Analyses of How Code Organization Impacts Development-Time Process

Damian Rouson
Sandia National Laboratories

Joel Koplik, Xiaofeng Xu, Karla Morris
City University of New York

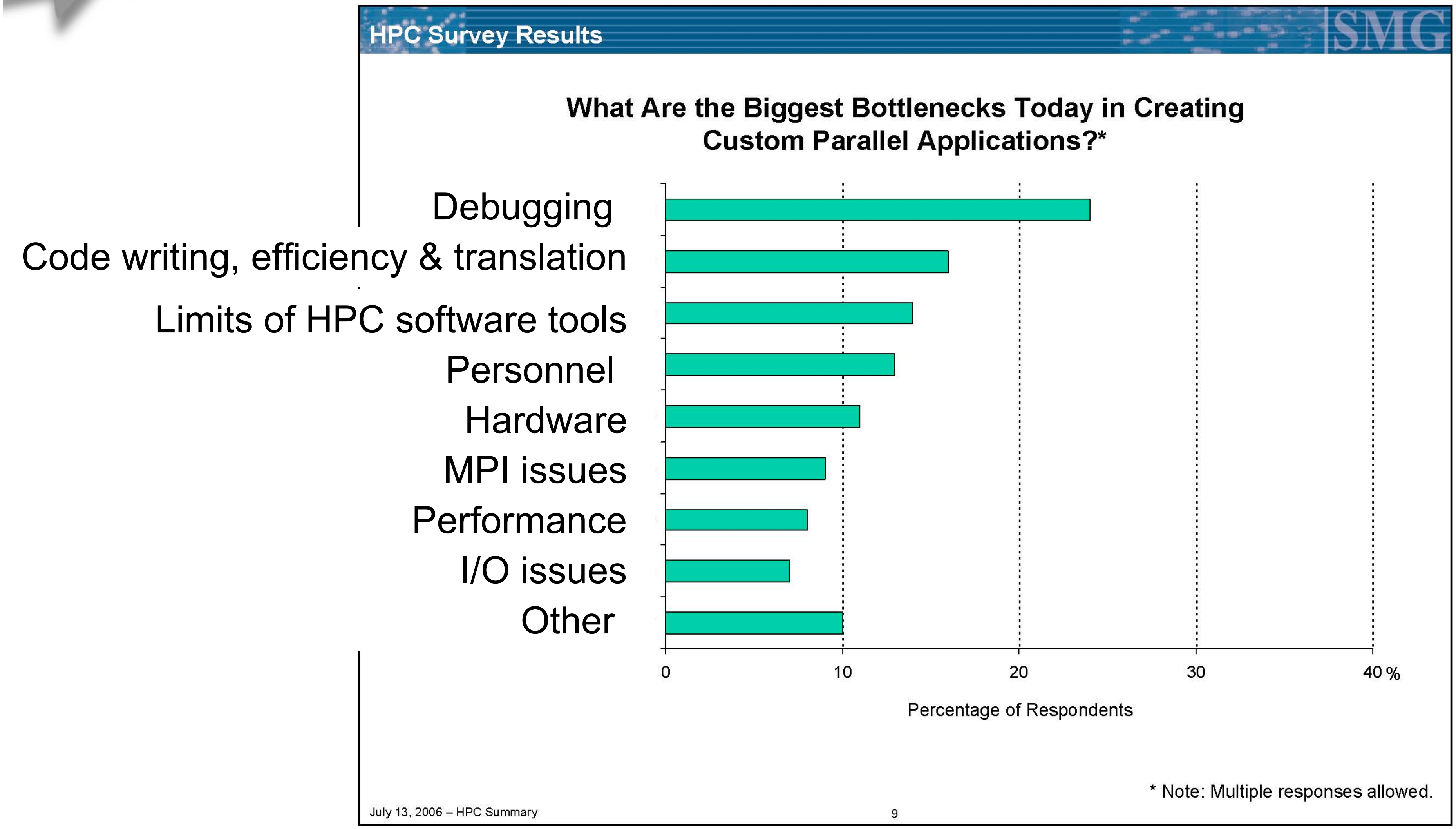
Robert Rosenberg
U.S. Naval Research Laboratory

Stavros Kassinos, Irene Moulitsas
University of Cyprus

Sponsors:
Office of Naval Research, National Science Foundation, European Commission

Sandia National Laboratories

Motivation



HPC Survey Results
SMG
What Are the Biggest Bottlenecks Today in Creating Custom Parallel Applications?*
Debugging
Code writing, efficiency & translation
Limits of HPC software tools
Personnel
Hardware
MPI issues
Performance
I/O issues
Other
0
10
20
30
40 %
Percentage of Respondents
* Note: Multiple responses allowed.
July 13, 2006 – HPC Summary
9

Objectives

1. To analyze how development time scales with program size & how this depends on the choice of abstractions.
2. To develop strategies for reducing development time.
3. To demonstrate *scalable development* of multiphysics models.

Sandia National Laboratories

Outline

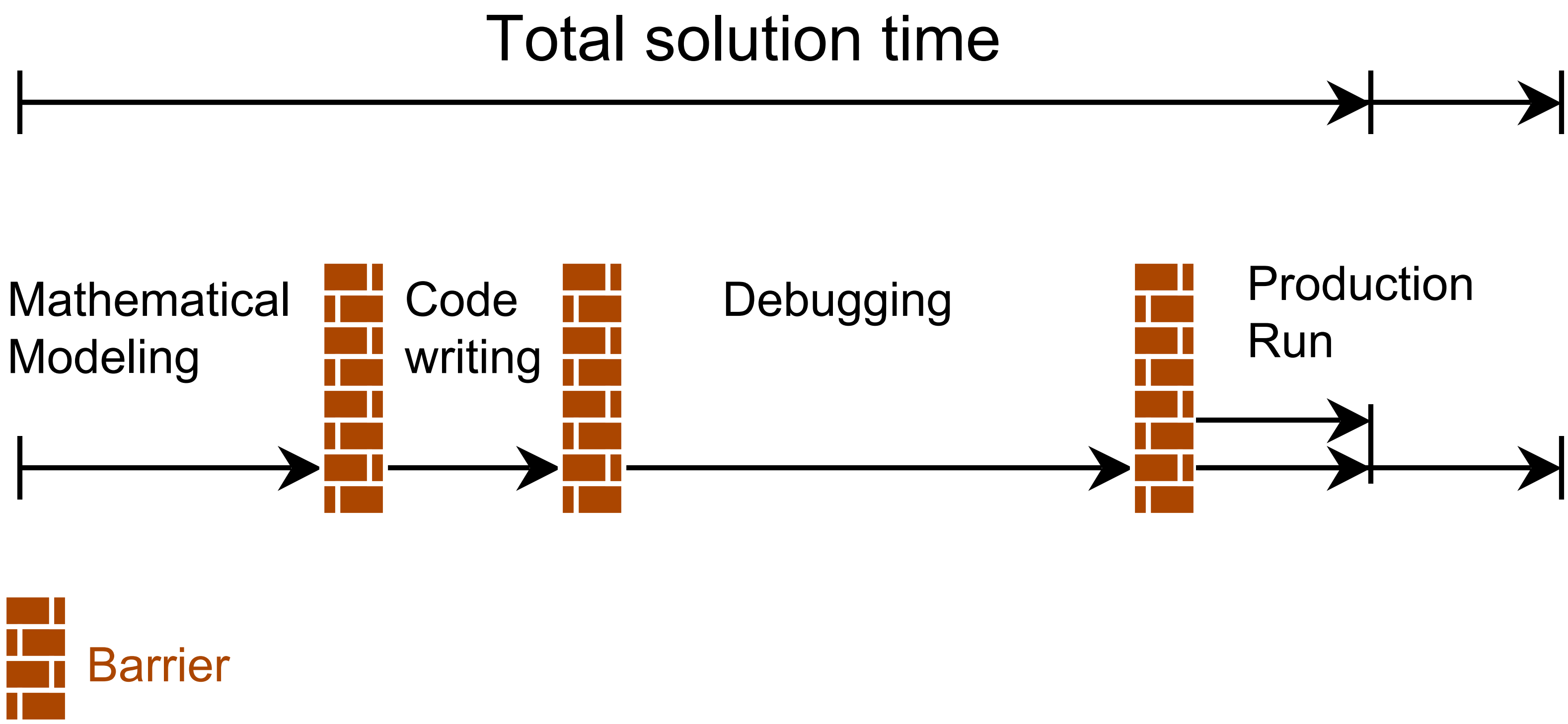
1. Analyses of the impact of
   a. Coding efficiency on total solution time.
   b. Programming paradigm on debugging.
   c. Abstraction choice on interface content.
2. Multiphysics model demonstrations
3. Conclusions & Future Work

