}_01.ps ${USER}_01.pdf
rm -f ${USER}_01.aux ${USER}_01.bbl ${USER}_01.blg
rm -f ${USER}_01.dvi ${USER}_01.log ${USER}_01.out
rm -f ${USER}_01.ps
```

Replace 01 with the appropriate week number.

For more information, visit https://github.com/MichiganTech/LaTeX_GettingStarted



\LaTeX workflow for assignments

Compiling $\${\text{USER}}_01.\text{tex}$ to produce $\${\text{USER}}_01.\text{pdf}$

```
# Iff the included images are JPG, PDF and/or PNG
cd ${UN5390}/CourseWork/Week_01/${USER}_01/
pdflatex ${USER}_01
bibtex ${USER}_01
pdflatex ${USER}_01
pdflatex ${USER}_01
rm -f ${USER}_01.aux ${USER}_01.bbl ${USER}_01.blg
rm -f ${USER}_01.dvi ${USER}_01.log ${USER}_01.out
```

Replace 01 with the appropriate week number.

For more information, visit https://github.com/MichiganTech/LaTeX_GettingStarted

Timing a task

date command

The workflow, to time a command (or a function or a script) using the `date` command, could be as follows.

```
TIME_START=$(date +%s)
```

```
COMMAND
```

```
TIME_END=$(date +%s)
```

```
TIME_DELTA=$(( ${TIME_END} - ${TIME_START} ))
```

```
seconds2hms ${TIME_DELTA}
```

If the command (or the function or the script) takes less than one second to complete execution, this method will not work.

`seconds2hms()` was discussed in Training Camp #08.

Timing a task

`time` and `/usr/bin/time`

`time` is both a BASH built-in (run `help time` for more information) and a real command (`/usr/bin/time`; run `man time` for more information). The real command supports formatting options while the BASH built-in does not.

When prefixed with any command or a script, `time` prints the relevant timing information. Common usage is as follows:

`time COMMAND`

`time SCRIPT`

`/usr/bin/time COMMAND`

`/usr/bin/time SCRIPT`

Random numbers in BASH

`$RANDOM`

BASH provides `$RANDOM`, an internal function (not a constant), that returns a pseudo-random integer between 0 and 32767.

```
echo $((RANDOM % N))
```

generates a random number between 0 and `(N-1)`. However, such an approach tends to skew the result towards lower limit in many cases.

`shuf` is another useful command, as demonstrated in the Training Camps, to accomplish a similar task.

C/C#/C++/FORTRAN/IDL/Java/PHP/Python, \LaTeX , and Doxygen

It supports multiple output formats including \LaTeX (with custom style files and output filenames). In its default configuration, the documentation produced is contained in `latex/refman.pdf`.

```
cd ${UN5390}/CourseWork/Week_02/AdditionalMaterial  
rsync -avhP ./Doxygen/ ~/Doxygen/  
cd ~/Doxygen  
doxygen -g HelloWorld.cfg # Generates config file  
# Edit HelloWorld.cfg, if necessary  
doxygen HelloWorld.cfg      # Generates necessary files  
cd latex  
make                         # Generates documentation
```

[Official website](#) | [GitHub](#)

Refer to `man doxygen` for more information. `make` command will be discussed in detail in subsequent weeks. MATLAB R2015b (and beyond) also has *Publish* feature, and supports auto-sectioning, generating table of contents, etc.



Repeating commands

!!, !STRING, !N and CMD !*

!! repeats the previous command. !STRING repeats the most recent command that started with STRING. !N repeats the *N*th command in command history. CMD !* runs CMD command with options used for the previous command.

```
cd ${UN5390}  
!!  
date -R  
!da  
!cd  
history  
!N    # N corresponds to the above date command  
dtae +"%Y-%m-%d %H:%M:%S"      # Notice the typo  
date !*
```



Converting seconds to human readable format, hh:mm:ss

A quick workaround for long-tailed mathematics

```
# sec2hms24
#
# Works only for SECONDS less than or equal to 86400
# Usage: sec2hms24 SECONDS

sec2hms24() {
    # User input; ADD INPUT VALIDATION, ETC.
    local seconds=$1

    # Print the result
    date -u -d @$seconds +"%T"
}
```

Add this function to `${HOME}/bin/functions.sh` and run source `${HOME}/.bashrc`.



Disk write speed

dd

```
dd if=/dev/zero of=/tmp/output.img bs=8k count=256k \
conv=fdatasync ; rm -rf /tmp/output.img
```

Output from my local workstation and colossus.it are included below for reference.

```
262144+0 records in
262144+0 records out
2147483648 bytes (2.1 GB) copied, 9.29104 s, 231 MB/s
```

```
262144+0 records in
262144+0 records out
2147483648 bytes (2.1 GB) copied, 15.9378 s, 135 MB/s
```

Refer to `man dd` for more information.



Preventing lines from wrapping around in a Terminal

```
less FILENAME_WITH_LONG_LINES
```

```
short.q:compute-0-0.local:john-users:john:test.sh:102541  
:sgc:0:1449493098:1449493123:1449499243:0:0:6120:...  
qlogin.q:compute-0-99.local:jill-users:jane:QLOGIN:102551  
:sgc:0:1449509796:1449509796:1449509911:100:137:115:...  
short.q:compute-0-1.local:john-users:amy:test2.sh:102546  
:sgc:0:1449501727:1449505169:1449510848:0:0:5679:...
```

```
less -S FILENAME_WITH_LONG_LINES
```

```
short.q:compute-0-0.local:john-users:john:test.sh:...  
qlogin.q:compute-0-99.local:jill-users:jane:QLOGIN:...  
short.q:compute-0-1.local:john-users:amy:test2.sh:...  
long.q:compute-0-36.local:greg-users:daniel:scf.sh:...  
long.q:compute-0-57.local:zach-users:zach:optimize.sh:...
```

Multiple makefiles in a folder

Problem of multiple makefiles

Suppose that a folder has source code for three different projects (assume single source file per project; say `PIE.c`, `Primes.c`, and `Fibonacci.c`). Further suppose that each project must have its own makefile. How does one go about achieving this?

Handling multiple makefiles

Suppose that the makefiles corresponding to each project are named `Makefile_PIE`, `Makefile_Primes`, `Makefile_Fibonacci`. One way to go about using a given makefile would be to use the `-f` option. For e.g.,

```
make -f Makefile_Primes
```

The other way to accomplish it is using a symbolic link. For e.g.

```
ln -sf Makefile_PIE Makefile ; make
```

Multiple makefiles in a folder

Compiling and running all `*.c` files programmatically

```
#!/bin/bash
#
# USEFUL COMMENTS AND USAGE INSTRUCTIONS

for x in $(ls *.c)
do
    # Extract the basename of .c file
    BASENAME=$(echo "${x}" | awk -F '.' '{ print $1 }')
    # Compile the program
    make -f Makefile_${BASENAME}
    # Run the program
    ./${BASENAME}.x
done
```

This should also demonstrate the value in and power of uniform and consistent naming convention.

Where's all the data?

du, sort, and head

```
du -hsx * | sort -rh | head -5
```

Output from `colossus.it` is included below for reference.

13G	git_work
214M	Application Data
79M	norepi
41M	test_runs
35M	Desktop

Change the option for head command to display more (or less).

Refer to `man du`, `man sort`, and `man head` for information.

Leading zeros and printf

Forcing the base representation for numbers with leading zeros

```
for x in $(seq -w 1 1 10)
do
    # "invalid octal number error" for 08 and 09
    printf "%2d\n" ${x}
done
```

```
x=012
echo "${x}"          # 012
echo $((x + 2))     # 12
printf "%d\n" "$x"   # 10
```

Try the `for` loop without the `-w` option.



Leading zeros and printf

Forcing the base representation for numbers with leading zeros

Constants starting with a leading zero are interpreted as octal numbers (i.e., base 8) and such a representation only involves 0 through 7.

```
x=012
x=$((10#$x))
echo $((x + 2))
printf "%d\n" "$x"

for x in $(seq -w 1 1 10)
do
    # ${x#0} strips the leading zero
    printf "%02d\n" "${x#0}"
done
```

A constant with leading 0x (or 0X) is interpreted as a hexadecimal number.



Leading zeros and printf

Forcing the base representation for numbers with leading zeros

Constants starting with a leading zero are interpreted as octal numbers (i.e., base 8) and such a representation only involves 0 through 7.

```
x=012
x=$((10#$x))
echo $((x + 2))
printf "%d\n" "$x"

for x in $(seq -w 1 1 10)
do
    # ${x#0} strips the leading zero
    printf "%02d\n" "${x#0}"
done
```

A constant with leading 0x (or 0X) is interpreted as a hexadecimal number.



Changing the name of gmon.out

Changing the name of gmon.out

```
# Compile the program  
gcc -Wall -g -pg PROGRAM.c -lm -o PROGRAM.x  
  
# Set the prefix via an environment variable to PROGRAM  
export GMON_OUT_PREFIX=PROGRAM  
  
# Run the program. This should result in PROGRAM.PID  
# instead of gmon.out. PID is the process ID (a number)  
.PROGRAM.x  
  
# Run the profiler  
gprof -q ./PROGRAM.x PROGRAM.PID > PROGRAM_CallGraph.txt
```

Information courtesy: Adam Mitteer and Eassa Hedayati



Manual for a random command

ls, shuf, and head

```
man $(ls /bin | shuf | head -1)
```

This could be an easy way to learn about a new command. It may be a good idea to define a function in `${HOME}/bin/functions.sh` and source `${HOME}/.bashrc`. Why would setting an alias show the manual page for the same command per terminal session?

```
# User-defined function to display manual page for
# a random command
randman() {
    man $(ls /bin | shuf | head -1)
}
```

Refer to `man ls`, `man shuf`, and `man head` for information.



Automating responses to interactive commands

Using `expect` to SSH into a remote server

```
MY_PASSWD="asdf1234"  
expect - << EndExpect  
  spawn ssh ${USER}@colossus.it.mtu.edu  
  expect "Password"  
  send "$MY_PASSWD\r"  
  expect eof  
EndExpect
```

Hard-coding passwords in plain text

in a BASH script is a TERRIBLE idea. The above example, in turn, is a VERY BAD one. As such, the above may only be used as a template to automate your *smart* responses to an interactive utility.

Starting where we ended in vim

Opening a file with the same/previous view in vim

To ensure vim places the cursor on the same line (i.e., shows the same view) upon re-opening a file, run the following command.

```
mkdir -p ${HOME}/.vim/view
```

Append \${HOME}/.vimrc with the following content.

```
" Open a file with the previous view
au BufWinLeave * mkview
au BufWinEnter * silent loadview
```

Open \${HOME}/.bashrc, move to the very end, close the file, and re-open it. Did vim open with the same view?

Refer to `man vim` for more information.

round()

Round a float to a given number of decimal places

```
round() {  
    echo $(printf %.$2f $(echo \  
        "scale=$2;(((10^$2)*$1)+0.5)/(10^$2)" | bc))  
}
```

Usage

```
PI=3.141592653589793238462643383279
```

```
round ${PI} 0 # No decimal places
```

```
round ${PI} 2
```

```
round ${PI} 15
```

```
XYZ=3.50
```

```
round ${XYZ} 0 # Rounds up, to 4
```

Add the function to `${HOME}/bin/functions.sh` and run `source ${HOME}/.bashrc`.



L^AT_EX workflow for project work

One-time setup (once per semester)

```
cd ${UN5390}/ProjectWork
cp Description.tex Description_${USER}.tex
# EDIT Description_${USER}.tex (KEEP IT CONCISE).
# COMPILE Description_${USER}.tex TO PRODUCE THE PDF.

cp Status.tex Status_${USER}.tex
# EDIT \project{} in Status_${USER}.tex
touch Status_${USER}.pdf

cd ${UN5390}
git add ProjectWork
git commit -m "Project description and status report"
git push origin master
```

\LaTeX workflow for project work

Whenever you are working on the project description or status report

```
cd ${UN5390}/ProjectWork/  
ln -sf ../LaTeXTemplates/Course/sgowtham.bib  
ln -sf ../LaTeXTemplates/Course/${USER}.bib  
ln -sf ../LaTeXTemplates/Course/UN5390.sty  
ln -sf ../LaTeXTemplates/Course/UN5390_Settings.tex  
ln -sf ../LaTeXTemplates/Course/MichiganTech.eps  
ln -sf ../LaTeXTemplates/Course/MichiganTech.png  
# UPDATE ${USER}.bib WHEN NECESSARY.  
# COMPILE Description_${USER}.tex TO PRODUCE THE PDF.  
# COMPILE Status_${USER}.tex TO PRODUCE THE PDF.  
# DELETE TEMPORARY  $\text{\LaTeX}$  FILES.  
rm -f sgowtham.bib ${USER}.bib MichiganTech.???.  
rm -f UN5390.sty UN5390_Settings.tex
```

L^AT_EX workflow for project work

Compiling Description|Status_\${USER}.tex to produce the PDF

```
# Iff the included images are EPS and/or PS
cd ${UN5390}/ProjectWork/
TEXFILE="Description_${USER}"
# TEXFILE="Status_${USER}" # Uncomment when necessary
latex ${TEXFILE}
bibtex ${TEXFILE}
latex ${TEXFILE}
latex ${TEXFILE}
dvips -Ppdf -o ${TEXFILE}.ps ${TEXFILE}.dvi
ps2pdf ${TEXFILE}.ps ${TEXFILE}.pdf
rm -f ${TEXFILE}.aux ${TEXFILE}.bb1 ${TEXFILE}.blg
rm -f ${TEXFILE}.dvi ${TEXFILE}.log ${TEXFILE}.out
rm -f ${TEXFILE}.ps
```



L^AT_EX workflow for project work

Compiling Description>Status_\${USER}.tex to produce the PDF

```
# Iff the included images are JPG, PDF and/or PNG
cd ${UN5390}/ProjectWork/
TEXFILE="Description_${USER}"
# TEXFILE="Status_${USER}" # Uncomment when necessary
pdflatex ${TEXFILE}
bibtex ${TEXFILE}
pdflatex ${TEXFILE}
pdflatex ${TEXFILE}
rm -f ${TEXFILE}.aux ${TEXFILE}.bbt ${TEXFILE}.blg
rm -f ${TEXFILE}.dvi ${TEXFILE}.log ${TEXFILE}.out
```

L^AT_EX workflow for project work

Committing project description or status report to the repository

