# Unstructured integrator using MPI

Frank Tsai[1,2]

[1](SUNY at Buffalo)

[2](Göteborgs universitet)

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period
- total amount of stress exerted on a wing over a given area

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period
- total amount of stress exerted on a wing over a given area
- probability of an event in a probability space

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period
- total amount of stress exerted on a wing over a given area
- probability of an event in a probability space

How do we integrate functions?

- Analytic solution
- Numerical solution

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period
- total amount of stress exerted on a wing over a given area
- probability of an event in a probability space

How do we integrate functions?

- Analytic solution
  - Some functions have nonelementary antiderivatives, e.g., $\exp(x^2)$
- Numerical solution

# Motivation

We integrate functions to calculate the...

- total amount of heat generated over a fixed period
- total amount of stress exerted on a wing over a given area
- probability of an event in a probability space

How do we integrate functions?

- Analytic solution
  - Some functions have nonelementary antiderivatives, e.g., $\exp(x^2)$
- Numerical solution
  - Convergence, accuracy, and conservation

# Integration review

1. Intuition: integration accumulates local data into an integral [nLa24]

# Integration review

1. Intuition: integration accumulates local data into an integral [nLa24]
2. Classical approach: integration is defined by extending the "integration" of locally constant functions to measurable functions [Kle07]

# Integration review

1. Intuition: integration accumulates local data into an integral [nLa24]
2. Classical approach: integration is defined by extending the "integration" of locally constant functions to measurable functions [Kle07]
3. Measures are $\sigma$-additive: $\mu(\biguplus_i A_i) = \sum_i \mu(A_i)$

# Integration review

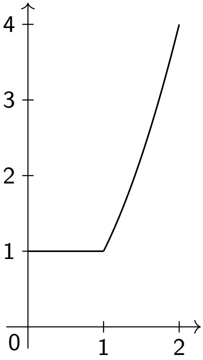
1. Intuition: integration accumulates local data into an integral [nLa24]
2. Classical approach: integration is defined by extending the "integration" of locally constant functions to measurable functions [Kle07]
3. Measures are $\sigma$-additive: $\mu(\biguplus_i A_i) = \sum_i \mu(A_i)$

**Idea**

To compute $\int_I f \, d(\sigma)$, divide $I$ into "equal" disjoint subsets.

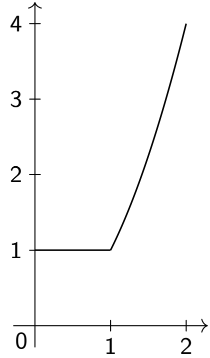
# Naïve (numerical) solution

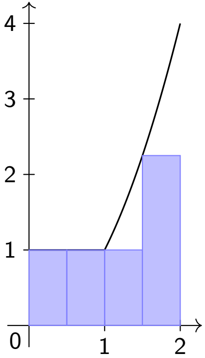
We want to integrate the function below



# Naïve (numerical) solution

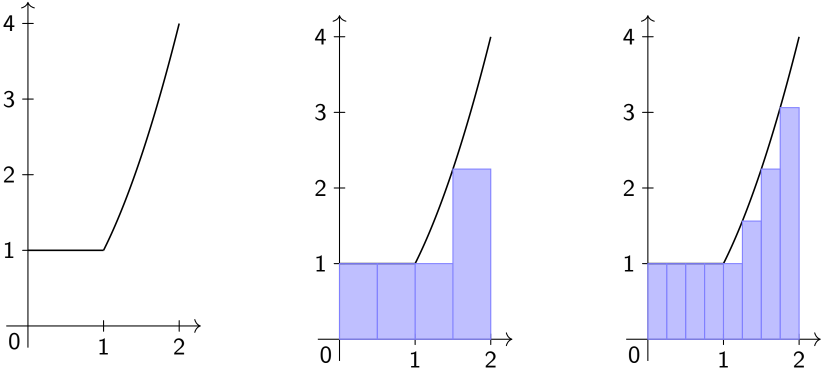
Divide the domain uniformly and calculate the area of each individual bar



# Naïve (numerical) solution

Refine the domain



# Observations

1. Evident parallelization of the naïve solution

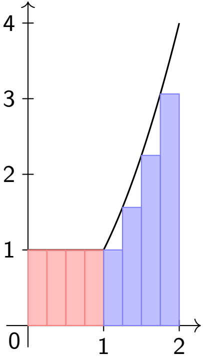
2. Coarse mesh is sufficient for constant functions

