

---

# Application Performance Profiling on the Cray XD1 using HPCToolkit

**John Mellor-Crummey<sup>1</sup>, Nathan Tallent<sup>1</sup>  
Michael Fagan<sup>1</sup>, and **Jan E. Odegard<sup>2</sup>****

**<sup>1</sup>Computer Science**

**<sup>2</sup>Computer & Information Technology Institute  
Rice University**



---

<http://hipersoft.cs.rice.edu/hpctoolkit/>

CITIA@20

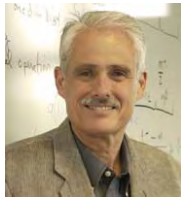
The image displays the logo for CITIA@20. The text 'CITIA@20' is rendered in a bold, blue, sans-serif font. A large, grey, five-pointed star is positioned behind the text, with its center overlapping the '@' symbol. The '@' symbol is white with a thin blue outline. The entire logo is set against a light grey rectangular background.

# Computer and Information Technology

To build a community of scholars that engages in collaborative research and education covering virtually every aspect of information technology and computing

## Directors:

Ken Kennedy (1986-1992)



Sidney Burrus (1992-1998)



Willy Zwaenepoel (1998-2001)



Moshe Vardi (2001-...)



6 schools  $\leftrightarrow$  20 departments  $\leftrightarrow$  140 members  
7 centers  $\leftrightarrow$  12+ ad hoc research groups

# Research Centers

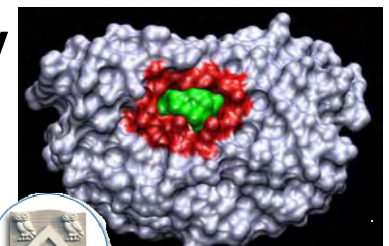
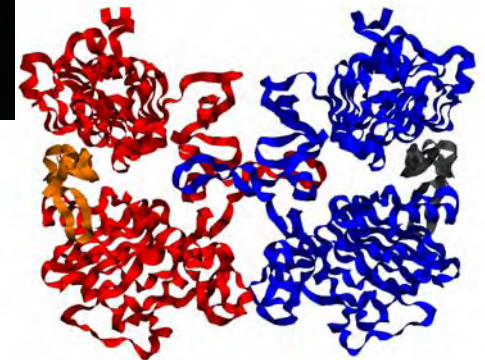
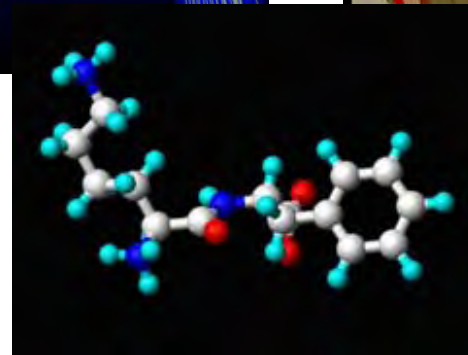
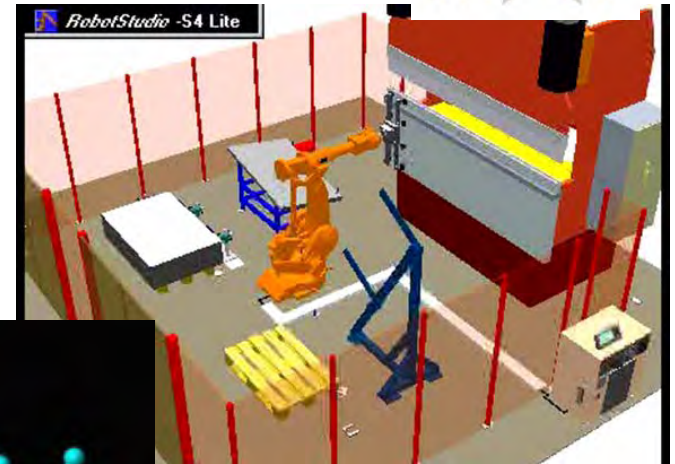
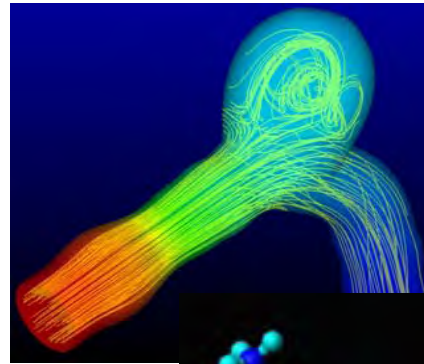