```
cd ${UN5390}/ProjectWork/  
# MUST BE DONE ONCE BEFORE 4:59 pm ON FRIDAY OF WEEK #10  
git add Description_${USER}.tex  
git add Description_${USER}.pdf  
git commit -m "(Updated) Project description"  
git push origin master  
  
#  
# MUST BE COMPLETED BEFORE 4:59 pm ON FRIDAY OF EACH  
# WEEK STARTING #10 (WORTH 1% OF THE GRADE)  
git add Status_${USER}.tex  
git add Status_${USER}.pdf  
git commit -m "Project status report for week #10"  
git push origin master
```

Replace 10 with the appropriate week number.



ucfirst()

Convert first letter of a string to uppercase

```
ucfirst() {  
    if [ $# != 1 ]  
    then  
        echo  
        echo " Usage: ${FUNCNAME} STRING"  
        echo " e.g.: ${FUNCNAME} 'this is a test'"  
        echo  
    else  
        local str=$(echo "$1" | sed 's/\(.*/\U\1/' )  
        echo ${str}  
    fi  
}
```

Add the function to `${HOME}/bin/functions.sh` and run source `${HOME}/.bashrc`.



ucwords()

Convert first letter in every word of a string to uppercase

```
ucwords() {  
    if [ $# != 1 ]  
    then  
        echo  
        echo " Usage: ${FUNCNAME} STRING"  
        echo " e.g.: ${FUNCNAME} 'this is a test'"  
        echo  
    else  
        local str=$(echo "$1" | sed 's/\b\(.*/\u\1/g')  
        echo ${str}  
    fi  
}
```

Add the function to `${HOME}/bin/functions.sh` and run `source ${HOME}/.bashrc`.



smart_ucwords()

A smarter version of ucwords()

```
smart_ucwords() {  
    # ucwords() functionality  
    local str=$(echo "$1" | sed 's/\b\(.*/\u\1/g')  
    ignore_list=(A An And If Into Is For Of Or)  
  
    for x in "${ignore_list[@]}"  
    do  
        lc_x=$(echo "${x}" | tr '[:upper:]' '[:lower:]')  
        str=$(echo "${str}" | sed "s/ ${x} / ${lc_x} /g")  
    done  
    echo ${str}  
}
```

Add the function to `${HOME}/bin/functions.sh` and run source `${HOME}/.bashrc`.

gcc_omp()

A compact function to compile OpenMP programs

```
gcc_omp() {
    if [ $# != 1 ]
    then
        echo
        echo " Usage: ${FUNCNAME} PROGRAM.c"
        echo " e.g.: ${FUNCNAME} HelloWorld_omp_01.c"
        echo
    else
        local cbase=$(echo "$1" | awk -F "." '{ print $1 }')
        gcc -Wall -g -lm -fopenmp $1 -o ${cbase}.x
    fi
}
```

Add this function to `${HOME}/bin/functions.sh` and run source `${HOME}/.bashrc`. One may instead write a well-commented `gcc_omp.sh` with suitable error codes that will check the existence, non-zero file size and usefulness of the C program before compiling and running it with a specified number of threads.



- * Accepts two arguments: **FILENAME** and **OMP_NUM_THREADS**
- * Checks **FILENAME** for existence, non-zero file size, and useful (i.e., non-comment and non-blank line) content
- * Checks the current directory and **BASENAME.x** for write permission
- * Compiles **FILENAME** using

```
gcc -Wall -g -lm -fopenmp FILENAME -o BASENAME.x
```

- * Runs **BASENAME.x** using **OMP_NUM_THREADS** threads

Lock file

Brainstorm

Suppose that a shell script, `summary.sh`, is designed to extract necessary information from a series of simulations and write them to `summary.txt`.

How can one ensure that a second (or n^{th}) inadvertent execution of the same script does not write to/update the `summary.txt` file while the first one is still running?

Lock file

Version #1

```
#!/bin/bash
# Useful comments

LOCKFILE="/tmp/${USER}_summary.lck"
if [ -f "${LOCKFILE}" ]
then
    echo " Lock file exists.  Exiting the script."
    exit
else
    touch ${LOCKFILE}
    # Commands to extract and summarize information (core)
    rm -f ${LOCKFILE}
fi
```

Lock file

Problem with version #1

The script works fine as long as it is not interrupted/terminated while the commands to extract and summarize information (i.e., the core) are running. If killed during the core's execution, the lock file will continue to exist.

Subsequent legitimate executions of the script won't run until the lock file has been manually deleted.

Lock file

Version #2

```
#!/bin/bash
# Useful comments

LOCKFILE="/tmp/${USER}_summary.lck"
if [ ! -e "${LOCKFILE}" ]
then
    trap "rm -f ${LOCKFILE}; exit" INT TERM EXIT
    echo $$ > ${LOCKFILE}
    # Commands to extract and summarize information (core)
    rm -f ${LOCKFILE}
else
    echo "${LOCKFILE} exists; owned by $(cat ${LOCKFILE})"
fi
```

Lock file

Problem with version #2

`trap` command ensures that the lock file is also removed when the script is interrupted/killed.

However, there is potential for a race condition between the time one tests for the lockfile and time one creates it.

Lock file

Version #3

```
#!/bin/bash
# Useful comments

LOCKFILE="/tmp/${USER}_summary.lck"
if (set -o noclobber; echo $$ > ${LOCKFILE}) 2> /dev/null
then
    trap 'rm -f "${LOCKFILE}"; exit $?' INT TERM EXIT
    # Commands to extract and summarize information (core)
    rm -f ${LOCKFILE}
    trap - INT TERM EXIT
else
    echo "${LOCKFILE} exists; owned by $(cat ${LOCKFILE})"
fi
```

Download and install Julia 0.5.0

```
OS="linux" ; ARCH="x86_64" ; VERSION="0.5.0"
JULIA="julia-${VERSION}-${OS}-${ARCH}"
mkdir -p ${HOME}/apps/julia/
cd ${HOME}/apps/julia/
tar -zvxf ${HOME}/tmp/${JULIA}.tar.gz
mv julia-* ${VERSION}
```

```
# Update ${HOME}/.bash_${USER} and source it
JULIA="${HOME}/apps/julia/0.5.0"
PATH="${PATH}:${JULIA}/bin"
LD_LIBRARY_PATH="${LD_LIBRARY_PATH}:${JULIA}/lib"
MANPATH="${MANPATH}:${JULIA}/share/man"
```

Test (Iterative solution of golden ratio)

```
# What is the output of the following command?  
# which julia
```

```
cd ${UN5390}  
git pull
```

```
cd CourseWork/Week_12/${USER}_12  
rsync -avhP ../AdditionalMaterial/Julia/ ./Julia/
```

```
cd Julia  
julia GoldenRatio.jl  
julia GoldenRatioSml.jl
```

Sample files provided by Adam Mitteer, UN5390 in Fall 2016, are included in week #12 AdditionalMaterial.

Sort an array

BASH

```
#! /bin/bash
# Useful comments

# Unsorted and sorted arrays
ARRAY_UNSORTED=( simon amy zachary carrie jill henry )
ARRAY_SORTED=( $(
    for tmp in ${ARRAY_UNSORTED[@]}
    do
        echo "${tmp}"
    done | sort) )

# Useful stuff with sorted array
```



Sort multi-column data file

Creating a multi-column data file with *random* information

N=10

```
touch file.dat
```

```
truncate -s0 file.dat
```

```
for (( i=1; i<=${N}; i++ ))
```

```
do
```

```
    j=$((expr ${i} * ${i} - ${i}))
```

```
    j=$(printf "%03d" "$j")
```

```
    TEXT=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | \
          fold -w 32 | head -n 1)
```

```
    echo "${i} ${j} ${N} ${RANDOM} ${TEXT}" >> file.dat
```

```
done
```

Refer to `man fold`, `man sort` and `man tr` for more information.



Sort multi-column data file

Commands

```
# Default; 1st field; use -r for reverse; -n for numeric
sort file.dat
sort -n file.dat