# Observations

1. Evident parallelization of the naïve solution
   - Each processor is responsible for a chunk of the domain
2. Coarse mesh is sufficient for constant functions

# Observations

1. Evident parallelization of the naïve solution
   - Each processor is responsible for a chunk of the domain
2. Coarse mesh is sufficient for constant functions
   - Refining the red region does not improve accuracy



# Unstructured integrator

An unstructured integrator using unstructured domain

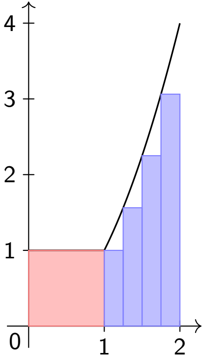
**Input:** mesh elements

**Output:** integral of a given function over the given mesh elements

# Unstructured integrator

An unstructured integrator using unstructured domain

| | |
|---|---|
| **Input:** | mesh elements |
| **Output:** | integral of a given function over the given mesh elements |



# Sequential algorithm

Suppose that the input contains chunks $M_1, M_2, \ldots, M_n$, where each $M_i$ is a (disjoint) collection of mesh elements.

1. Compute the integration of the given function $f$ over each $M_i$ and store the result in $s_i$

# Sequential algorithm

Suppose that the input contains chunks $M_1, M_2, \ldots, M_n$, where each $M_i$ is a (disjoint) collection of mesh elements.

1. Compute the integration of the given function $f$ over each $M_i$ and store the result in $s_i$
2. Compute the sum of $s_i$

# Parallel algorithm

Given $n$ processors, suppose that the input contains chunks $M_1, M_2, \ldots, M_n$, where each $M_i$ is a (disjoint) collection of mesh elements.

1. Distribute each chunk $M_i$ to a processor $P_i$

# Parallel algorithm

Given $n$ processors, suppose that the input contains chunks $M_1, M_2, \ldots, M_n$, where each $M_i$ is a (disjoint) collection of mesh elements.

1. Distribute each chunk $M_i$ to a processor $P_i$
2. In parallel, each processor $P_i$ computes the integration of the given function $f$ over $M_i$

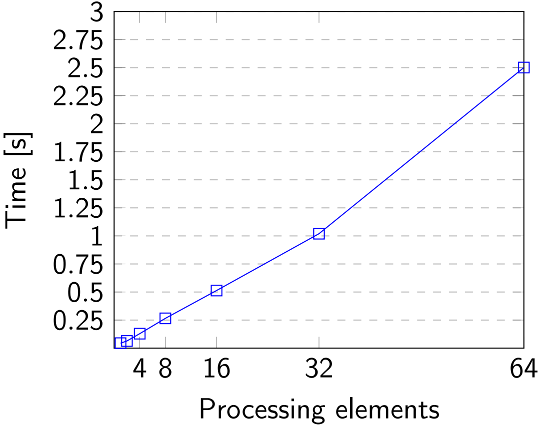
# Parallel algorithm

Given $n$ processors, suppose that the input contains chunks $M_1, M_2, \ldots, M_n$, where each $M_i$ is a (disjoint) collection of mesh elements.

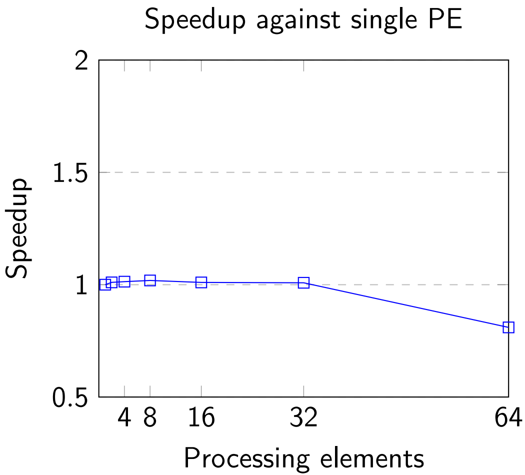
1. Distribute each chunk $M_i$ to a processor $P_i$
2. In parallel, each processor $P_i$ computes the integration of the given function $f$ over $M_i$
3. Compute the sum of the results via collective communication, and store the final result in the master processor $P_0$