Outline

1. Analyses of the impact of
   a. Coding efficiency on total solution time.
   b. Programming paradigm on debugging.
   c. Abstraction choice on interface content.
2. Multiphysics model demonstrations
3. Conclusions & Future Work

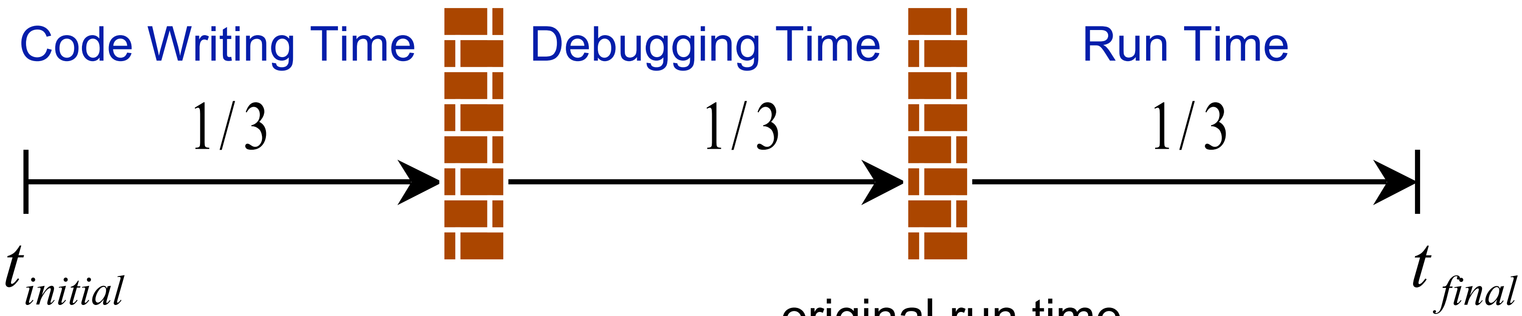
Sandia National Laboratories

Conventional Development



Amdahl's Law

Representative case study for a published run[1,2]:



Run-time speedup: $S_{run} \equiv \dfrac{\text{original run time}}{\text{optimized run time}}$

Total speedup: $$S_{tot} = \frac{1}{\frac{2}{3} + \frac{1}{3}\frac{1}{S_{run}}} \Rightarrow \lim_{S_{run} \to \infty} S_{tot} = 1.5$$

The speedup achievable by focusing solely on decreasing run time is very limited.

[1]Rouson et al. (2008) *Physics of Fluids*.

[2]Rouson et al. (2008) *ACM Transactions on Mathematical Software*.

Sandia National Laboratories

Case Study: Isotropic Turbulence

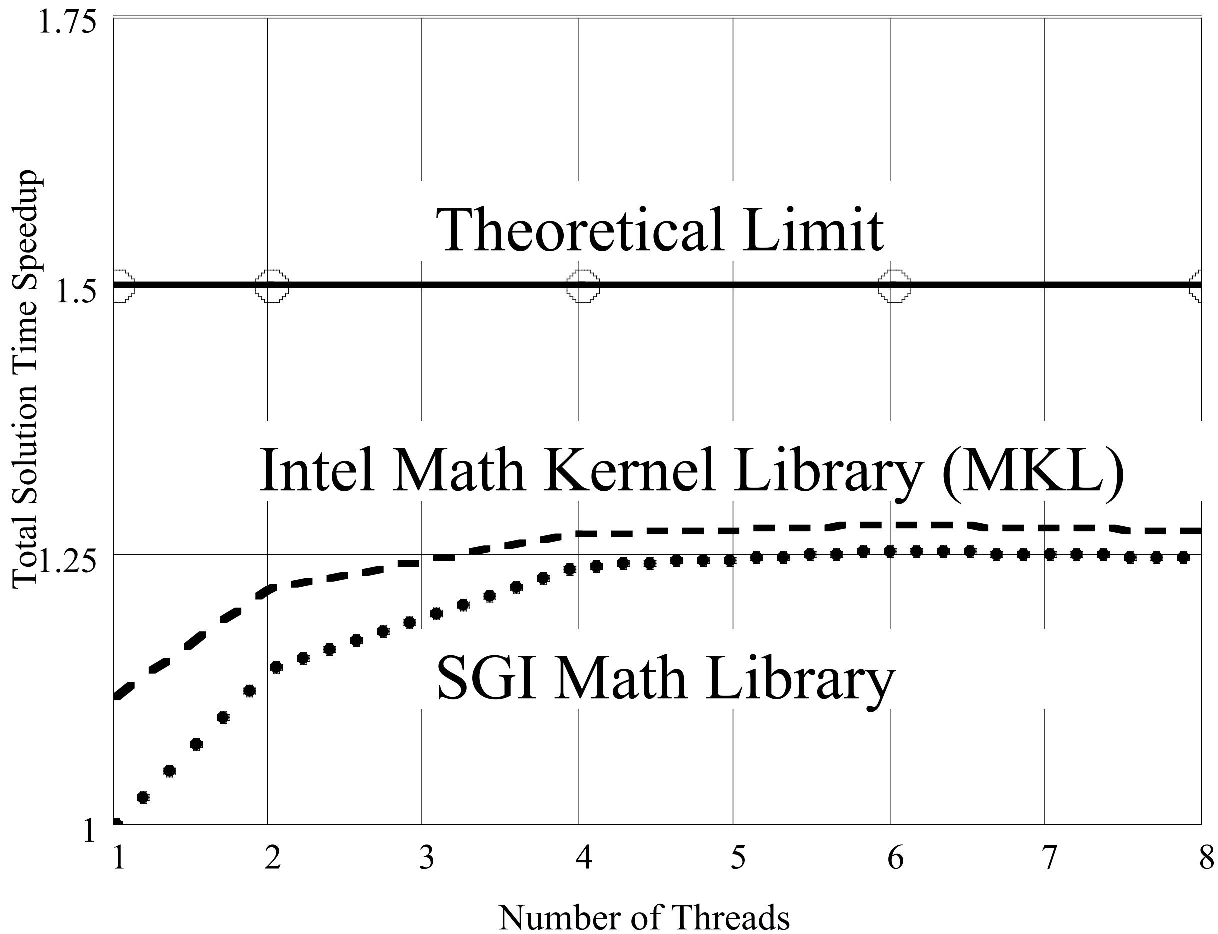
Procedure	Inclusive Run-Time Share (%)
`main`	100.0
`operator(.x.)`	79.5
`RK3_Integrate()`	47.8
`Nonlinear_Fluid()`	44.0
`Statistics_`	43.8
`transform_to_fourier`	38.7
`transform_to_physical`	23.6

Calls

- 5% procedures occupy nearly 80% of run time.
- Structure 95% of procedures to reduce development time.