# 4th field; -k 4,4n to force numeric
sort -k 4 file.dat
sort -k 4,4n file.dat

# 3rd field (numeric), then 5th field (to break ties)
sort -k 3,3n -k 5 file.dat

# 3rd field (n), then 5th field, and then 4th field (n)
sort -k 3,3n -k 5 -k 4,4n file.dat
```



Opportunities

They do knock every once in a while



<http://dilbert.com/strip/2009-09-24/>

IT-managed Linux labs

- * `colossus.it.mtu.edu` and `guardian.it.mtu.edu`
 - * Intel Xeon X5675 3.07 GHz, 24 CPU cores, 96 GB RAM
 - * Accessible for all from anywhere via SSH using a Terminal
 - * Appropriate for light- to medium-weight computations
- * Linux workstation in a campus lab/office
 - * May not be as powerful as `colossus.it` or `guardian.it`
 - * May not be directly accessible from off-campus
 - * <https://www.it.mtu.edu/computer-labs.php>

All IT-managed workstations in Linux labs run RHEL 7.x and will mount the campus home directory.



Network of Expertise

UN5390; CRN: 84758

#	Name	Email	Dept/Program	Advisor
01	Adam Mitteer	aamittee	Data Science	Mari Buche
02	Ashley Kern	ankern	Data Science	Mari Buche
03	Eassa Hedayati	hedayati	Physics	John Jaszcak
04	Hashim Mahmud	hnalmahm	ME-EM	Gregory Odegard
05	Jeffrey Brookins *	jmbrooki	MSE	Jaroslaw Drellich
06	Paul Roehm	pmroehm	ME-EM	Gregory Odegard
07	Qing Guo	qinguo	Physics	Ravindra Pandey
08	Subin Thomas	subint	Physics	Raymond Shaw

* Undergraduate students



Network of Expertise

BE5390: Biomedical Engineering CRN: 84759

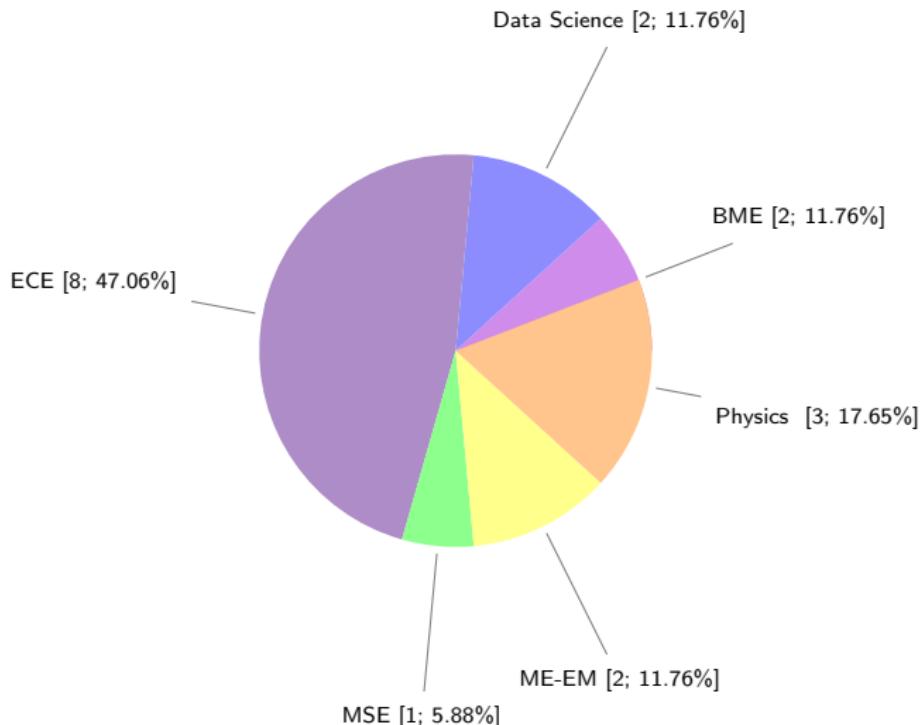
#	Name	Email	Advisor
09	Cal Riutta *	cdriutta	Jinfeng Jiang

EE5390: Electrical and Computer Engineering; CRN: 84760

10	Akhil Kurup	amkurup	Michael Roggemann
11	Avinaash Kovvuri	askovvur	Michael Roggemann
12	Ian Cummings	itcummin	Timothy Havens
13	Prithvi Kambhampati	pkambham	Michael Roggemann
14	Sandeep Lanka	slanka	Michael Roggemann
15	Sameer Saraf	svsaraf	Michael Roggemann
16	Shuo Wang	wshuo	Jeremy Bos
17	Zhiqiang Zhao	qzzhao	Zhuo Feng

* Undergraduate students

Network of Expertise



17 registered students.

NSF Graduate Research Fellowship Program 2017

- * Applicant must be a US citizen or a permanent resident
- * Fellowship supports 3 years of study
 - \$34k of stipend per year +
 - \$12k of cost-of-education allowance to the university per year
- * MS and PhD candidates in STEM and STEM education
 - Must be in first two years of graduate study
 - Senior undergraduates are also encouraged to apply
- * Michigan Tech Information Session
 - 5 pm, 7th September 2016 (Wednesday), Admin 404



CareerFEST and Career Fair

- * More details at <http://www.mtu.edu/career/careerfest/>
- * Create/Update your two-page résumé
- * Have it critiqued by Michigan Tech Career Services
- * Develop the habit of reviewing/updating it once per month
- * Use the \LaTeX template in [\\$\{UN5390\}/\text{LaTeXTemplates}/\text{Resume}/\\$](#)
- * Additional resources
 - <http://www.mtu.edu/career/students/toolbox/resumes/examples/>
 - <http://owl.english.purdue.edu/owl/resource/719/1/>
 - <http://www.sharelatex.com/templates/cv-or-resume>
 - <http://www.latextemplates.com/cat/curricula-vitae>

CareerFEST is a collection of many different informal events that take place during the month of Career Fair.



- * Commonly used Linux commands
- * Extensive shell scripting
- * Revision control (Git)
- * Workflow development
- * Statistical analysis (Python, R and Gnuplot)
- * Visualization (Python, R and Gnuplot)
- * White papers and internal publications (\LaTeX)



- * Commonly used Linux commands
- * Extensive shell scripting
- * Revision control (Git/Subversion)
- * Workflow development
- * Domain-specific expertise
- * Modeling, simulation, analysis and visualization
 - Choice of language/toolset depends on a project
- * White papers, internal and external publications (\LaTeX)



Keweenaw Climate Science Event

#1 of four-part event

The Orpheum Theater

6 – 8 pm on Thursday, 8th September 2016

Subsequent events

6th October 2016

3rd November 2016

1st December 2016

No admission fee

Free pizza and soft drinks

[More information](#)

Organized by [Keweenaw Climate Community](#), and sponsored by the local chapter of the [American Chemical Society](#) and the [Department of Social Sciences](#) at Michigan Tech.



Keweenaw Climate Science Event

#2 of four-part event

The Orpheum Theater

6 – 8 pm on Thursday, 6th October 2016

Subsequent events

3rd November 2016

1st December 2016

No admission fee

Free pizza and soft drinks

[More information](#)

Organized by [Keweenaw Climate Community](#), and sponsored by the local chapter of the [American Chemical Society](#) and the [Department of Social Sciences](#) at Michigan Tech.



ICC Distinguished Lecture

CS For All: Considering The Implications Of 'For All'

Dr. Kamau Bobb

Research Scientist, Georgia Tech

Program Officer, CISE, NSF



4th October 2016 1 pm, ME-EM 406

<https://www.ceismc.gatech.edu/about/staffdirectory/kamau-bobb>

- * Applicant must be a US citizen
- * Open to junior and senior undergraduates
- * Paid 10-week research internship in Los Alamos, NM