# Strong scaling

Each process element computes 10 million mesh elements

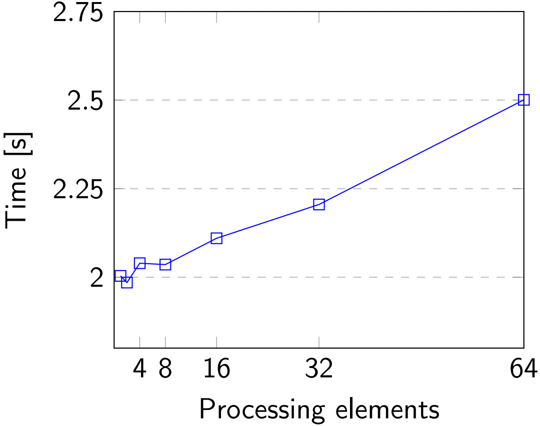


# Strong scaling

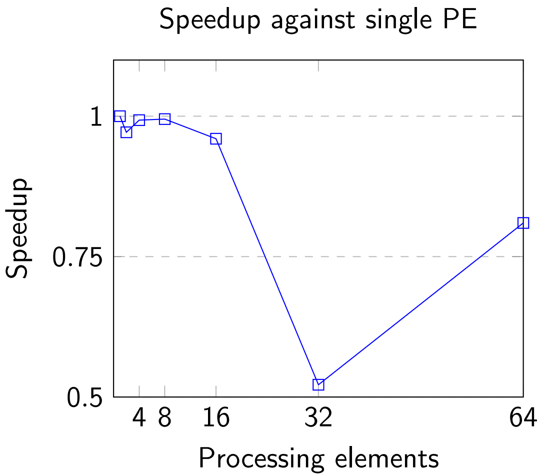


# Weak scaling

All processing elements collectively compute 640 million elements



# Weak scaling



# Future work

- Adaptive refinement [Tra97, MK06, BKL$^{+}$16]

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
    - Discretization by hand is not easy

# Future work

- Adaptive refinement [Tra97, MK06, BKL$^{+}$16]
  - Discretization by hand is not easy
  - Garbage in, garbage out

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A “domain specific language” for measure theory

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A "domain specific language" for measure theory
  - A *synthetic* development of measure theory

# Future work

- Adaptive refinement [Tra97, MK06, BKL$^{+}$16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A "domain specific language" for measure theory
  - A *synthetic* development of measure theory
  - Euclidean geometry is a synthetic theory

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A "domain specific language" for measure theory
  - A *synthetic* development of measure theory
  - Euclidean geometry is a synthetic theory
- Topos theoretic approach to measure theory

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A "domain specific language" for measure theory
  - A *synthetic* development of measure theory
  - Euclidean geometry is a synthetic theory
- Topos theoretic approach to measure theory
  - Topoi are generalized sets

# Future work

- Adaptive refinement [Tra97, MK06, BKL+16]
  - Discretization by hand is not easy
  - Garbage in, garbage out
- A “domain specific language” for measure theory
  - A *synthetic* development of measure theory
  - Euclidean geometry is a synthetic theory
- Topos theoretic approach to measure theory
  - Topoi are generalized sets
  - Similar work has been done in [Jac06]

# Bibliography I


Nicolas Barral, Matthew G Knepley, Michael Lange, Matthew D Piggott, and Gerard J Gorman.
Anisotropic mesh adaptation in firedrake with petsc dmplex.
*arXiv preprint arXiv:1610.09874*, 2016.

Matthew Jackson.
*A sheaf theoretic approach to measure theory*.
PhD thesis, University of Pittsburgh, 2006.

Achim Klenke.
*Probability Theory: A Comprehensive Course*.
Springer, 2007.

# Bibliography II


Matthias Möller and Dmitri Kuzmin.
Adaptive mesh refinement for high-resolution finite element schemes.
*International journal for numerical methods in fluids*, 52(5):545–569, 2006.

nLab authors.
integral.
`https://ncatlab.org/nlab/show/integral`, March 2024. Revision 27.

B. A. Shadwick, John C. Bowman, and P. J. Morrison.
Exactly conservative integrators.
*SIAM Journal on Applied Mathematics*, 59(3):1112–1133, 1998.

Christoph T Traxler.
An algorithm for adaptive mesh refinement in n dimensions.
*Computing*, 59:115–137, 1997.