Sandia National Laboratories

Total Solution Time Speedup



Pareto Principle

When participants (lines) share resources (run time), there always exists a number $k \in [50,100)$ such that (1-k)% of the participants occupy k% of the resources:

Limiting cases:

- k=50%, equal distribution
- k→100%, monopoly

Rule of thumb: 20% of the lines occupy 80% of the run time

Scalability requirements determine the percentage of the code that can be focused strictly on programmability:

$$S_{\max} = \lim_{S_{k\%} \to \infty} \frac{1}{0.2 + 0.8 / S_{k\%}} = 5$$

Sandia National Laboratories

Outline

1. Analyses of the impact of
   a. Coding efficiency on total solution time.
   b. Programming paradigm on debugging.
   c. Abstraction choice on interface content.
2. Multiphysics model demonstrations
3. Conclusions & Future Work

Sandia National Laboratories

“Separate the physics from the data.”

Jaideep Ray

Sandia National Laboratories, ca. 2005

“Software abstractions should resemble blackboard abstractions.”

Kevin Long

Texas Tech. Univ., ca. 2007

Abstract Data Type Calculus

Blackboard abstraction	Software abstraction (Fortran 2003):
$T = T(x, y, z, t)$	`type(field) :: T,dT_dt,laplacian_T`
$T(x = 0, y, z, t) = T_0$	`call T%boundary(x,0,T0)`
$T_t \equiv \frac{\partial T}{\partial t}$	`T%t()`
$T^{n+1} = T^n + \Delta t T_t^n$	`T = T + dt*T%t()`
$\frac{\partial T}{\partial t} = \frac{1}{\alpha} \nabla^2 T$	`dT_dt = (1./alpha)*laplacian(T)`
$\nabla^2 T \equiv T_{xx} + T_{yy} + T_{zz}$	`laplacian_T = T%xx()+T%yy()+T%zz()`

Sandia National Laboratories

Abstract Data Type (ADT)

```
module field_class                      !C++ namespace
  implicit none
  private
  type, public :: field                 !C++ class
    private                             !C++ data members
    real, dimension(:,:,:), allocatable :: nodalValues
  contains
    procedure :: boundary               !C++ member functions
    procedure :: plus                   !C++ overloaded operator
    generic, public :: operator(+)=>plus
  end type
contains
  subroutine boundary(this,direction,location,value)
    class(field) :: this                !C++ dynamic dispatching
    ...
  end subroutine
  function plus(lhs,rhs) result(total)
    class(field), intent(in)  :: lhs,rhs
    class(field), allocatable :: total
    ...
```

Sandia National Laboratories

“Procedural programming is like an N-body problem.”

Lester Dye, Stanford University, ca. 1994

“What are the metrics?”

Oyekunle Olukotun, ca. 1996

Stanford University

Sandia National Laboratories

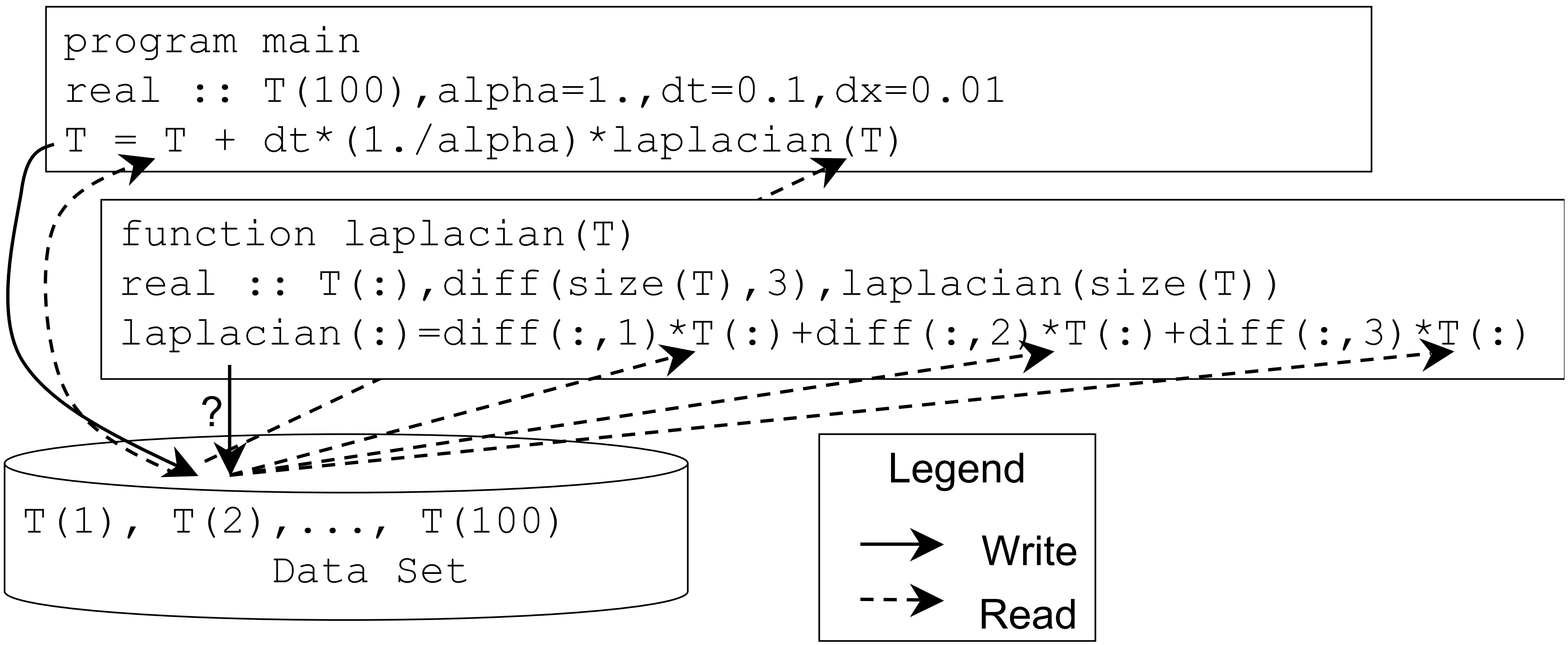
“Not much time is spent fixing bugs. Most of the time is spent *finding* bugs.”