- * Introduction to techniques and practices of cluster computing via
Lectures (seminars from HPC practitioners and researchers) +
Laboratory (hands-on setup, configuration, administration, testing,
monitoring, scheduling, etc. of supercomputer clusters) +
Technical broadening (research project) +
Professional development (resume and poster development, verbal and
written communication, technical project execution) +
Tour of the Metropolis Supercomputing Center



Los Alamos National Laboratory | Los Alamos National Laboratory on GitHub
Computer System, Cluster and Networking Summer Institute | Application deadline: 01 December 2016



- * Applicant must be a US citizen
- * Stipend available (up to \$650/week)
- * Travel to and from appointment site
- * Open to undergraduate and graduate students, and post-graduates
- * Opportunities to learn from top scientists and subject matter experts, and career possibilities
- * Accounting and Finance, Business, Communications, Computer Science and Information Technology, Engineering, Environmental Sciences, Law, Physical and Mathematical Sciences, Policy, Program Management, Safety and Health, and other related areas



US Department of Energy is the nation's leading sponsor for scientific research.

US Department of Energy Scholars Program | Application deadline: 15th December 2016

US DOE Computational Science Graduate Fellowship 2017

- * Applicant must be a US citizen
- * Open to senior undergraduates and first year graduate students (MS/PhD candidates without MS)
- * Fellowship renewable up to 4 years
 - \$36k of stipend per year +
 - Full tuition and other fees +
 - \$5k academic allowance in first year +
 - \$1k academic allowance for each renewed year
- * Opportunity to attend annual program review
- * 12-week research practicum



US Department of Energy is the nation's leading sponsor for scientific research.
US DOE Computational Science Graduate Fellowship | Application deadline: 18th January 2017

US DHS HS-STEM Summer Internship 2017

- * Applicant must be a US citizen
- * Open to undergraduates and graduate students majoring in homeland security disciplines
- * 10-week program with stipend
\$6k for undergraduates / \$7k graduate students + Limited travel expenses
- * Exposure to and participation in research in DHS mission-relevant research areas
- * Networking (w/ scientists & laboratories) and career opportunities



Oak Ridge Institute for Science and Education (ORISE) administers this internship program through an interagency agreement between the US Department of Energy and the US Department of Homeland Security. ORISE is managed by Oak Ridge Associated Universities (ORAU) for US DOE.
US DHS HS-STEM Summer Internship Program | [@GovCareerPaths](#) | Application deadline: 7th December 2016

Keweenaw Climate Science Event

#3 of four-part event

The Orpheum Theater

6 – 8 pm on Thursday, 3rd November 2016



Subsequent event

1st December 2016

No admission fee

Free pizza and soft drinks

[More information](#)

Robert Handler (1982 – present): Adj. Asst. Professor in Civil and Environmental Engineering and Operations Manager for Sustainable Futures Institute, Michigan Tech; PhD in Environmental Engineering from The University of Iowa (2009)

Stephen Handler (1982 – present): Climate Change Specialist in US Forest Service and Northern Institute of Applied Climate Science (NIACS); MS in Resource Conservation and International Conservation and Development from University of Montana (2007)

Organized by Keweenaw Climate Community, and sponsored by the local chapter of the American Chemical Society and the Department of Social Sciences at Michigan Tech.

41 North Film Festival



Destination Cinema at Michigan Tech
ROZSA CENTER FOR THE PERFORMING ARTS
NOVEMBER 3-6, 2016

Predator/Prey: The Fight For Isle Royale Wolves

Saturday, 5th November 2016, 7:30 pm
Open and free to public

<http://hdmzweb.hu.mtu.edu/41north/2016/> | <https://www.facebook.com/41northfilmfest/>

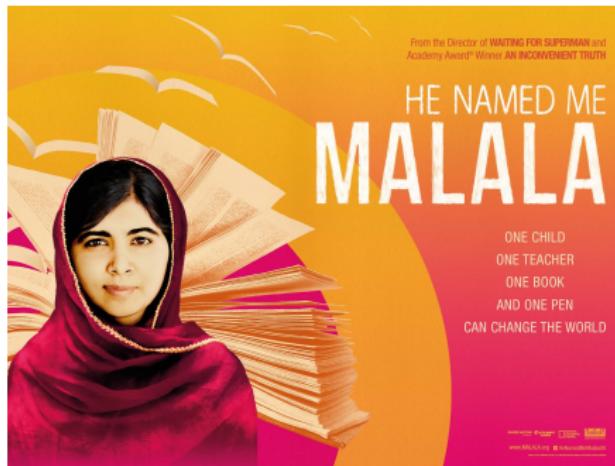


Before The Flood



Limited free screening on [iTunes](#), [National Geographic](#) and [YouTube](#)

International Education Week



Friday, 11th November 2016, 5:30 pm, Fisher 135
Free admission

Malala Yousafzai (1997 – present): Pakistani activist for female education, and a recipient of the 2014 Nobel Peace Prize.
Sponsored by Michigan Tech Provost Office, International Programs and Services, and Michigan Tech Film Board.

Physics Colloquium

HPC and Simulation

The Third Pillar of Contemporary Science

Dr. Siegfried Hoefinger

Vienna University of Technology



10 am on Friday, 18th November 2016

Fisher Hall 133

<https://www.linkedin.com/in/siegfried-hoefinger-a512541>

NVIDIA Graduate Fellowship Program

2017-18

- * Open to 2nd year PhD students
Computer Science/Engineering, Systems Architecture,
Electrical Engineering or a related area
- * Unrestricted funds to further student research
Up to \$50,000 towards tuition, books, living stipend, salary and travel to
NVIDIA Research Summit, SIGGRAPH, or relevant conferences
- * Re-applicable in subsequent years



