Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method

MCS 572 Lecture 23
Introduction to Supercomputing
Jan Verschelde, 3 March 2023
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
The exam starts on Monday 6 March, at 2pm, in two ways:

- as a paper-and-pencil exam, with open book and notes, but without computer experimentation; due by 9pm on the same day.

or

- as a use-computer version due Wednesday 8 March, at 2pm, which requires computer experimentation.

The decision to do either version can be postponed, till Monday 6 March, 8:59pm.

Not submitting the paper-and-pencil version defaults to the use-computer version.
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
scaled speedup

Benchmarking of a program running on a 12-processor machine shows that 5% of the operations are done sequentially, i.e.: that 5% of the time only one single processor is working while the rest is idle.

Compute the scaled speedup.
Scaled speedup \( S_s(p) \) \( \leq \frac{st + p(1 - s)t}{t} = s + p(1 - s) = p + (1 - p)s. \)

Evaluate for \( s = 0.05, p = 12 \): \( S_s(12) = 12 + (1 - 12)0.05 = 11.45. \)
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
network topologies

Show that a hypercube network topology has enough connections for a fan-in gathering of results.
solution for $8 = 2^3$ nodes

Three steps:

1. $001 \rightarrow 000$; $011 \rightarrow 010$; $101 \rightarrow 100$; $111 \rightarrow 110$

2. $010 \rightarrow 000$; $110 \rightarrow 100$

3. $100 \rightarrow 000$
proof by induction

The base case: we verified for 1, 2, 4, and 8 nodes.

Assume we have enough connections for $2^k$ hypercube.

Need to show: have enough connections for $2^{k+1}$ hypercube:

1. In the first $k$ steps:
   - node 0 gathers from nodes $1, 2, \ldots 2^k - 1$;
   - node $2^k$ gathers from nodes $2^k + 1, 2^k + 2, \ldots, 2^{k+1} - 1$.

2. In step $k + 1$: node $2^k$ can send to node 0, because only one bit in $2^k$ is different from 0.
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
work stealing

Explain the concept of work stealing.

What is work stealing?

What is its purpose?

Describe an example of an environment and an application that uses work stealing.
In scheduling threads on processors, we distinguish between work sharing and work stealing:

- In work sharing, the scheduler attempts to migrate threads to under-utilized processors in order to distribute the work.
- In work stealing, under-utilized processors attempt to steal threads from other processors.

The purpose of work stealing is thus to utilize all processors. The Intel TBB task scheduler uses work stealing.

One application could be the computation of the best move as defined by the traversal of the game tree of all possible moves.
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
Define data parallelism and give an example.

What type of parallel computers target data parallel tasks?

Relate to the classification of Flynn.

Describe briefly a programming model to exploit data parallelism.
An application has data parallelism if many arithmetical operations can be executed independently on the data.

An example of data parallelism is the addition of two arrays of numbers, both of length $n$. The addition requires $n$ addition operations which can be performed independently on the $2n$ input elements.

In a data parallel program, all processors execute the same instruction but on different data. In Flynn’s classification, this is abbreviated as SIMD (Same Instruction Multiple Data).

CUDA (Compute Unified Device Architecture) defines a programming model where blocks of threads are launched on data arrays, for execution on a GPU (Graphics Processing Unit).
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
Consider the application of the composite Trapezoidal rule to compute the integral of \( f \) over \([a, b]\) by a distributed parallel program.

Let \( m \) be the number of floating point operations needed to evaluate \( f \) at one point. The cost of the composite Trapezoidal rule over \( n \) subintervals of \([a, b]\) requires \( n + 1 \) function evaluations.

1. Describe a high level parallel distributed memory with just enough detail to define the communication and computation cost, in terms of \( m \) and \( n \) as defined above, and \( p \) the number of processors.

2. Define the efficiency and analyze the scalability.
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
a parallel bisection method

Let $f$ be a continuous function over an interval $[a, b]$, with $f(a)f(b) < 0$.

- One step of the bisection method to find one root of $f$ in $[a, b]$ computes the midpoint $m = (a + b)/2$ and continues to bisect $[a, m]$ if $f(a)f(m) < 0$ or to bisect $[m, b]$ if $f(m)f(b) < 0$.

- Consider the application of the bisection method to the problem of approximating all roots of $f$ over an interval $[a, b]$.

- Given a tolerance $\delta$, the output consists in a list of subintervals of $[a, b]$ which contain one root of $f$, where the width of each subinterval is no larger than the value of $\delta$.

Answer the following questions:

1. Describe a shared memory parallel implementation. Use a high level task based description of the algorithm.

2. How does work stealing apply?
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
The modified Gram-Schmidt orthogonalization takes on input

\[ A \in \mathbb{R}^{m\times n}, A = [a_1 \ a_2 \ \cdots \ a_n]. \]

On output are
- \( Q = [q_1 \ q_2 \ \cdots \ q_n] \in \mathbb{R}^{m\times n}; \) and
- an upper triangular matrix \( R \) such that \( A = QR \).

Pseudo code is below.

for \( k \) from 1 to \( n \) do
   \[ q_k := a_k / \sqrt{a_k^T a_k} \]
   \[ r_{kk} := q_k^T a_k \]
   for \( j \) from \( k+1 \) to \( n \) do
      \[ r_{kj} := q_k^T a_j \]
      \[ a_j := a_j - r_{kj} q_k \]
for $k$ from 1 to $n$ do
  $q_k := a_k / \sqrt{a_k^T a_k}$
  $r_{kk} := q_k^T a_k$
  for $j$ from $k + 1$ to $n$ do
    $r_{kj} := q_k^T a_j$
    $a_j := a_j - r_{kj} q_k$

1. Define a parallel algorithm using tasking. The granularity of the algorithm should be fine enough to compute all inner products $a_k^T a_k$, $q_k^T a_k$, and $q_k^T a_j$ in parallel, for it to apply when $m \gg n$.

2. Draw the directed acyclic graph of all tasks for $m = 4$ and $n = 4$. In this graph, what is the length of a critical path? Explain how your analysis bounds the amount of parallelism.
Review for the Midterm Exam

1. Exam on Monday 6 March, at 2pm
   - paper-and-pencil or use-computer version

2. Some Past Computational Science Prelim Questions
   - scaled speedup
   - network topologies
   - work stealing
   - data parallelism

3. Some Previous Midterm Exam Questions
   - pleasingly parallel computations
   - a parallel bisection method
   - the modified Gram-Schmidt orthogonalization
   - CGMA ratio of Jacobi’s method
CGMA ratio of Jacobi’s method

The method of Jacobi is an iterative method to solve a linear system $A\mathbf{x} = \mathbf{b}$, defined by an $n$-by-$n$ coefficient matrix $A$, and an $n$-dimensional right hand side vector $\mathbf{b}$.

Starting at some approximation $\mathbf{z} \in \mathbb{R}^n$ one update $\Delta \mathbf{z}$ to $\mathbf{z}$ is computed by the following three steps:

(a) $\mathbf{r} = \mathbf{b} - A\mathbf{z}$;  
(b) $\Delta z_k = r_k / a_{k,k}$, for $k$ from 1 to $n$;  
(c) $\mathbf{z} = \mathbf{z} + \Delta \mathbf{z}$.

Answer the following questions:

1. Describe a data parallel version of the three steps above.
2. What is the CGMA ratio for the product $A\mathbf{z}$?