Shalloway & Trott (2002) *Design Patterns Explained*

Oliveira & Stewart (2006) *Writing Scientific Software*

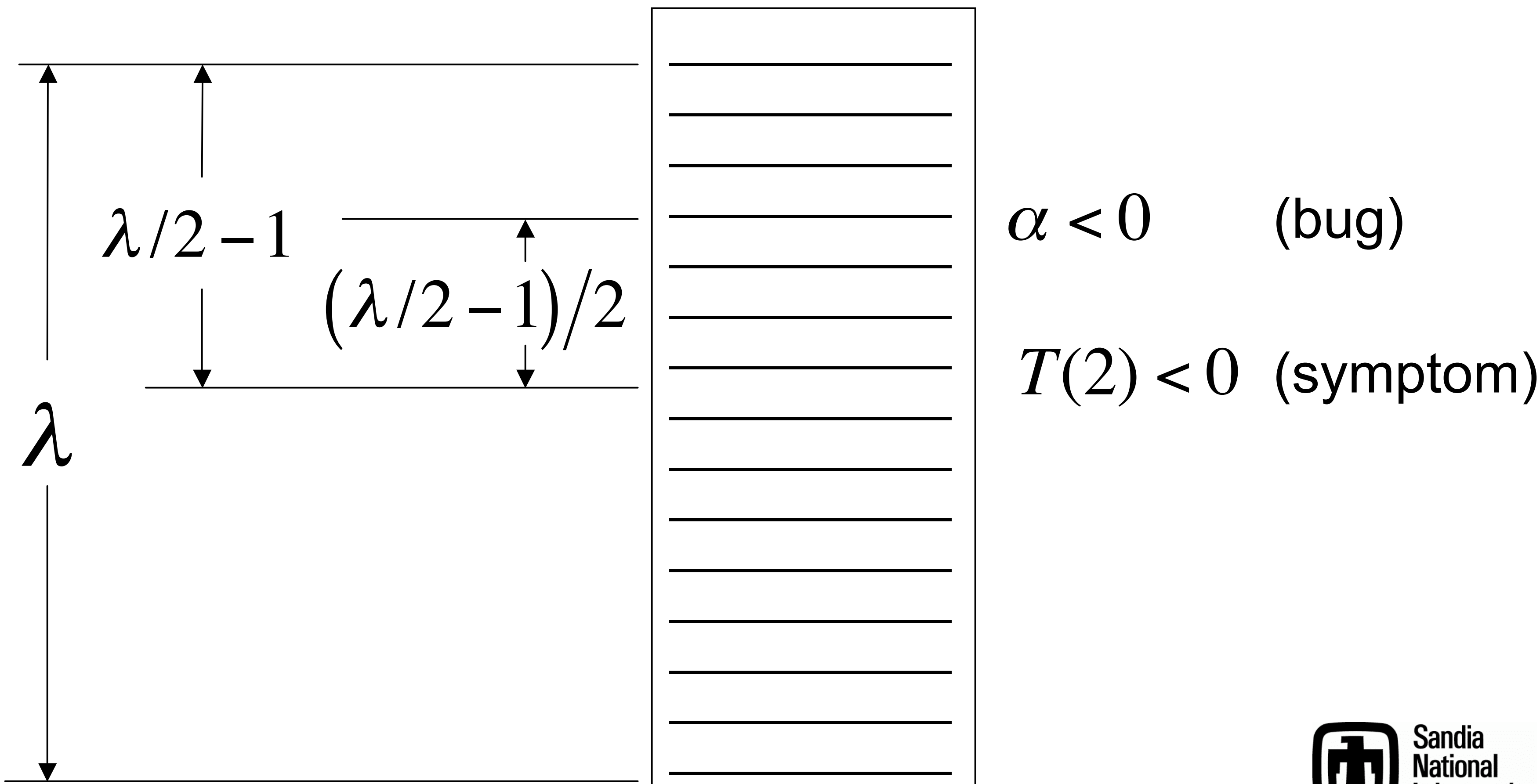
Sandia National Laboratories

Debugging Structured Programs

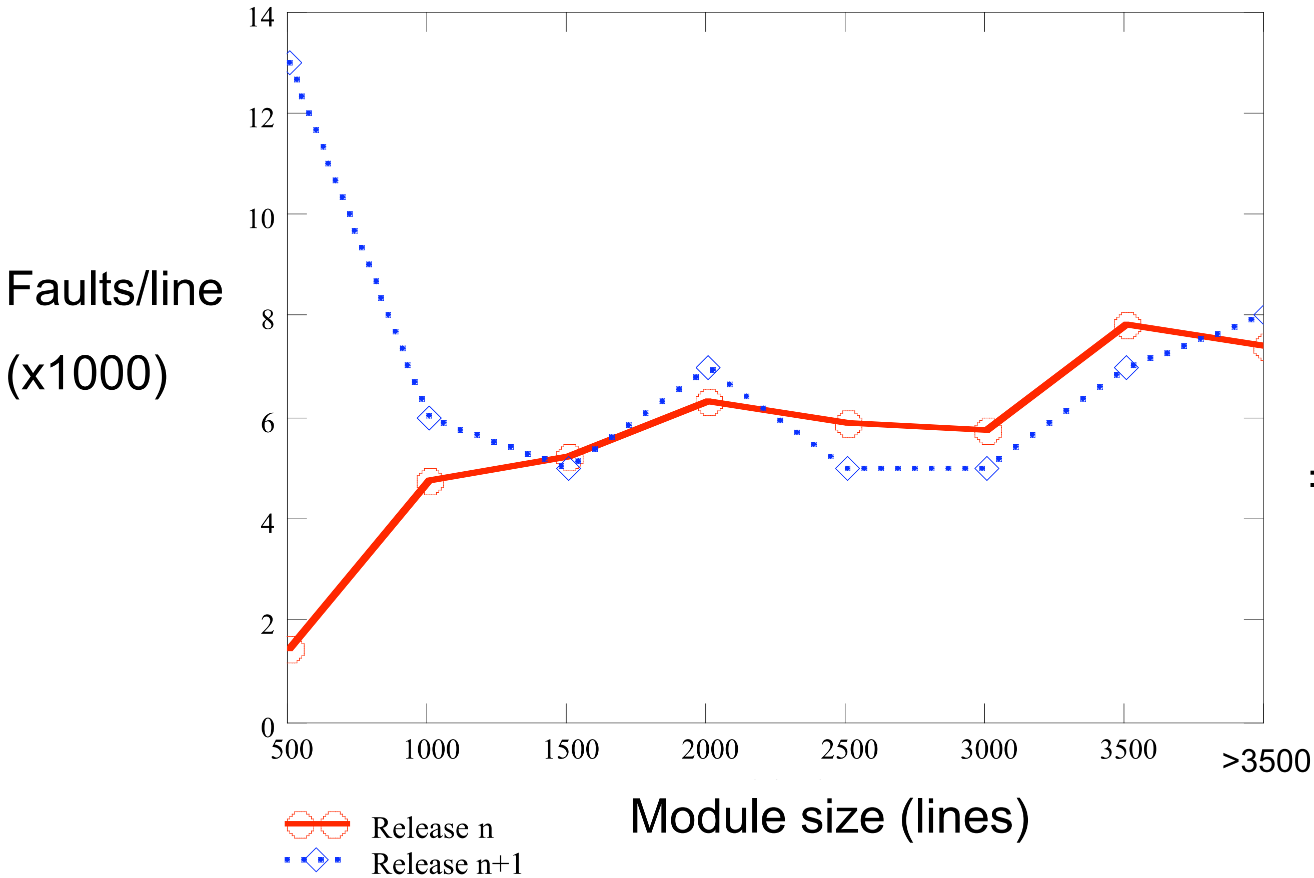


Fault Localization

"Computational" complexity theory: Derive a polynomial time estimate for fault localization in a chronological list of the <u>unique</u> program lines executed:



Fault Rate



$$\Rightarrow r \approx 6/1000$$

Source: Fenton & Ohlssen (2000) "Quantitative analysis of faults and failures in a complex software system," *IEEE Trans. Soft. Eng.*

Sandia National Laboratories

Scientific Code Faults

Observed faults in *commercially released* code*:

- 8 statically detectable faults/1000 lines of C code
- 12 statically detectable faults/1000 lines of Fortran 77 code
- more recent data finds 2-3 times as many faults in C++

$$\Rightarrow r \approx 0.006 - 0.036$$

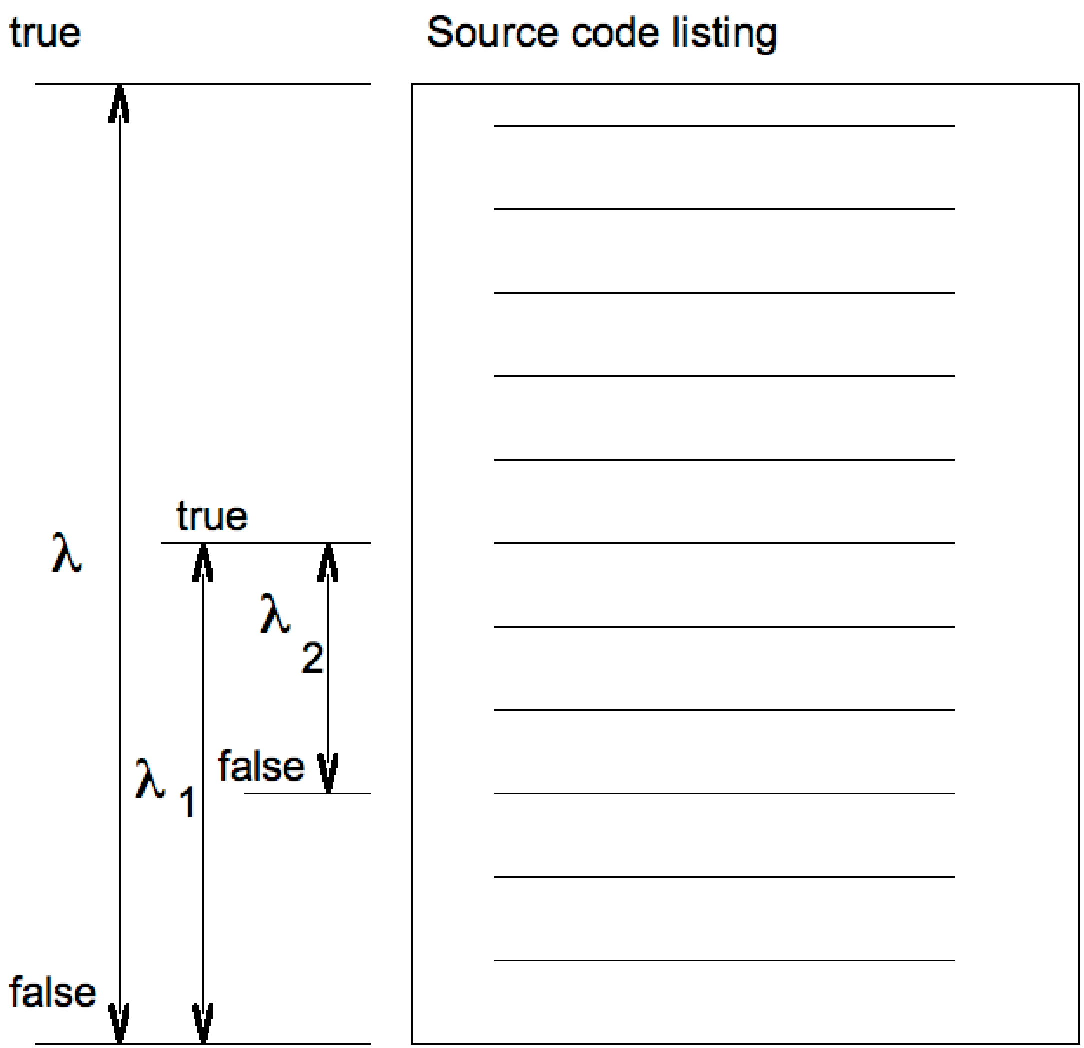
$$t_{search} = (\#\,bugs) \times (lines\ searched\ per\ bug)(\bar{t}_{line\ review})$$

$$= (r\lambda)\left[(\lambda/2 - 1)/2\right](\bar{t}_{line\ review})$$

*Source: Hatton, L. (1997) "The `T' Experiments – Errors in Scientific Software," *Comp. Sci. Eng.*

Sandia National Laboratories

Bisection Search Time



Localization error:

$$\lambda_n = \frac{\lambda}{2^n}$$

Convergence criterion:

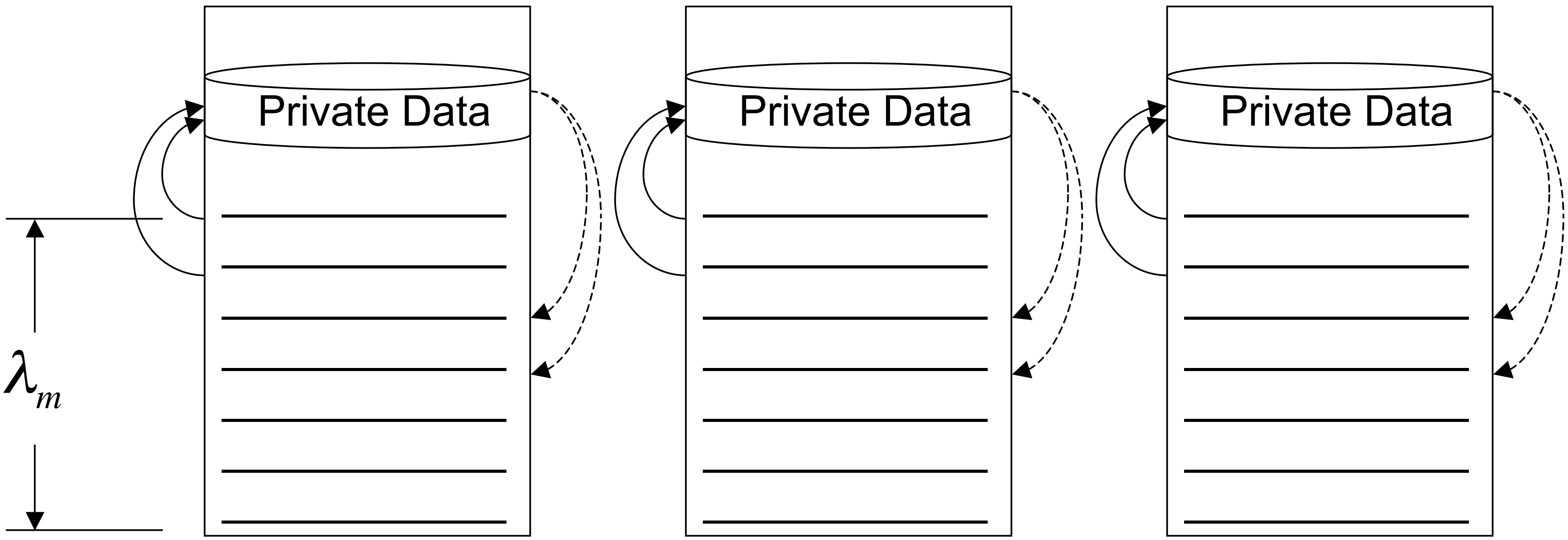
$$\frac{\lambda}{2^q} = 1$$

$$q = \log_2 \lambda$$

Search time metric:

$$\lambda_{\text{searched}} = r\lambda \log_2 \lambda$$

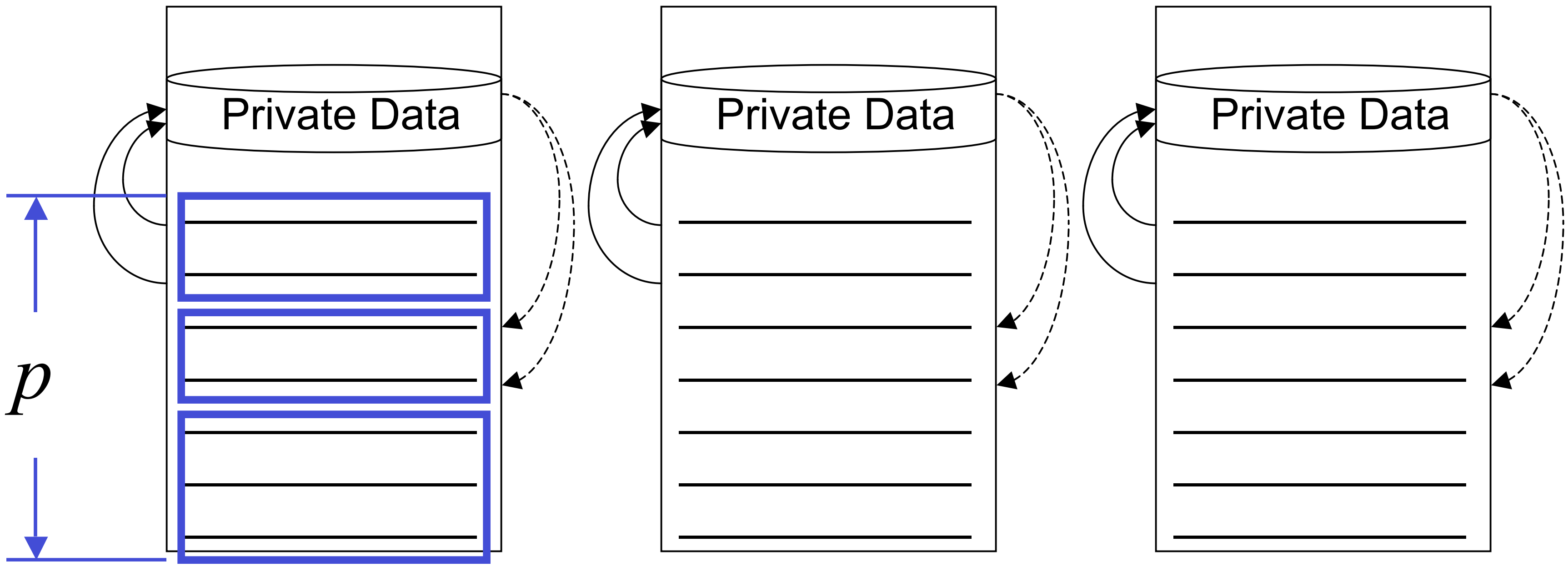
Object-Oriented Program



$$\lambda_{searched} = r\lambda_m \log_2 \lambda_m$$

$$\lambda_m << \lambda$$

ADT Calculus



Procedural line density:

$$\rho \equiv \frac{\lambda_m}{p} = \frac{\textit{lines per module}}{\textit{procedures per module}} \quad \Rightarrow \quad \lambda_{searched} = (r\rho p)\log_2 \rho p$$

For ADT calculus, $\left.\begin{array}{l}\rho \approx const \\ p \approx const\end{array}\right\} \Rightarrow \lambda_{searched} \approx const$

Outline

1. Analyses of the impact of
   a. Coding efficiency on total solution time.
   b. Programming paradigm on debugging.
   c. Abstraction choice on interface content.
2. Multiphysics model demonstrations
3. Conclusions & Future Work

Interface Content

Abstract class interface (Unified Modeling Language):

integrable_model
+ operator(+)(integrable_model,integrable_model) : integrable_model + operator(*)(real,integrable_model) : integrable_model + t(integrable_model) : integrable_model

A single interface describes all of the public information for all classes that extend this class.

Information Entropy

Shannon (1948) “A mathematical theory of communication,” *Bell System Tech. J.*

The class interfaces embody inter-developer communications. Consider the set of all (N) possible messages that can be transmitted between two developers:

> “If the number of messages in the set is finite, then this number or any monotonic function of this number can be regarded as a measure of the information produced when one message is chosen from the set, all choices being equally likely.”

Shannon chose the logarithm because it satisfies several constraints that match our intuitive understanding of information:

$$H = -\sum_{i=1}^{N} p_i \log_2 p_i = -\sum_{i=1}^{N} \frac{1}{N} \log_2 \frac{1}{N} = \log_2 N$$

Sandia National Laboratories

Minimum Information Growth

```
subroutine integrate(integrand)
  class(integrable_model) :: integrand
  integrand = integrand + dt*integrand%t()
end subroutine
```

If only one class extends integrable_model, the executable line only has one possible interpretation, so *H=0*. Each subsequent subclass increases the information content by

$$\Delta H = \log_2(N+1) - \log_2 N$$

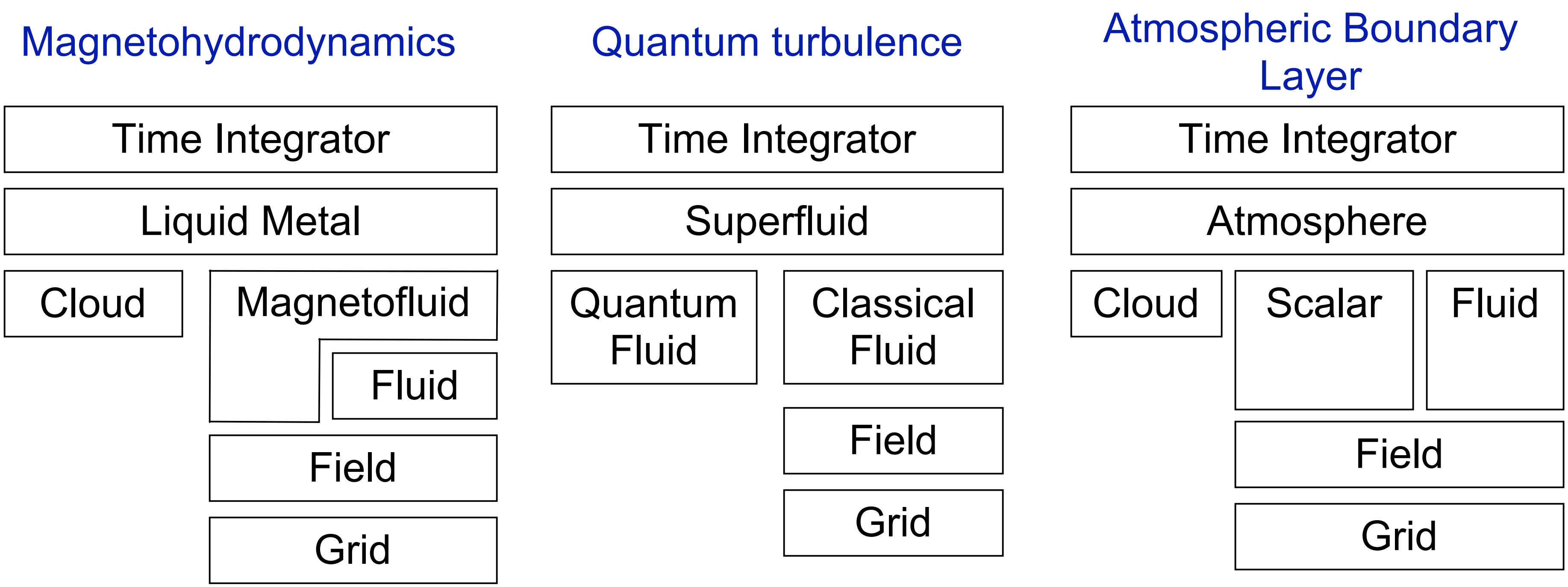
which is the minimum information growth.

Sandia National Laboratories

Outline

1. Analyses
2. Multiphysics model demonstrations
   a. Particle transport in magnetohydrodynamics.
   b. Quantum turbulence in superfluid ^{4}He.
   c. Atmospheric boundary layer.
   d. Lattice-Boltzmann bio-fluid dynamics.
3. Conclusions & Future Work

Morfeus



Lattice Boltzmann bio-fluid dynamics:
Xu & Lee (2008) “Application of the lattice Boltzmann method to flow in aneurysm with ring-shaped stent obstacles,” *Int. J. Numerical Methods in Fluids.*

Sandia National Laboratories

Particle-Laden MHD

Strong magnetic fields damp velocity variations in the field direction, leading to 2D/3C state:

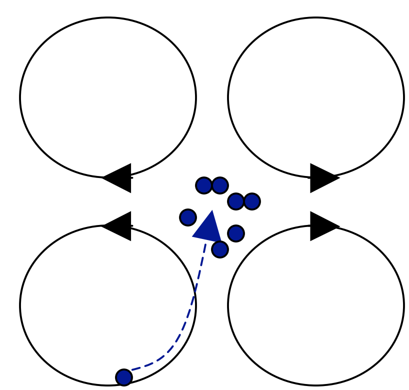
$$\frac{\partial}{\partial t}\vec{u}(\vec{x},t) = \ldots + \frac{1}{\eta}\nabla^{-2}\left(\vec{B}_A^{ext}\cdot\nabla\right)^2\vec{u}(\vec{x},t) + \ldots$$