NVIDIA | NVIDIA Graduate Fellowship Program

Application deadline: 16th January 2017

Introduction to Python for CS&E (PDF)

Introduction to Python for Computational Science and Engineering A Beginner's Guide

Hans Fangohr
Professor, Computational Modeling
University of Southampton, UK

Free PDF, and associated materials



Hans Fangohr (1973 – present): German computational scientist (@ProfCompMod)
<http://www.southampton.ac.uk/~fangohr/>

Keweenaw Climate Science Event

#4 of four-part event

The Orpheum Theater

6 – 8 pm on Thursday, 1st December 2016

No admission fee

Free pizza and soft drinks

[More information](#)

Organized by [Keweenaw Climate Community](#), and sponsored by the local chapter of the [American Chemical Society](#) and the [Department of Social Sciences](#) at Michigan Tech.



Memorial Union Building (MUB)

Ballroom B, 4 – 5:30 pm on Thursday, 1st December 2016

4:00 – 4:30 pm – Mingling and refreshments

4:30 – 5:00 pm – 2-minute presentations

5:00 – 5:30 pm – Discussions, if interested

Speakers and abstracts

Dress rehearsal for upcoming Research Marketing III

TechTalks are rapid-paced samplings of work from a number of faculty via a two slides in two-minute format. TechTalks showcase both published and unpublished work. These are a part of the Michigan Tech Research Forum – a university presentation series showcasing the work of Michigan Tech faculty, postdocs, and researchers to strengthen our community.



Mathematical Results

Standing the test of time

Mathematics, rightly viewed, possesses not only truth, but supreme beauty – a beauty cold and austere, like that of sculpture, without appeal to any part of our weaker nature, without the gorgeous trappings of painting or music, yet sublimely pure, and capable of a stern perfection such as only the greatest art can show.

– Bertrand Russell, A History of Western Philosophy (1945)



Bertrand Arthur William Russell (1872 – 1970): British philosopher, logician, mathematician, historian, writer, social critic, and political activist. 1950 Nobel Laureate in Literature.

Fundamental Theorem of Algebra

Every non-constant single-variable polynomial with complex coefficients has at least one complex root. Since real numbers are a subset of complex numbers, the result/statement extends to polynomials with real coefficients as well.

Alternate statement #1 (proved using successive polynomial division)

Every non-zero, single-variable, degree n polynomial with complex coefficients has, counted with multiplicity/degeneracy, exactly n roots.

Alternate statement #2

The field of complex numbers is algebraically closed.

Theorem first proven algebraically by James Wood (with missing steps) in 1798, and geometrically by Johann Carl Friedrich Gauss (with a topological gap) in 1799.



Fundamental Theorem of Calculus

Suppose that $f(x)$ is defined and continuous on $[a, b]$. Suppose that $y(x)$ is an anti-derivative of $f(x)$. Then

$$\int_a^b f(x) dx = y(b) - y(a)$$

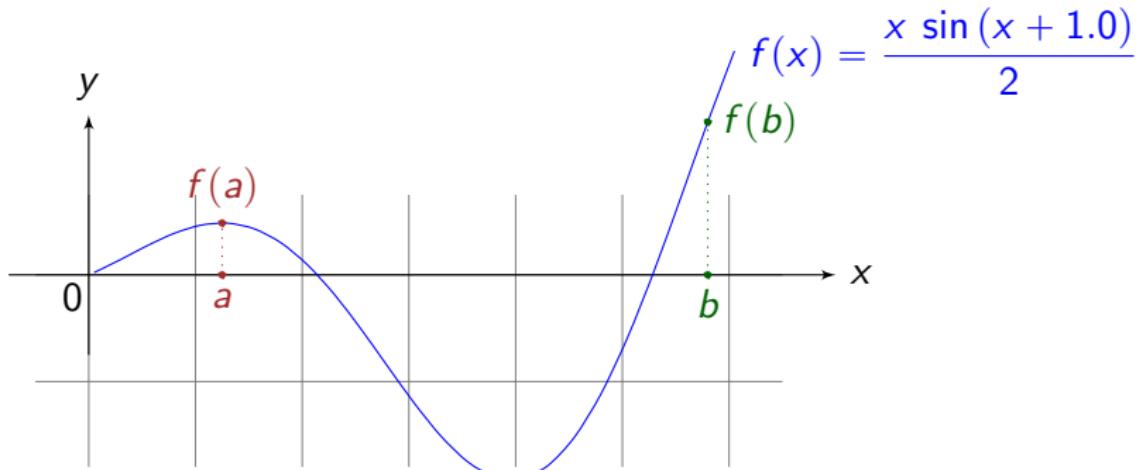
Changing the notations while retaining the underlying essence,

$$\int_{t_n}^{t_{n+1}} f(y, t) dt = y_{n+1} - y_n$$

Re-arranging the terms,

$$y_{n+1} = \boxed{y_n} + \boxed{\int_{t_n}^{t_{n+1}} f(y, t) dt}$$

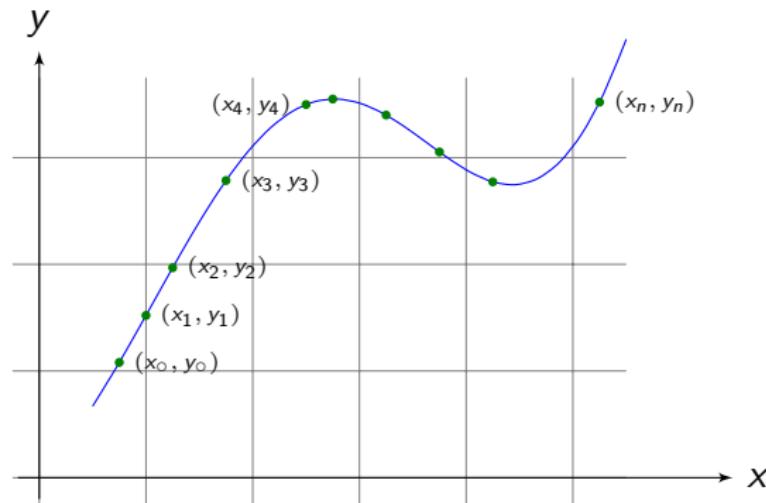
Intermediate Value Theorem (IVT)



For any function $f(x)$ that is continuous on $[a, b]$, and has values $f(a)$ and $f(b)$ at a and b respectively, then $f(x)$ also takes any value between $f(a)$ and $f(b)$ at some point within the interval.

Lagrange Polynomial Interpolation

Suppose that (x_i, y_i) , with $i = 0 : 1 : n$, are a set of $n + 1$ unique points



Joseph-Louis Lagrange (1736 – 1813): Italian mathematician and astronomer
[Interpolating Polynomials](#), L. Shure, MathWorks
[Lagrange Interpolating Polynomial](#), B. Archer, Wolfram

Lagrange Polynomial Interpolation

The general form of Lagrange interpolating polynomial, one that passes through $n + 1$ points

$$\mathcal{L}_n(x) = \sum_{i=0}^n l_i(x) y_i$$

Lagrange basis polynomials are given by

$$l_i(x) = \prod_{\substack{m=0 \\ m \neq i}}^n \frac{x - x_m}{x_i - x_m}$$

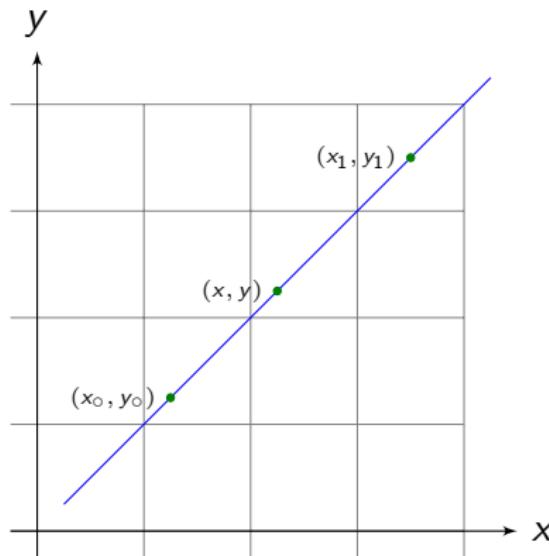
and are built to have the *Kronecker delta* property

$$l_i(x_j) = \delta_{ij}$$

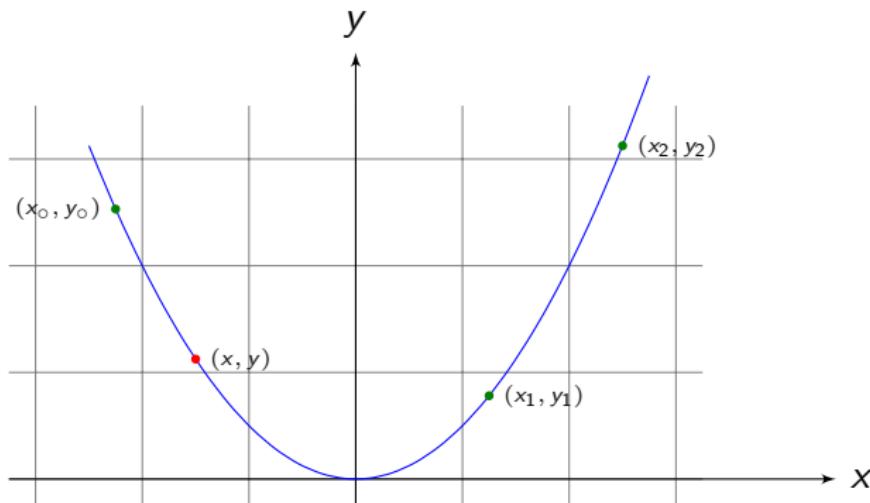
Lagrange Polynomial Interpolation

Linear

Suppose that (x_0, y_0) and (x_1, y_1) are two known points. The linear interpolant is then a straight line between these two points.



Lagrange Polynomial Interpolation Quadratic



$$L_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} y_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} y_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} y_2$$

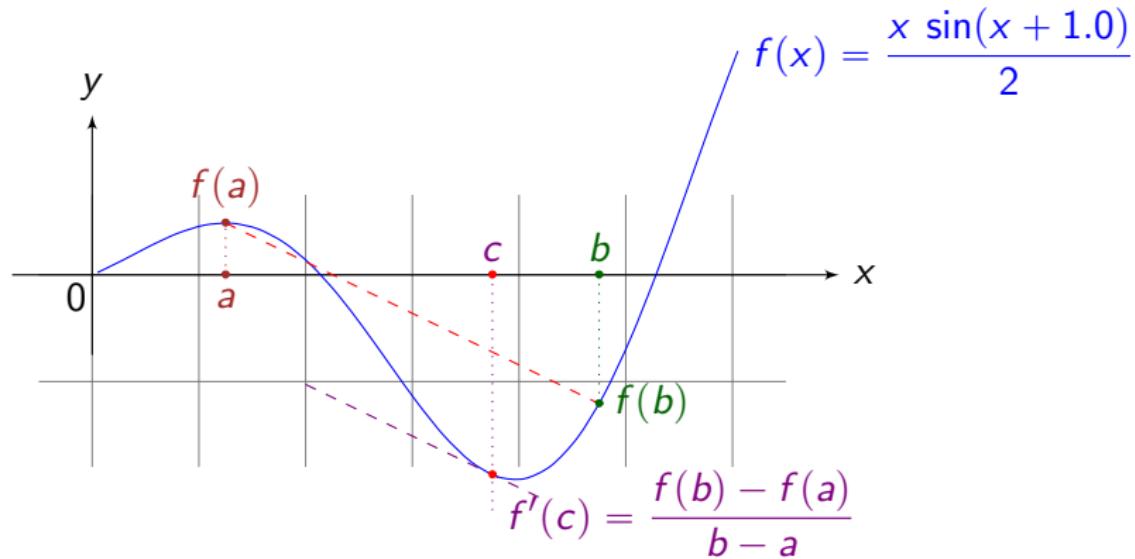
Lagrange Polynomial Interpolation

Error analysis

If $f(x)$ is $n + 1$ times continuously differentiable on a closed interval $[a, b]$, and $p_n(x)$ is a polynomial of degree at most n that interpolates $f(x)$ at $n + 1$ distinct points x_i , ($i = 0, 1, 2, \dots, n$) in that interval. Then

