Parallel Computing with Julia

1. Parallel Symbolic Computing
   - three paradigms
   - blackboxes or computer algebra by values
2. Parallel Computing with Julia
   - numerical linear algebra
   - numerical integration

MCS 320 Lecture 42
Introduction to Symbolic Computation
Jan Verschelde, 29 July 2024

Parallel Computing with Julia

1. Parallel Symbolic Computing
   - three paradigms
   - blackboxes or computer algebra by values
2. Parallel Computing with Julia
   - numerical linear algebra
   - numerical integration

Parallel Computing

All computers are parallel. We distinguish between three types.

1. Distributed memory parallel computing.
2. Shared memory parallel computing.
3. Acceleration with Graphics Processing Units.

Parallel programs are evaluated by speedup and performance.

- Speedup is the serial time over parallel time.
- The number of floating-point operations per second (flops) measures the performance.

Parallel Computing with Julia

1. Parallel Symbolic Computing
   - three paradigms
   - blackboxes or computer algebra by values
2. Parallel Computing with Julia
   - numerical linear algebra
   - numerical integration

Parallel Symbolic Computing

We view the expressions as functions to evaluate at many points.

The evaluation at different points happens independently, in parallel.

A typical application is interpolation.

1. A polynomial of degree d has $d + 1$ coefficients and
   is determined uniquely by $d + 1$ function values at distinct points.
2. Power series are computed via the Fast Fourier Transform.

Parallel Computing with Julia

1. Parallel Symbolic Computing
   - three paradigms
   - blackboxes or computer algebra by values

2. Parallel Computing with Julia
   - numerical linear algebra
   - numerical integration

Numerical Linear Algebra

We apply multithreading in a Jupyter notebook,
in a kernel installed with the environment variable set to 16 threads.

```
julia> using IJulia
julia> installkernel("Julia (16 threads)",
       env = Dict("JULIA_NUM_THREADS"=>"16"))
```

The matrix-matrix multiplication is executed by `mul!()` of BLAS,
where BLAS stands for the Basic Linear Algebra Subroutines.

Two issues we must consider.

1. Choose the size of the matrices large enough.
2. The time should not include the compilation time.

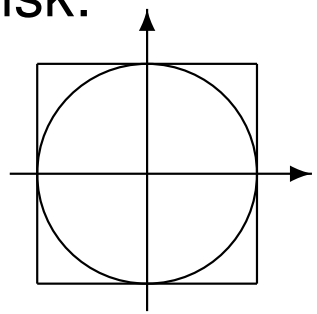
Parallel Computing with Julia

1. Parallel Symbolic Computing
   - three paradigms
   - blackboxes or computer algebra by values
2. Parallel Computing with Julia
   - numerical linear algebra
   - numerical integration

Numerical Integration

We can estimate π, via the area of the unit disk:

$$\int_0^1 \sqrt{1-x^2}dx = \frac{\pi}{4}$$



1. Generate random uniformly distributed points with coordinates
$$(x, y) \in [0, +1] \times [0, +1].$$
2. We count a success when $x^2 + y^2 \leq 1$.

By the law of large numbers,
the average of the observed successes converges to the expected value or mean, as the number of experiments increases.