Cross-stream dispersion segregates inertial particles:

$$dr_i / dt = \mathrm{v}_i$$

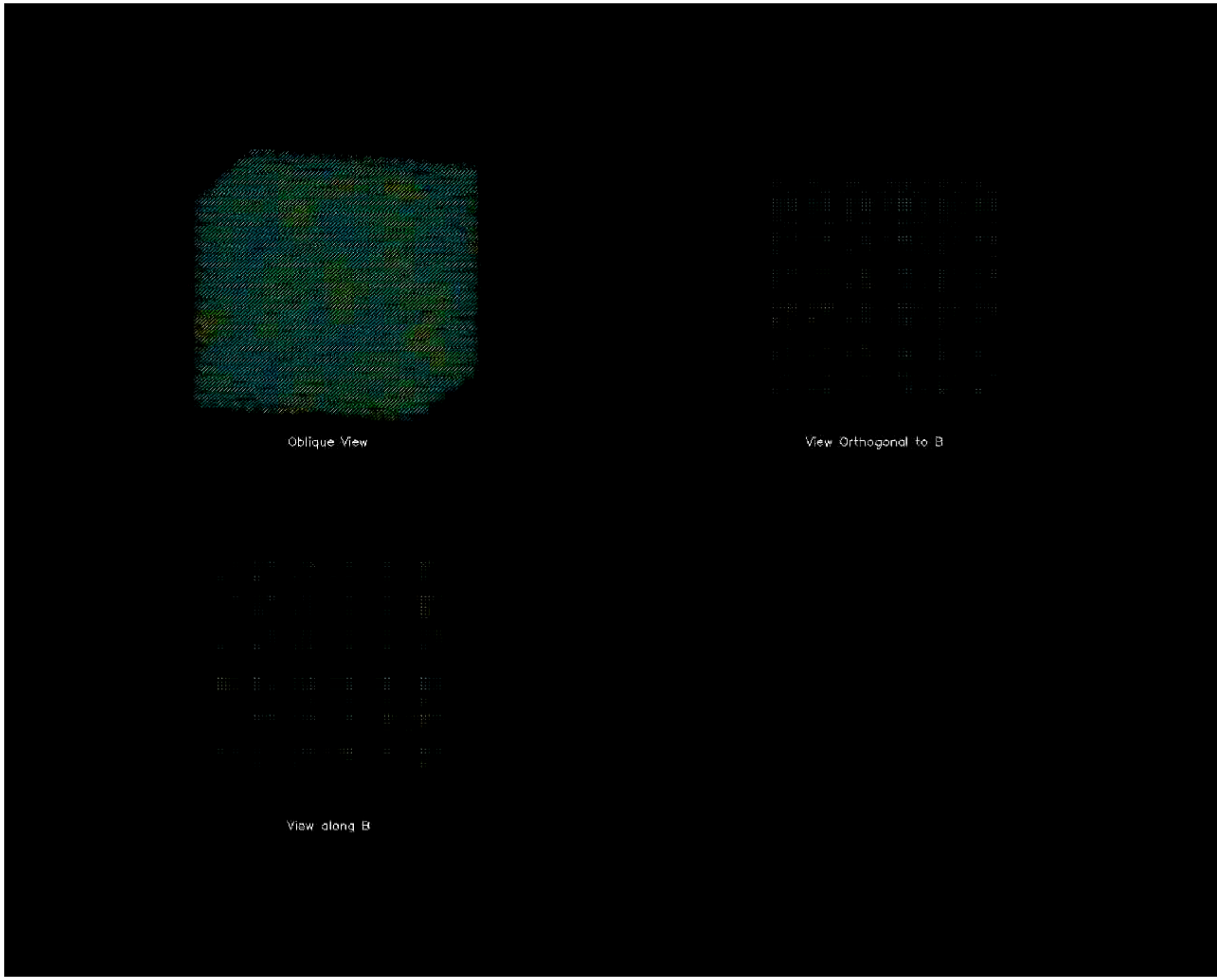
$$d\mathrm{v}_i / dt = \left[u_i(\vec{r},t) - \mathrm{v}_i\right] / St$$

$$St \equiv \tau_p / \tau_f$$



Sandia National Laboratories

Particle-Laden MHD



Rouson et al. (2008), "Dispersed-phase structural anisotropy in magnetohydrodynamic turbulence at low magnetic Reynolds numbers," *Physics of Fluids*.

Quantum Turbulence

Below 2.17 K, liquid ^{4}He flows as a two-fluid mixture with mutual friction between the two components:

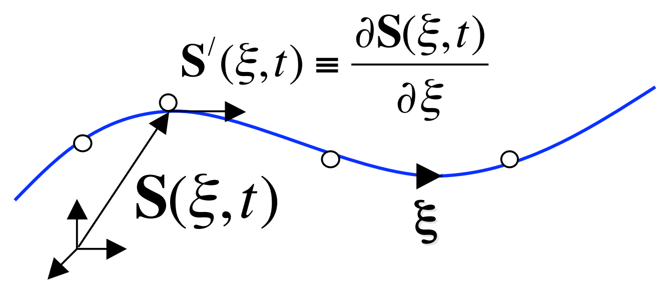
1. Normal viscous fluid

$$\frac{\partial}{\partial t}\vec{u} + \vec{u}\cdot\nabla\vec{u} = -\frac{1}{\rho}\nabla p + \nu\nabla^2\vec{u} + \vec{f}$$