$$\epsilon_n = \int_a^b [f(x) - p_n(x)] dx = \int_a^b \frac{f^{(n+1)}}{(n+1)!} \prod_{i=0}^n (x - x_i) dx$$

Mean Value Theorem



For any function that is continuous on $[a, b]$ and differentiable on (a, b) , there exists a point c in (a, b) such that the line joining $f(a)$ and $f(b)$ (i.e., the secant) is parallel to the tangent at c .



Weighted Mean Value Theorem for Integrals

Suppose that $f(x)$ and $g(x)$ are continuous on $[a, b]$. If $g(x)$ never changes sign and is positive, $g(x) \geq 0$, in $[a, b]$, then for some c in $[a, b]$

$$\int_a^b f(x) g(x) dx = f(c) \int_a^b g(x) dx$$

Newton-Cotes Formula

Suppose that $f(x)$ is defined and continuous on $[a, b]$.

Consider the integral

$$I = \int_a^b f(x) dx$$

If $f(x)$ can be approximated by an n^{th} order polynomial

$$p_n(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots + \alpha_{n-1} x^{n-1} + \alpha_n x^n$$

then the integral, I , takes the form

$$I = \int_a^b [\alpha_0 + \alpha_1 x + \alpha_2 x^2 + \dots + \alpha_{n-1} x^{n-1} + \alpha_n x^n] dx$$



Isaac Newton (1642 – 1727): English physicist and mathematician

Roger Cotes (1682 – 1716): English mathematician (no photo)

Taylor Series Expansion

If $f(x)$ is infinitely differentiable at x_0 , then

$$f(x) = \sum_{n=0}^{\infty} \frac{(x - x_0)^n}{n!} \left. \frac{d^n}{dx^n} f(x) \right|_{x=x_0}$$



A more general form that clearly identifies the error term is given by the p^{th} order Taylor series expansion of $f(x)$ with $\tilde{x} \in [x, x + \Delta x]$

$$f(x + \Delta x) = \sum_{n=0}^p \frac{(\Delta x)^n}{n!} \frac{d^n}{dx^n} f(x) + \frac{(\Delta x)^{p+1}}{(p+1)!} \frac{d^{p+1}}{dx^{p+1}} f(\tilde{x})$$

Brook Taylor (1685 – 1731): English mathematician

Random Variables and Distributions

The need

Random variables and their distributions provide a basis for developing probabilistic models and describing the behavior of important characteristics of interest (i.e., real data).

Y is a random variable if it is a function that assigns a real numbered value to every possible event in a sample space of interest. Since every possible set of values for a random variable Y corresponds to some event, it has a probability associated with it. A random variable's distribution details the probabilities associated with these sets of values in a meaningful way.

It is a common practice to use an uppercase alphabet to denote the random variable, and the corresponding lowercase alphabet to denote a specific value of this variable. A discrete random variable can assume at most a countable number of values. A continuous random variable can assume an uncountable number of values.

Random Variables and Distributions

PDF and CDF

The probability distribution function (PDF) of some random variable Y is given below. $P(Y = y_i)$ indicates the probability of the random variable Y taking on a given value, y_i . $F(y_i)$ represents the cumulative distribution function (CDF), and is used to model the behavior of Y .

y_i	$P(Y = y_i)$	$F(y) = P(Y \leq y_i)$
0	0.10	0.10
1	0.30	0.40
2	0.40	0.80
3	0.20	1.00

All random variables must have a cumulative distribution function.

Uniform Distribution

Discrete and continuous

Applicable when

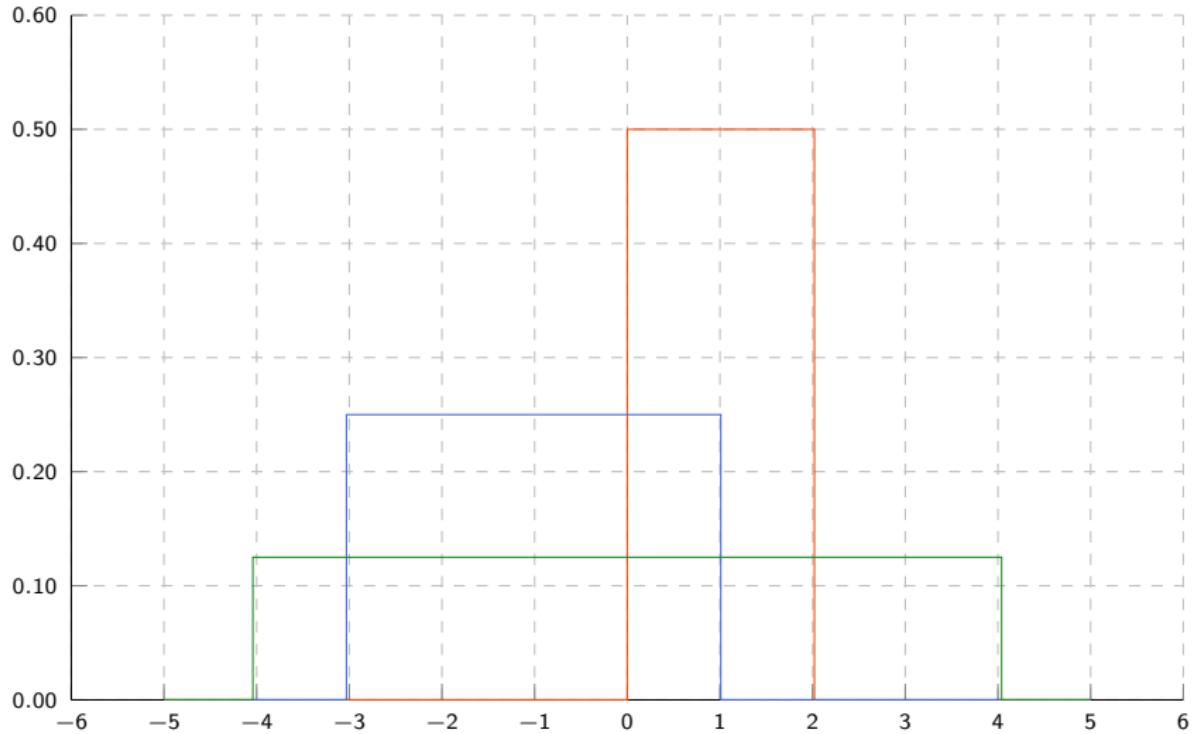
a finite number of values are equally likely to be observed. The probability density function on the interval $[a, b]$ is

$$f(x) = \begin{cases} 0 & x < a \\ 1/(b-a) & a \leq x \leq b, \text{ and } -\infty < a < b < \infty \\ 0 & x > b \end{cases}$$

Common example(s)

Throwing a fair die with possible values of 1, 2, ..., 6; each face of the die has a probability of 1/6.

Uniform Distribution



Bernoulli Distribution

Discrete

Applicable when

a random variable takes the value one with success probability of p and the value zero with a failure probability of $1 - p$. Bernoulli distribution is a special case of the Binomial distribution for $n = 1$.

Common example(s)

A coin toss where one and zero could be represented by *head* and *tail* respectively. For a fair coin, $p = 0.50$.

Binomial Distribution

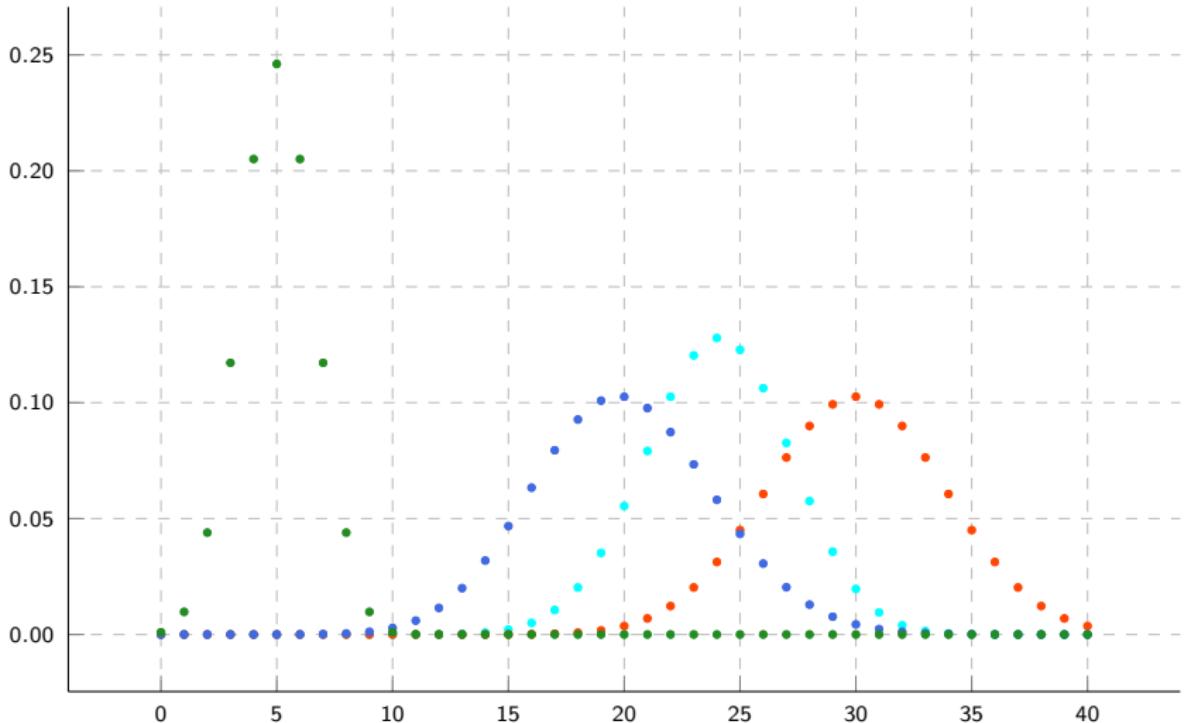
Discrete

Applicable when

a number of successes (e.g., a head or a tail) results in a sequence of n independent success/failure-type experiments, each of which yields success with a probability (or fairness factor) p . The probability of getting exactly x successes in n trials for a specified fairness value, p , is

$$P = \frac{n!}{x! (n-x)!} p^x (1-p)^{n-x}$$

Binomial Distribution



Poisson Distribution

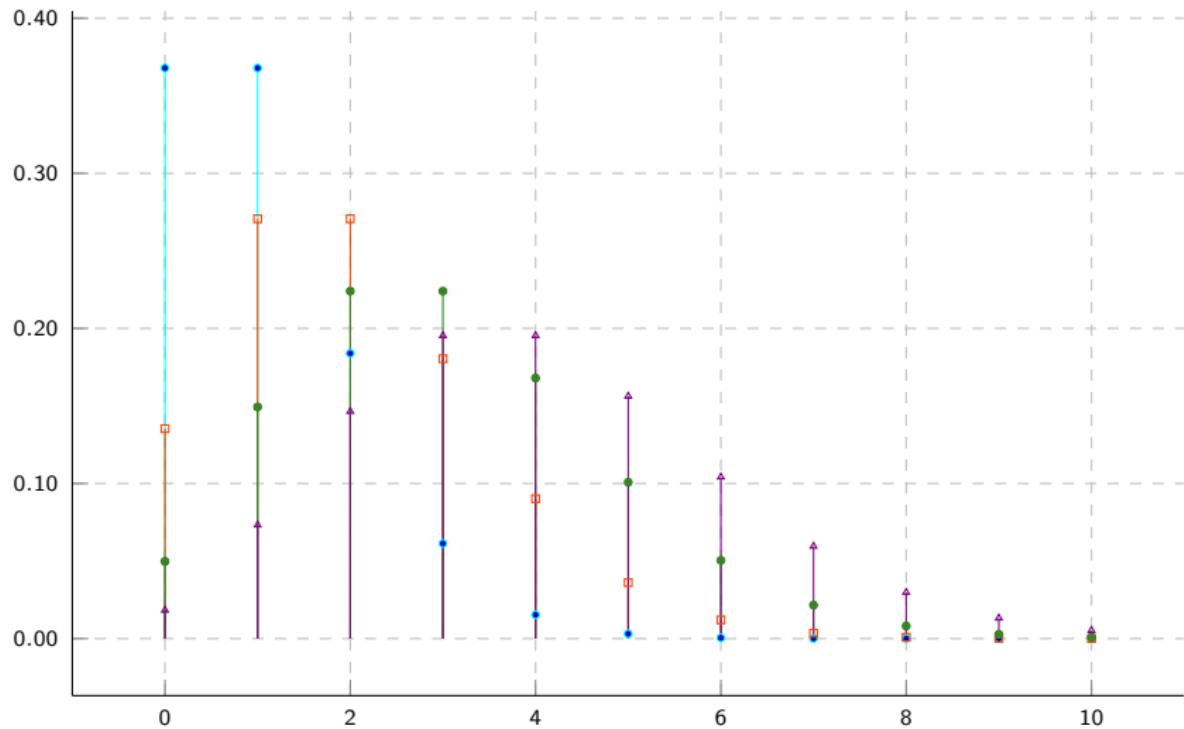
Discrete

Applicable when

a given number of events occur in a fixed interval of time if these events occur with a known average rate, independently of time since the last event, and two of them cannot occur at the same time. The probability of observing m events in an interval with the average number of events in an interval designated by λ is

$$P(m) = \frac{\lambda^m e^{-\lambda}}{m!}$$

Poisson Distribution



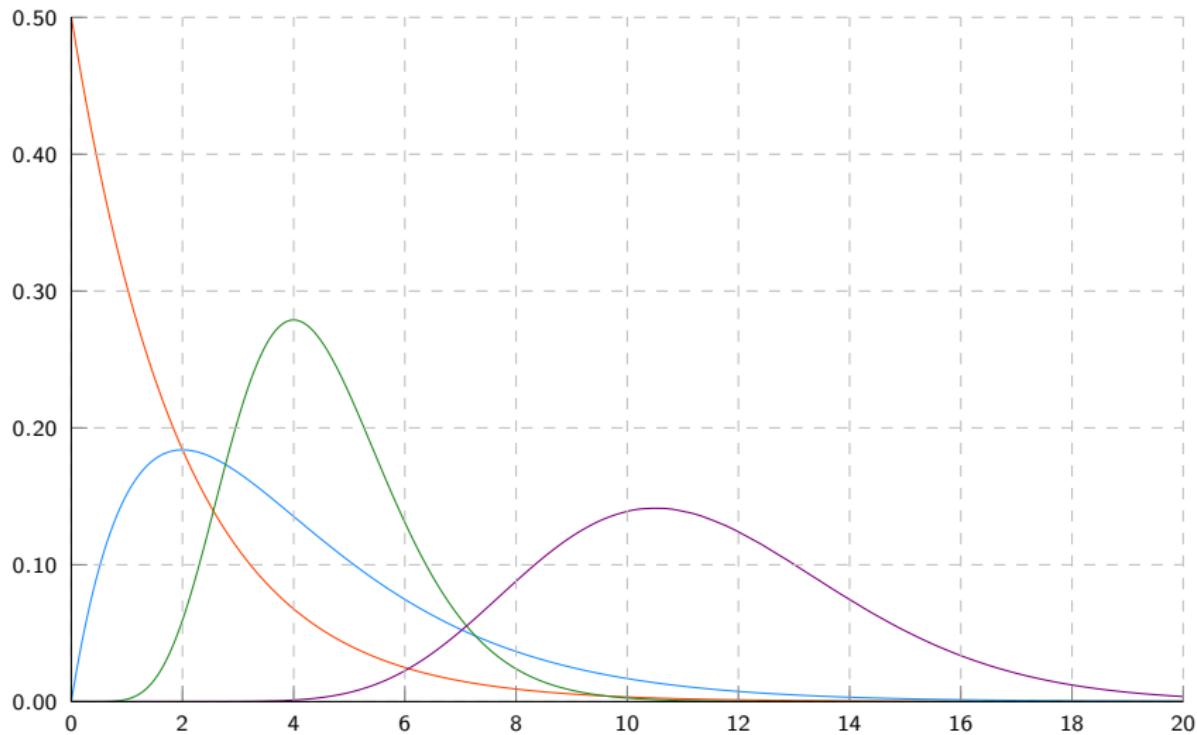
Gamma Distribution

Applicable when

the waiting times between Poisson distributed events are relevant. The probability density function with shape parameter α and scale parameter β (inverse of rate parameter) is

$$f(x) = \frac{1}{\Gamma(\alpha) \beta^\alpha} x^{\alpha-1} \exp\left(-\frac{x}{\beta}\right) \quad x \geq 0, \text{ and } \alpha, \beta > 0$$

Gamma Distribution



Normal/Gaussian Distribution

Continuous

Applicable as

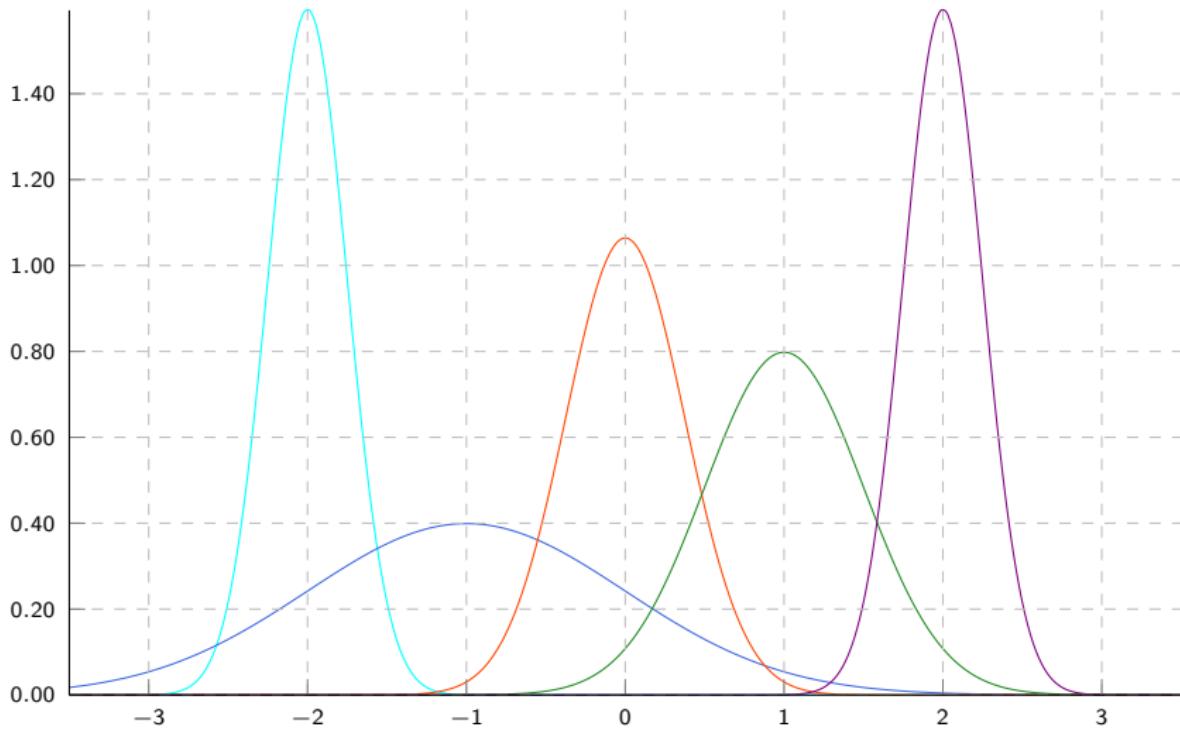
a limiting form of binomial distribution (De Moivre, 1733) and as a plausible distribution for measurement errors (Gauss, 1809). The probability density with mean μ and standard deviation σ is

$$f(x) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left[-\frac{(x - \mu)^2}{2\sigma^2}\right] \quad -\infty < x, \mu < \infty, \text{ and } \sigma > 0$$

Central limit theorem (Laplace)

Under very general conditions when n random variables, whatever their distributions, are added together, the distribution of the sum tends towards the normal (i.e., bell shape) as n increases.

Normal/Gaussian Distribution



Butcher Tableau

The general form of recursive relation for s -stage Runge-Kutta (RK) method



$$y_{n+1} = y_n + h \sum_{i=1}^s b_i k_i$$

$$k_1 = f(y_n, t_n)$$

$$k_2 = f(y_n + a_{21} k_1, t_n + c_2 h)$$

$$k_3 = f(y_n + a_{31} k_1 + a_{32} k_2, t_n + c_3 h)$$

$$k_s = f(y_n + a_{s1} k_1 + a_{s2} k_2 + \dots + a_{s,s-1} k_{s-1}, t_n + c_s h)$$

John Charles Butcher (1933 – present): New Zealand mathematician

Butcher Tableau

The choice of s (an integer), a_{ij} (the coefficients of $s \times s$ RK matrix), b_i ($i = 1, 2, \dots, s$; the weights), c_i ($i = 2, \dots, s$; the nodes), and relationship between a_{ij} and c_i uniquely identifies the s -stage RK method and ensures its consistency.

0	a_{11}	a_{12}	\dots	$a_{1,s-1}$	$a_{1,s}$
c_2	a_{21}	a_{22}	\dots	$a_{2,s-1}$	$a_{2,s}$
\vdots	\vdots	\vdots	\ddots	\vdots	\vdots
c_s	a_{s1}	a_{s2}	\cdots	$a_{s,s-1}$	$a_{s,s}$
	b_1	b_2	\cdots	b_{s-1}	b_s

$$\sum_{j=1}^{i-1} a_{ij} = c_i \quad i = 2, 3, \dots, s$$

Butcher tableau

Explicit RK2 method

0	0	0
1	1	0
	1/2	1/2

$[a_{ij}]$ needs to be a lower triangular matrix for explicit methods (i.e., $1 \leq j < i \leq s$). Drop the explicit mention of zeros

0		
1	1	
	1/2	1/2

Consistency check is satisfied

$$\sum_{j=1}^{i-1} a_{ij} = c_i \quad i = 2 \quad \Rightarrow \quad a_{21} = c_2$$



Recursive expression for RK2 (i.e., improved Euler) method

$$y_{n+1} = y_n + h \sum_{i=1}^2 b_i k_i$$

$$y_{n+1} = y_n + h b_1 k_1 + h b_2 k_2$$

Use b_i from the Butcher tableau and simplify

$$y_{n+1} = y_n + \frac{h}{2} (k_1 + k_2)$$

Butcher tableau

Explicit RK4 method

0	0	0	0	0
1/2	1/2	0	0	0
1/2	0	1/2	0	0
1	0	0	1	0
	1/6	1/3	1/3	1/6

Drop the explicit mention of zeros along the diagonal and above

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

Recursive expression for RK4 method

$$y_{n+1} = y_n + h \sum_{i=1}^4 b_i k_i$$

$$y_{n+1} = y_n + h b_1 k_1 + h b_2 k_2 + h b_3 k_3 + h b_4 k_4$$

Use b_i from the Butcher tableau

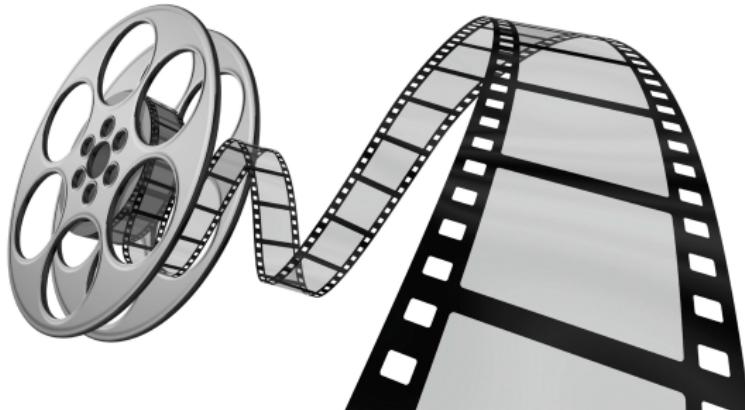
$$y_{n+1} = y_n + \frac{h}{6} k_1 + \frac{h}{3} k_2 + \frac{h}{3} k_3 + \frac{h}{6} k_4$$

Simplify

$$y_{n+1} = y_n + \frac{h}{6} (k_1 + 2k_2 + 2k_3 + k_4)$$

Videos

If a picture is worth a thousand words ...

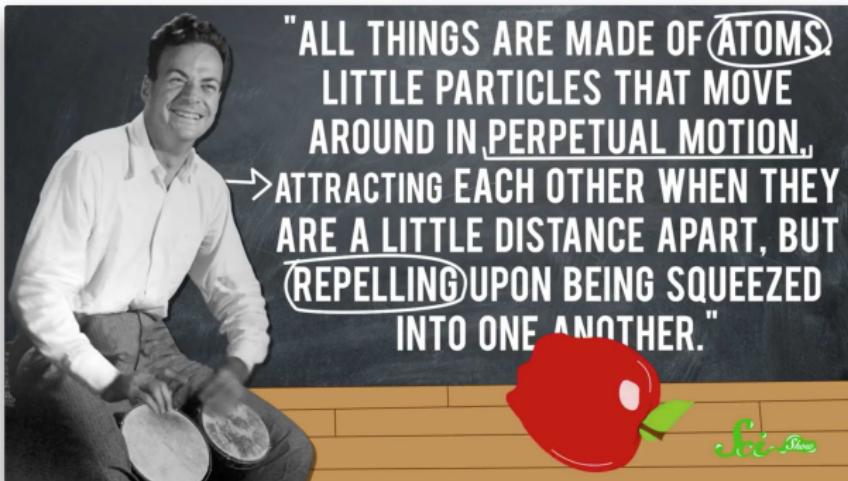


People and Personalities

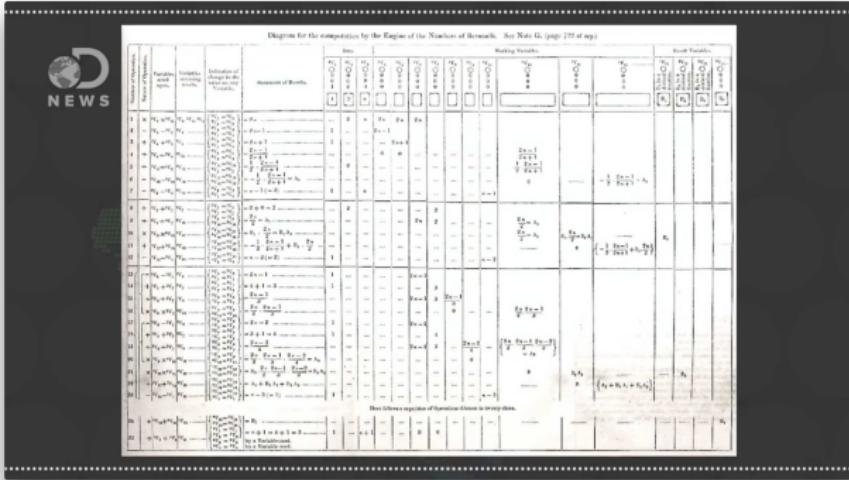


and their stories

Richard Phillips Feynman 1918 – 1988



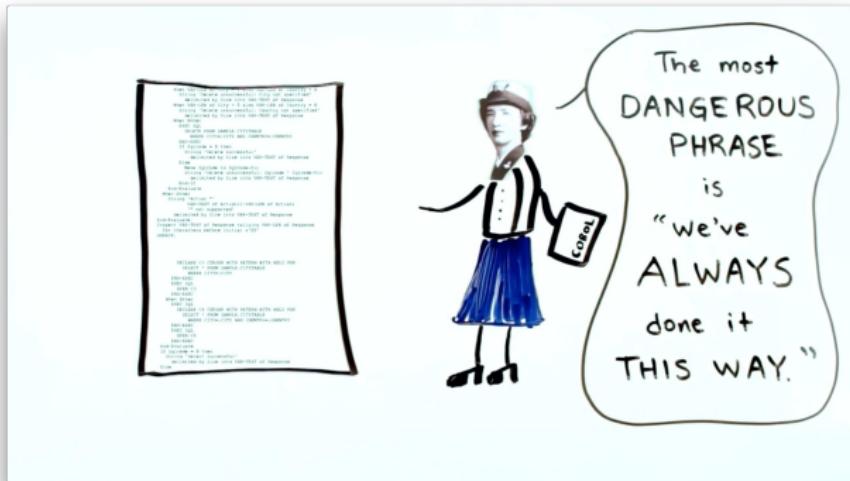
Ada August King, Countess of Lovelace 1815 – 1852



Alan Mathison Turing 1912 – 1954



Grace Brewster Murray Hopper 1906 – 1992



Margaret Heafield Hamilton 1936 – present

SHE BECAME
THE HEAD OF THE
APOLLO FLIGHT
SOFTWARE
DEVELOPMENT
TEAM

Credit: NASA



Sci

Computer History Museum

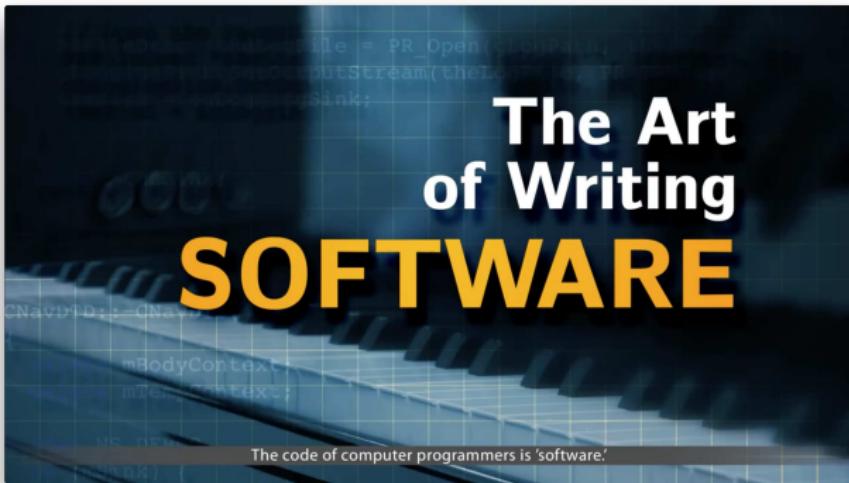


1401 N. Shoreline Blvd., Mountain View, CA 94043
(650) 810-1010

The Fairchild Notes



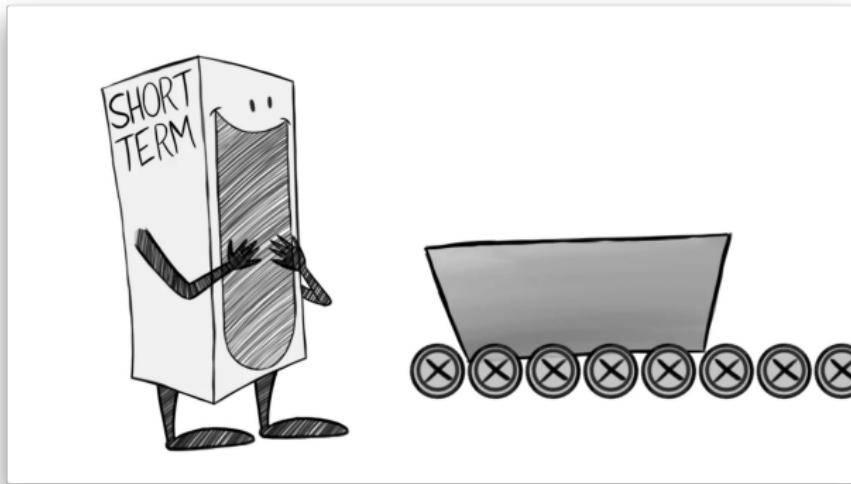
The Art of Writing Software



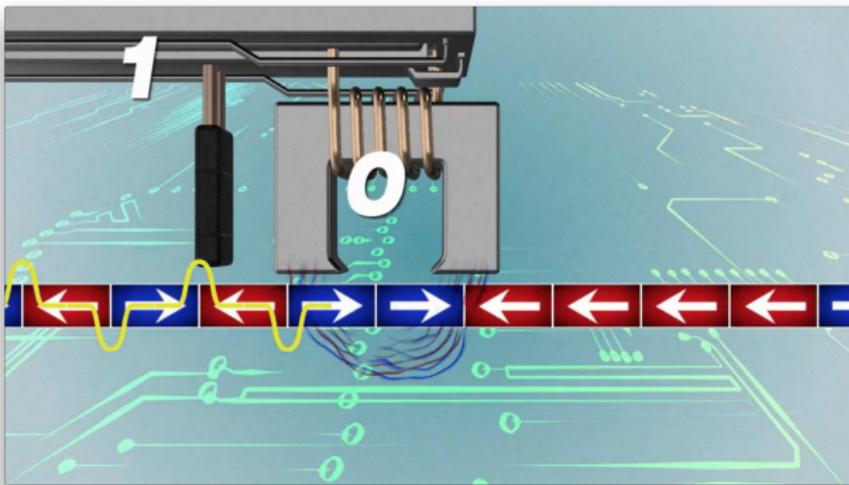
TED Ed

LESSONS WORTH SHARING

Computer Memory



Hard Drives



Turing Test



Algorithm

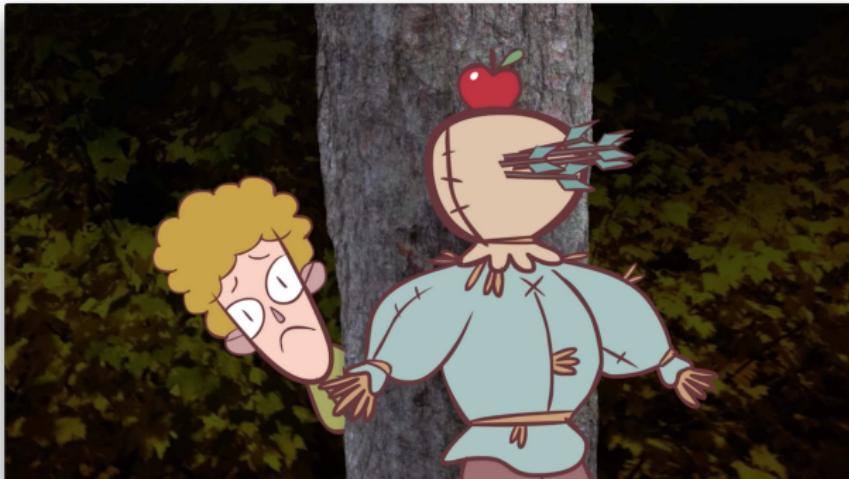
Pseudocode

let **N** = 0

For each person in room

Set **N** = **N** + 1

Accuracy vs Precision



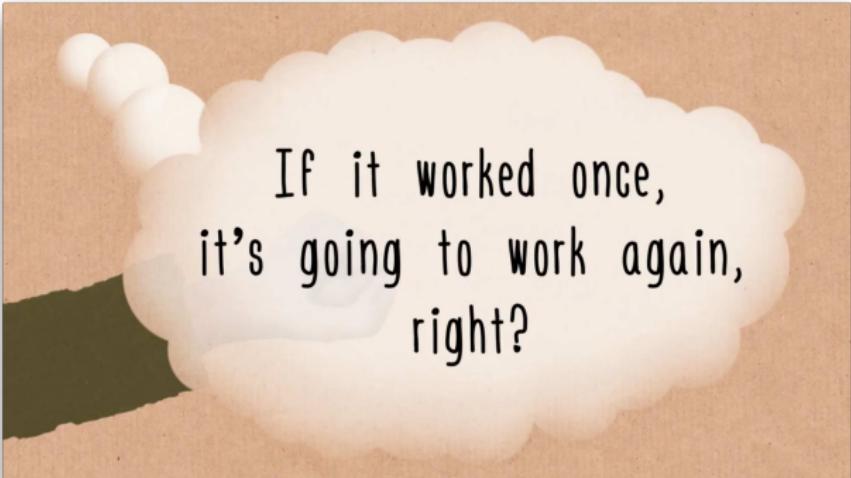
Big Data



Graph Theory

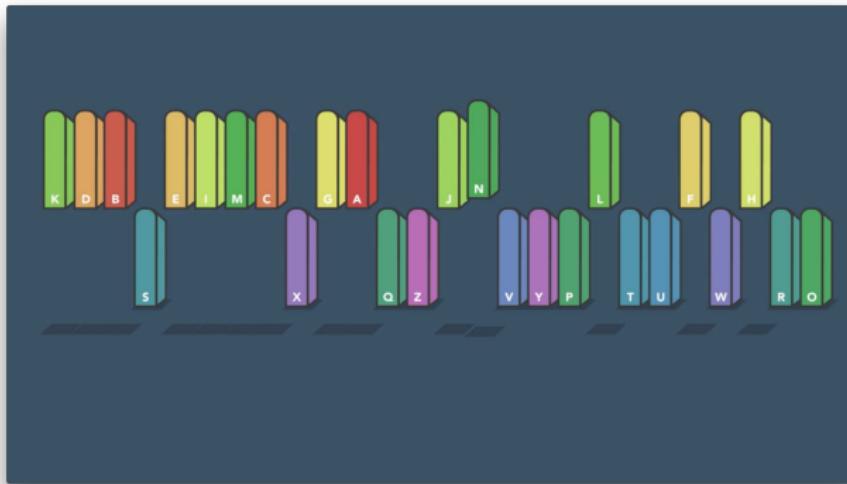


Game Theory



If it worked once,
it's going to work again,
right?

Sorting Books

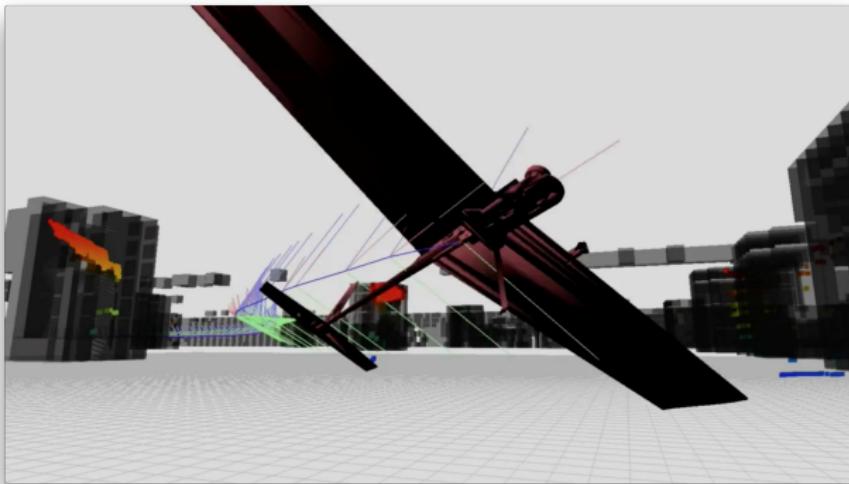




The International Conference for High Performance Computing,
Networking, Storage and Analysis

What is HPC?

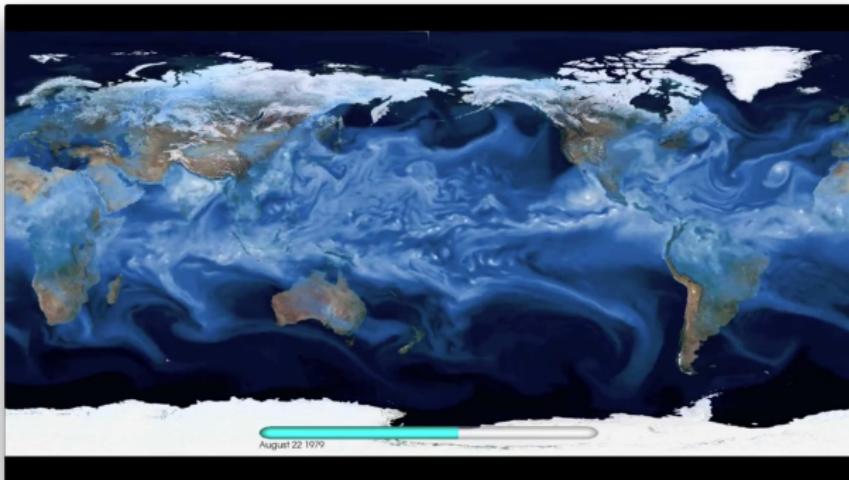
Aerospace



Batteries



Climate Modeling



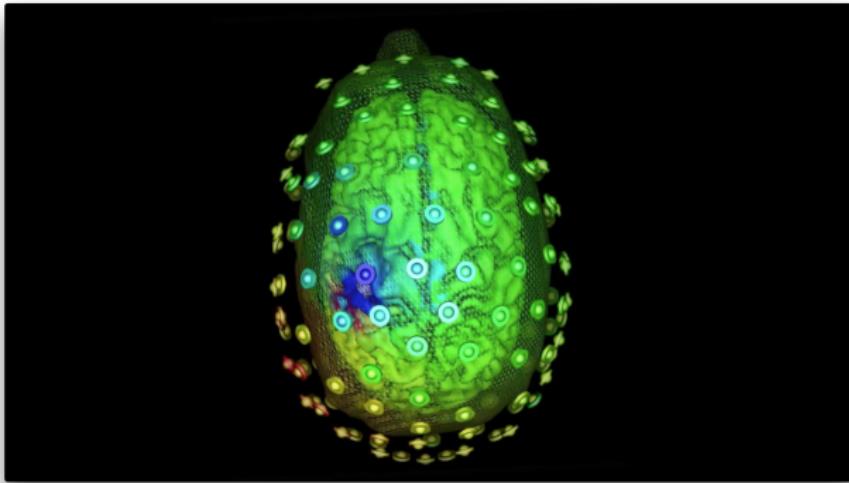
Diapers, Detergents, Shampoo



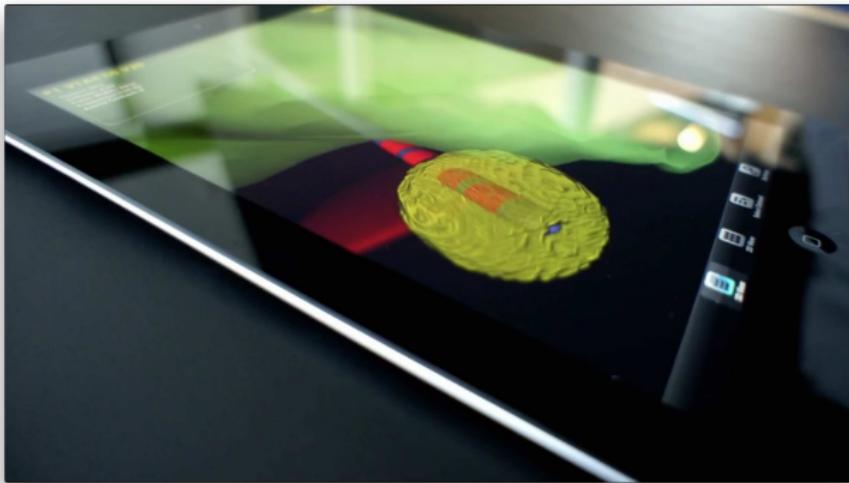
Entertainment



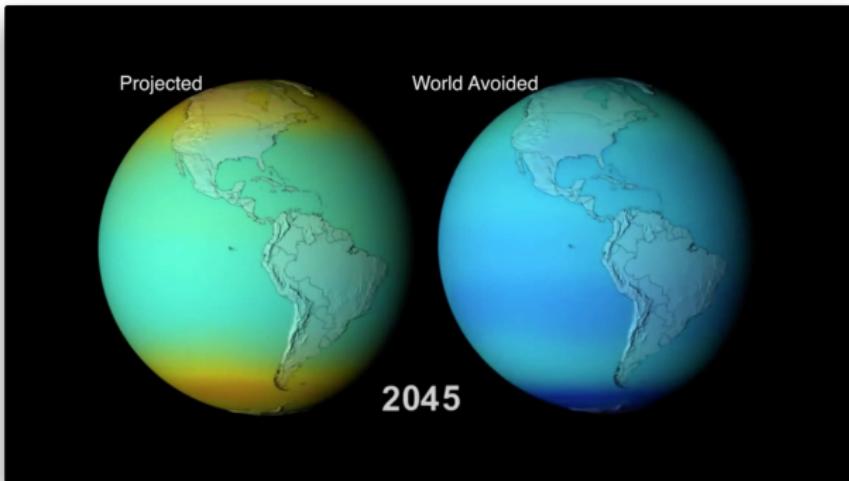
Epilepsy Treatments



Parkinson's Treatments



Human-Induced Climate Change



Missing Plane, MH370

Malaysia Airliner MH370: Water of an Airliner

Gowtham Chen, Cong Gu, Philip J. Morris, Er
Yi-Ching Wang, and Tomasz Wierzbicki