$$\nabla\cdot\vec{u} = 0$$

2. Inviscid superfluid with quantized vortex circulation

$$\kappa \equiv \hbar / m_{He}$$



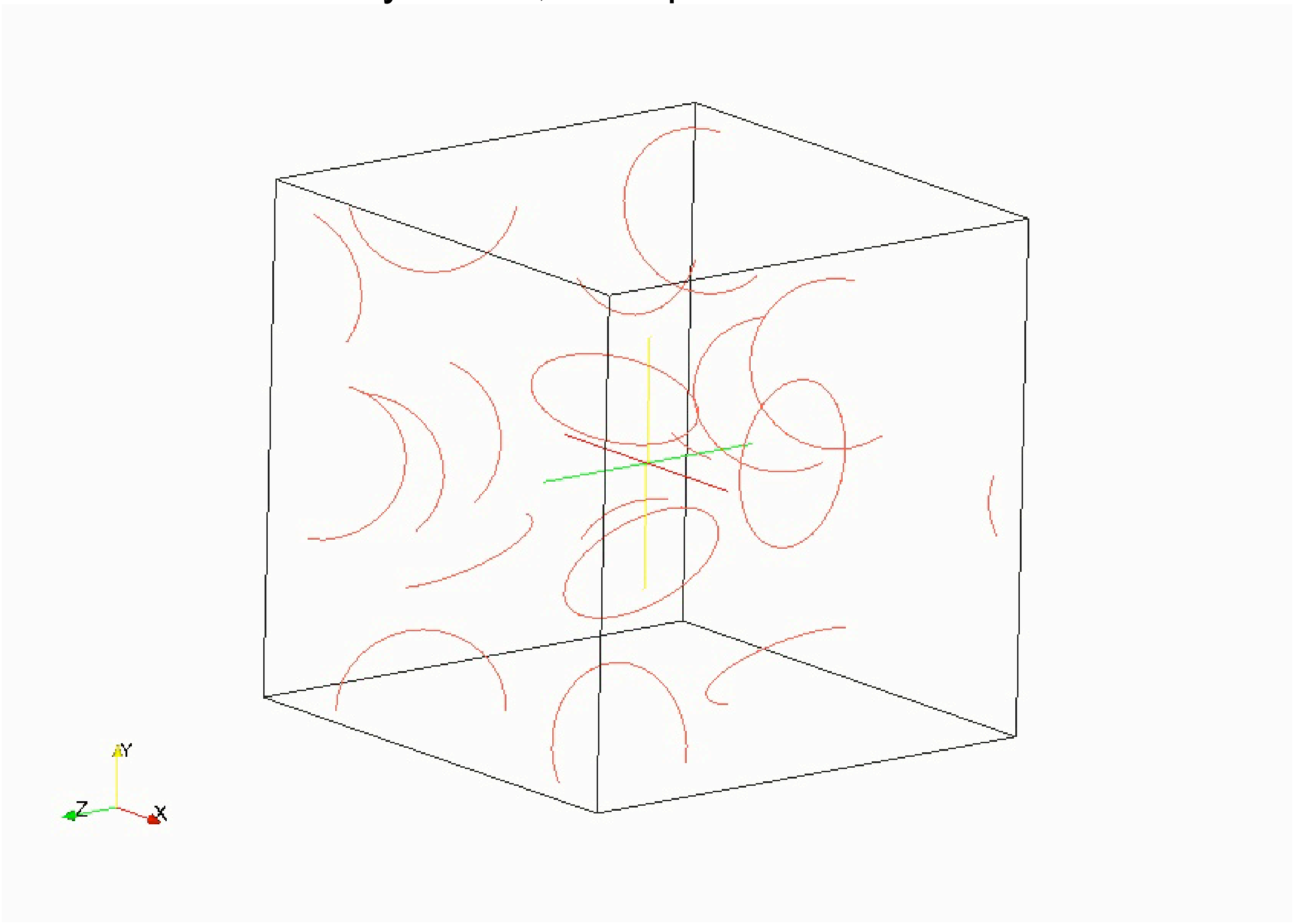
$$\mathbf{v}_i = \frac{\kappa}{4\pi}\int (\mathbf{S}_0 - \mathbf{r}) \otimes d\mathbf{S}_0 / \|\mathbf{S}-\mathbf{r}\|^3$$

$$\frac{d\mathbf{S}}{dt} = \mathbf{v}_i + \alpha\mathbf{S}' \otimes (\mathbf{v}_n - \mathbf{v}_i) - \alpha'\mathbf{S}' \otimes [\mathbf{S}' \otimes (\mathbf{v}_n - \mathbf{v}_i)]$$

Sandia National Laboratories

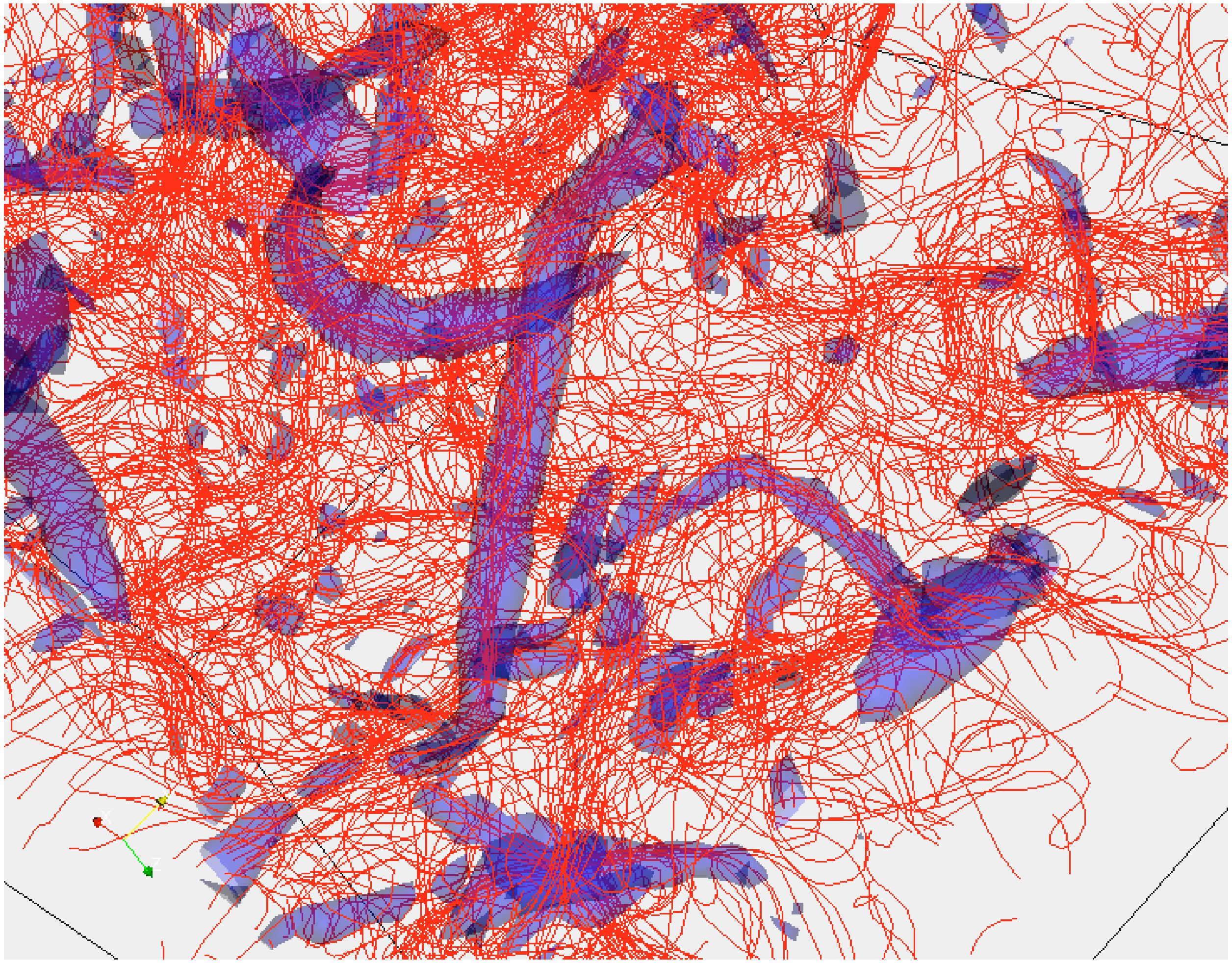
Quantum Turbulence

Quantum vortices driven by forced, isotropic normal-fluid turbulence at 2.1 K:



Vortex Locking

Quantum vortex alignment with classical vortices in frozen normal-fluid turbulence:



Morris, Koplik & Rouson (2008) “Vortex locking in direct numerical simulations of quantum turbulence,” *Phys. Rev. Lett.*

Sandia National Laboratories

Conclusions

- Applying Amdahl’s law to the *total* solution time suggests that optimizing runtime only severely limits speedup.
- The Pareto Principle determines the percentage of the code that can be focused strictly on programmability rather than runtime efficiency.
- ADT calculus renders bug search times very nearly scale-invariant and reduces interface information content.

Future Directions

- Demonstrate runtime scalability.
- Add empirical support for reductions in
  1. Fault localization time.
  2. Information entropy*.

*Kirk & Jenkins (2004) “Information theory-based software metrics and obfuscation.” *J. Systems & Software.*

“First they ignore you. Then they ridicule you. Then they fight you. Then you win.”

Mahatma Ghandi

Traditional Design Metrics

Structured programming:

- Source Lines of Code (SLOC)
- Cyclomatic complexity

Object-Oriented Programming:

- Afferent couplings (*Ca*): # packages that depend on a given one.
- Efferent couplings (*Ce*): # packages a given package depends on.
- Instability: $$I \equiv \frac{Ce}{Ce + Ca}$$

Sandia National Laboratories