On March 8, 2014, Malaysia Airlines Flight MH370 disappeared less than an hour after take-off on a flight from Kuala Lumpur to Beijing. The Boeing 777-200ER carried twelve crew members and 227 passengers. The search and rescue teams announced that "it is believed beyond doubt" that the aircraft had crashed in the Southern Indian Ocean. Though the exact fate of the aircraft is still unknown, the available evidence indicates a crash into the ocean. However, although as this is, not all emergency survival equipment would have been deployed, many of them are compressed and in traps. In the "Wreck on the Beach" article, we discussed the case of Captain Sullenberger's "Miracle on the Hudson" and the video animation referenced on the second page of this article showed what happened to the aircraft in a nose-diving configuration.

Gong Chen is currently a mathematics at Texas A&M University (TAMU) and Texas A&M University at Qatar (TAU). He is also a member of the Institute for Numerical Simulation (INS) at TAU. His email address is gchen@math.tamu.edu.

Cong Gu is currently in the mathematics department of TAMU. His email address is gucong@math.tamu.edu.

Philip J. Morris is a Boeing 737 Walkin Professor of Aerospace Engineering at the University of Southern Mississippi. His email address is pjm@usm.edu.



(a) nose dive water entry

(b) pressure distribution and mesh

Time: 0.500

5e+05
4e+05
3e+05
2e+05
1e+05
0e+00

1e+04

Figure 12. Pitch angle = -90° , angle of approach = 93° . This corresponds to Case 4. A video animation can be viewed at <https://www.dropbox.com/s/vaf0qenjw0lk5yz/comb-90.mp4>.



(a)

(b)

Figure 13. Schematics for nose-diving. The ocean current pushes the aircraft to the right, causing it possibly to finish belly-up on the ocean floor.
This corresponds to Case 4.

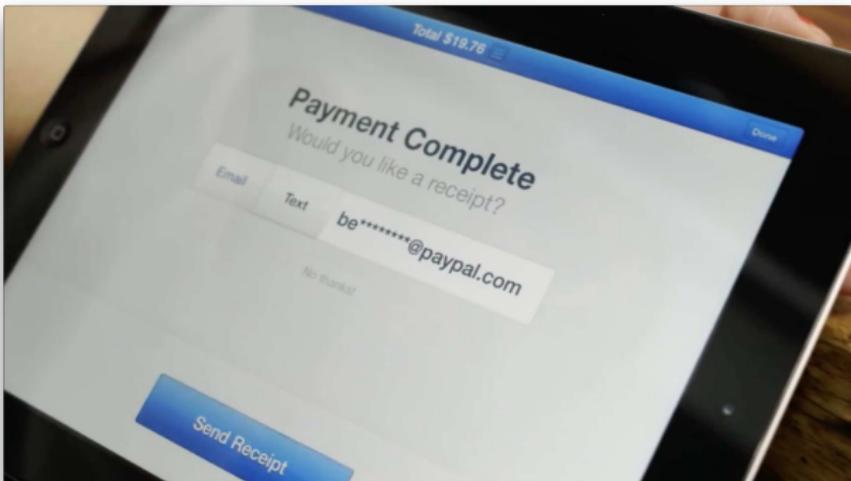
failure modes occur at low impact velocities, as has been demonstrated with a real model of a retired aircraft in DYCAST (Dynamic Crash Analysis of Structures) by NASA [FWR87]. These findings were published nearly three decades ago but remain valid today.

© Gowtham 2016

UN5390: Scientific Computing I · Week # 13 · Slide # 286

1889

PayPal



Storm Prediction



Precision Medicine



Preview of SC11



Preview of SC12



Preview of SC13



SCinet is a high-performance network that is built, once a year, in support of the annual International Conference for High Performance Computing, Networking, Storage and Analysis (SC). It is the primary network for the conference and is used by attendees to demonstrate and test high-performance distributed applications. Originated in 1991 as an initiative within the SC center.

Preview of SC14



Preview of SC15

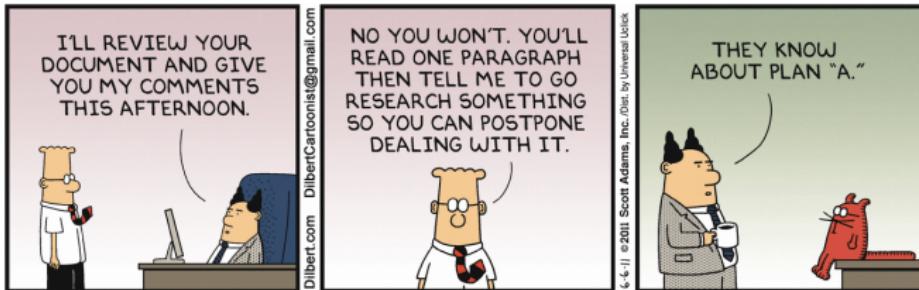


Preview of SC17

we cry. we rejoice. we reflect.

Review of Performance

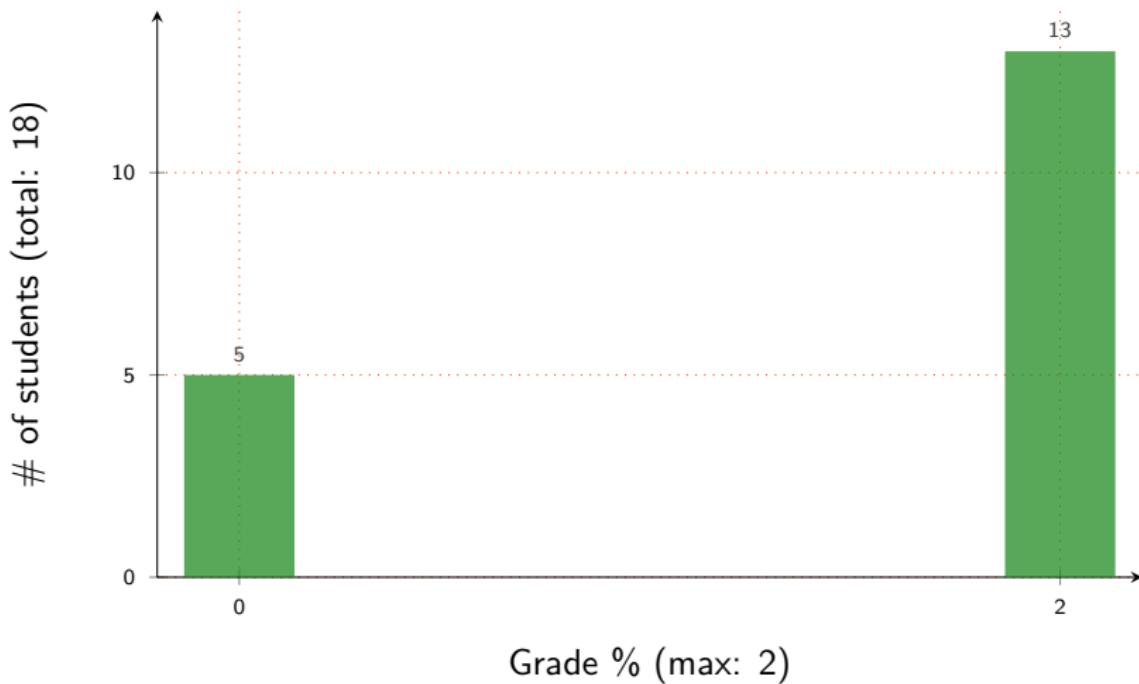
How well have we been performing?



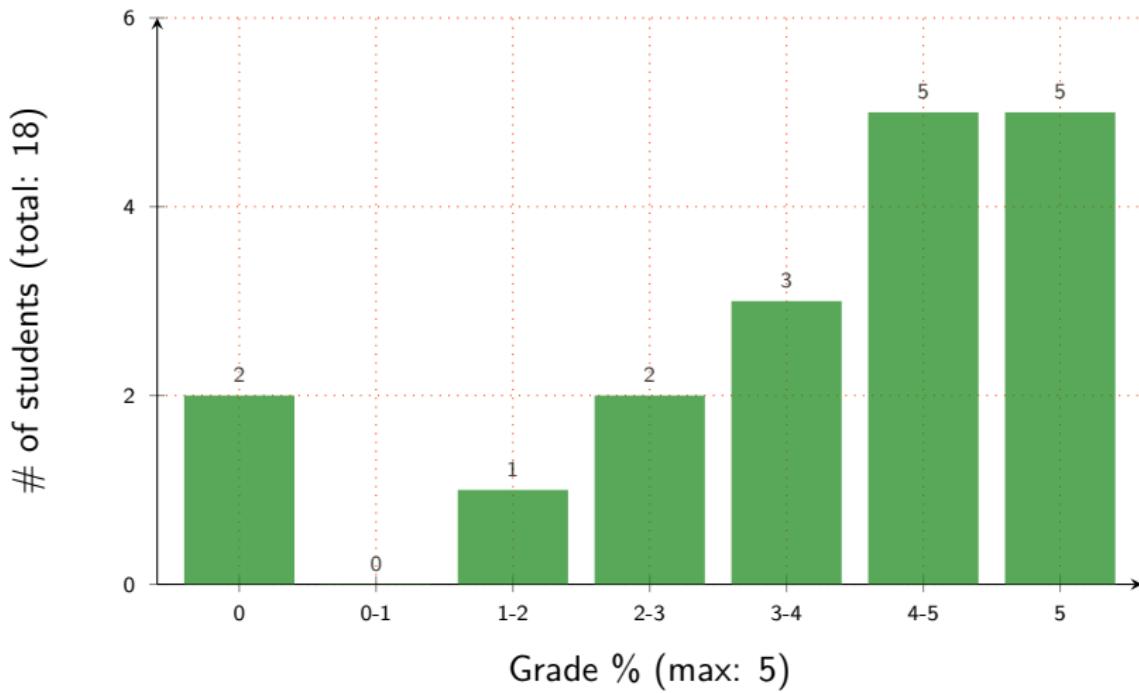
<http://dilbert.com/strip/2011-06-06/>

Active Participation #01

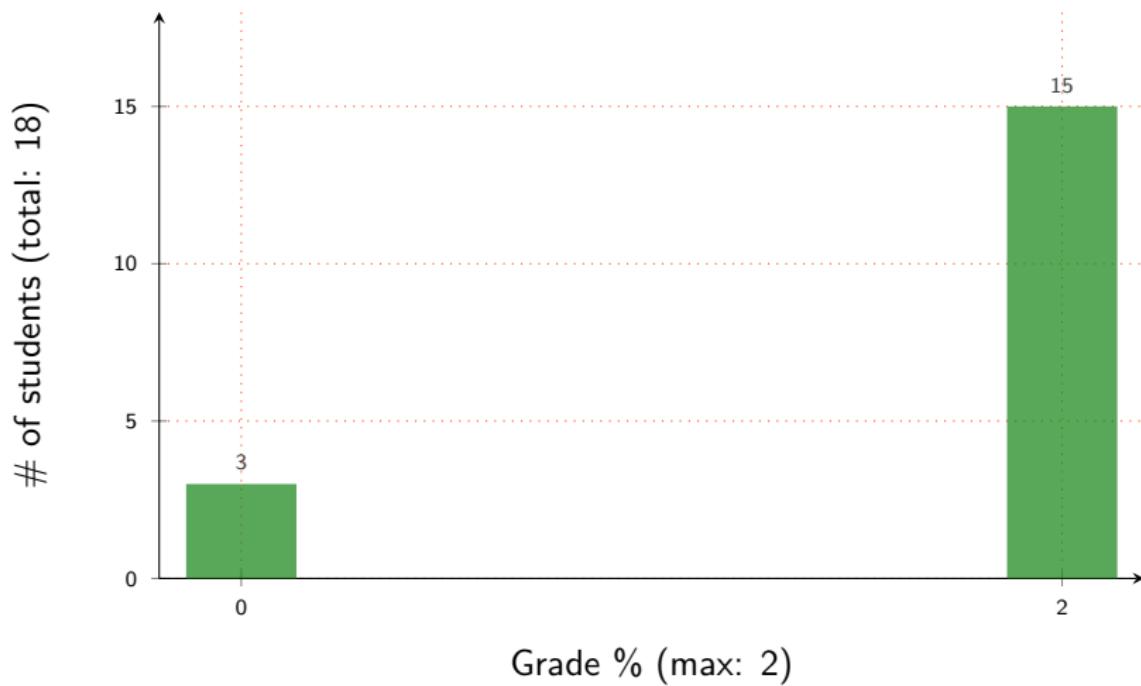
Research Marketing I: Twitter



Assignment #01

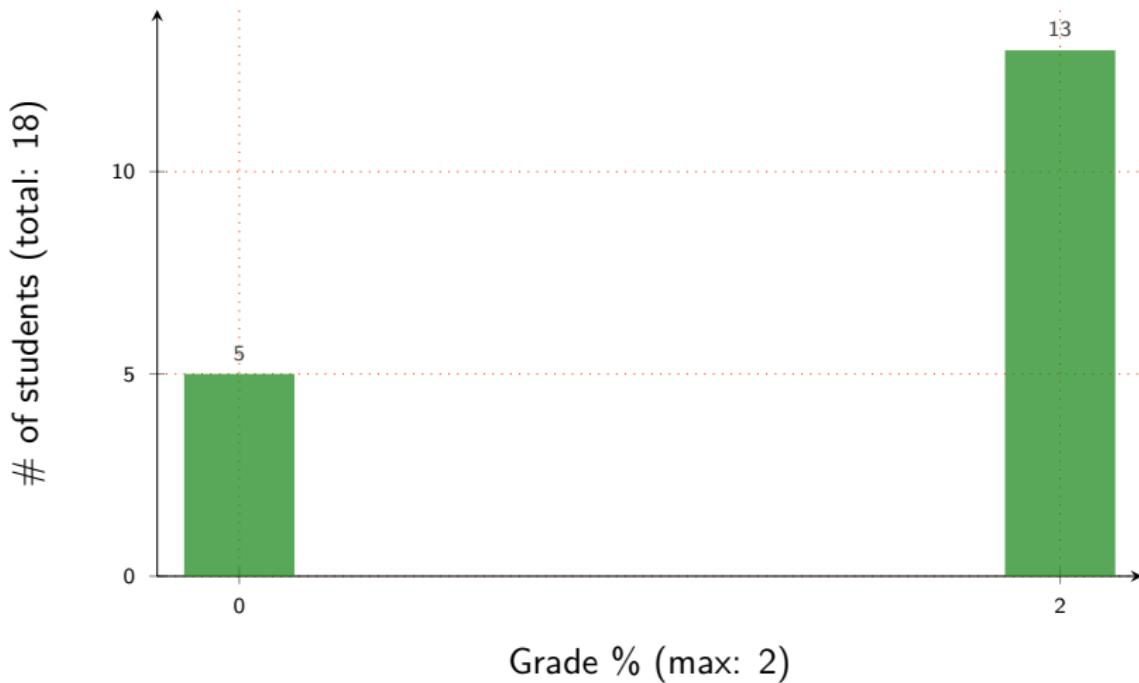


Active Participation #02 PB&J Sandwich Recipe

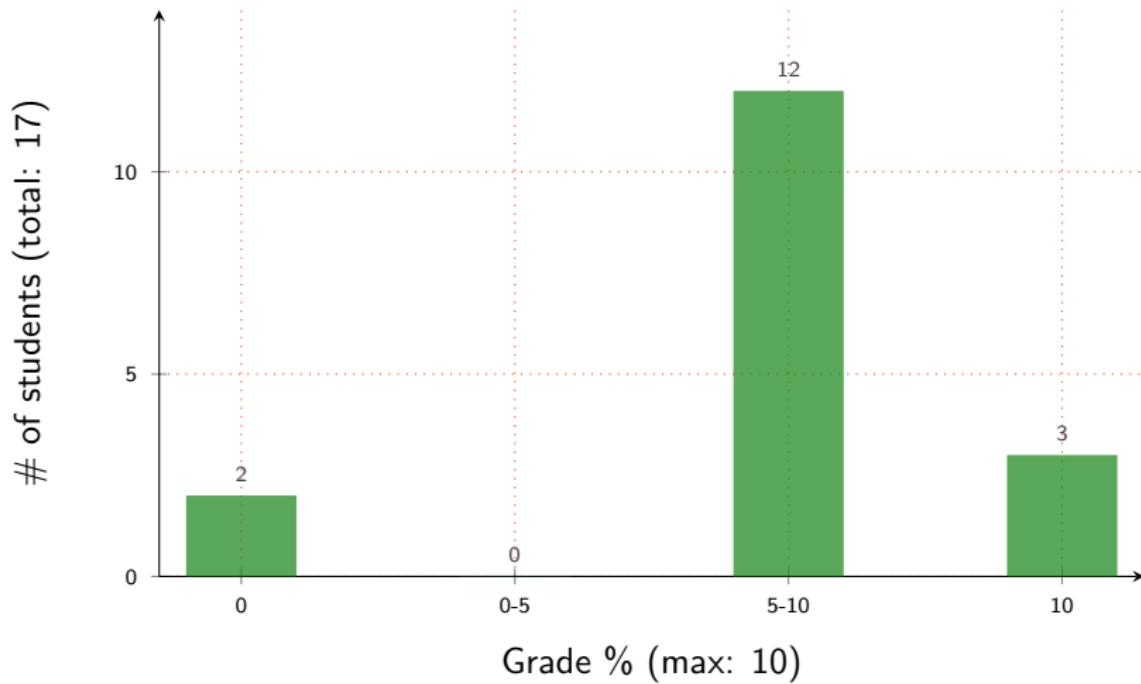


Active Participation #03

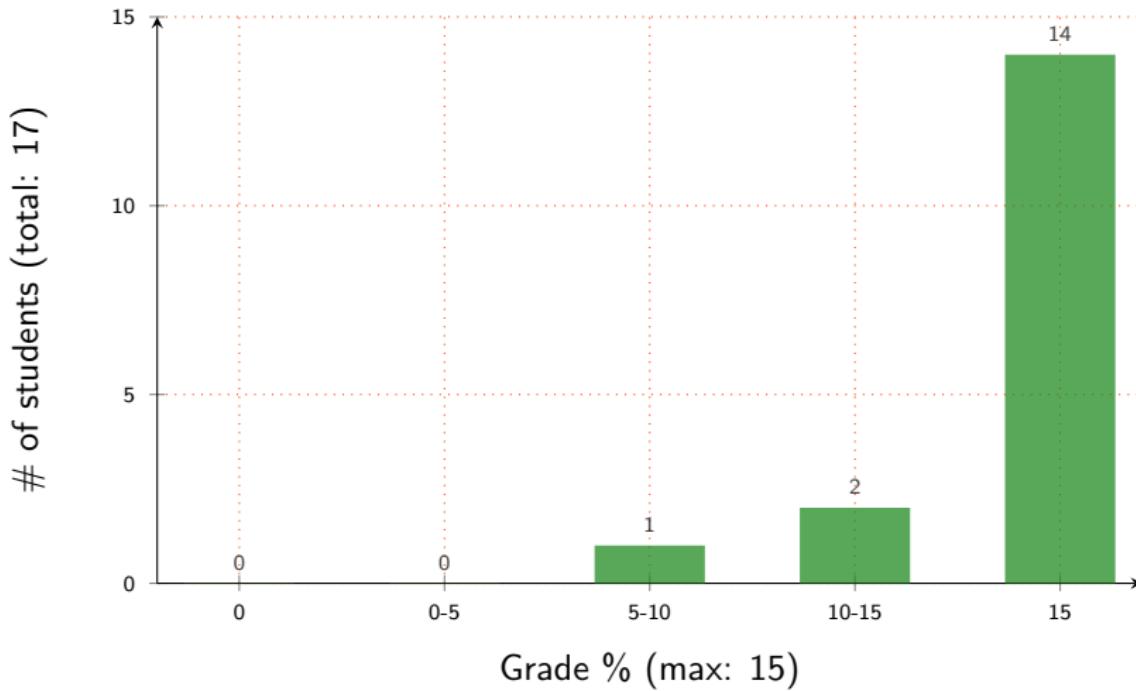
Research Marketing II: Professional/University Business Cards



Assignment #04



Assignment #07



Superior and Top 500

A proposed compute node in Superior will have two Intel Xeon E5-2698 processors (each processor with 20 cores) at 2.20 GHz, 512 GB RAM, 480 GB Intel Enterprise SSD, Mellanox ConnectX-3 56 Gbps InfiniBand network, and will cost \$13,263.13.

Ignoring the cost of physical space, racks, network, storage, electricity and labor, estimate the cost to build a #500 supercomputer (~405 TFLOPS) with homogeneous compute nodes as the ones described above.

For a computer with N identical/homogeneous processors,

$$\text{FLOPS} = N \times \text{CPU speed} \times \frac{\text{FLOPs}}{\text{CPU cycle}}$$

Celsius \longleftrightarrow Fahrenheit



Convert temperature between Celsius and Fahrenheit scales.

Is there a well-known technique to verify the conversion scheme?

Matrix elements



How many elements in a square matrix of order N ? How will this number change if the matrix is upper (or lower) triangular?

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \quad \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ 0 & b_{22} & \dots & b_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & b_{nn} \end{pmatrix}$$

The impact and limitations of Moore's Law



Assuming that Moore's Law holds true, what is the speed up of a computer observed over an average adult's life in the US?

Drawing queens



Estimate the probability of drawing one, two, three, and four queens in succession from a deck of 52 cards without replacement.

Got questions?

If you do, find a way to contact me; and do so sooner than later

EERC B39 · (906) 487-4096 · g@mtu.edu · [@sgowtham](https://twitter.com/@sgowtham)

Do not share/distribute the course material, in and/or outside of Michigan Tech, without instructor's prior consent