- **Center for High Performance Software (HiPerSoft)**  
— **Director: TBN**
- **Center for Multimedia Communication (CMC)**  
— **Director: Behnaam Aazhang, ECE**
- **Center for Computational Geophysics (CCG)**  
— **Co-directors: Bill Symes, CAAM / Alan Levander, ES**
- **Center for Computational Finance & Economic Systems (CoFES)**  
— **Director: Kathy Ensor, STAT**
- **LABoratory for NanoPhotronics (LANP)**  
— **Director: Naomi Halas, ECE**
- **Center for Technology in Teaching and Learning (CTTL)**  
— **Director: Tony Gorry, CS**
- **Center for Excellence and Equity in Education (CEEE)**  
— **Director: Richard Tapia, CAAM**

# Research Groups & Labs



- Gaming Group
- Robotics Group
- Sensor Nets Group
- Bioinformatics Group
- Rice Networking Group
- Digital Signal Processing
- Dynamical Systems Group
- Statistical Consulting Lab
- Rice Computer Architecture Group
- Complex Flow of Complex Fluids Group
- Theoretical and Computational Neuroscience
- Connexions: Open content education repository
- Advanced Research Initiative on the Emerging Library
- ...



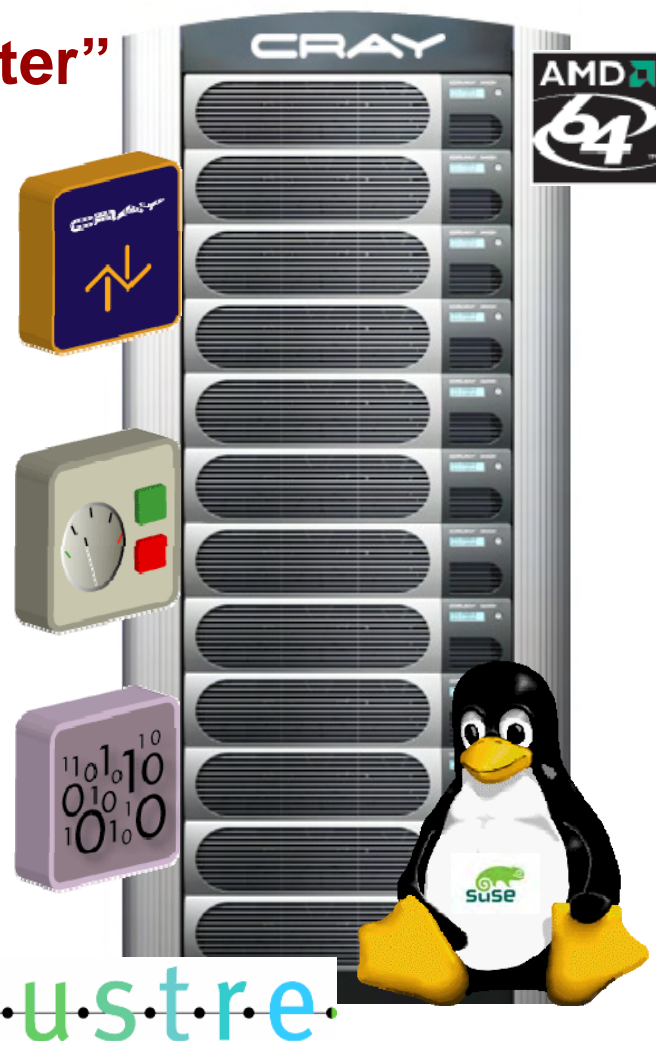
# Cray XD1 System

Dual-Core AMD Opteron™



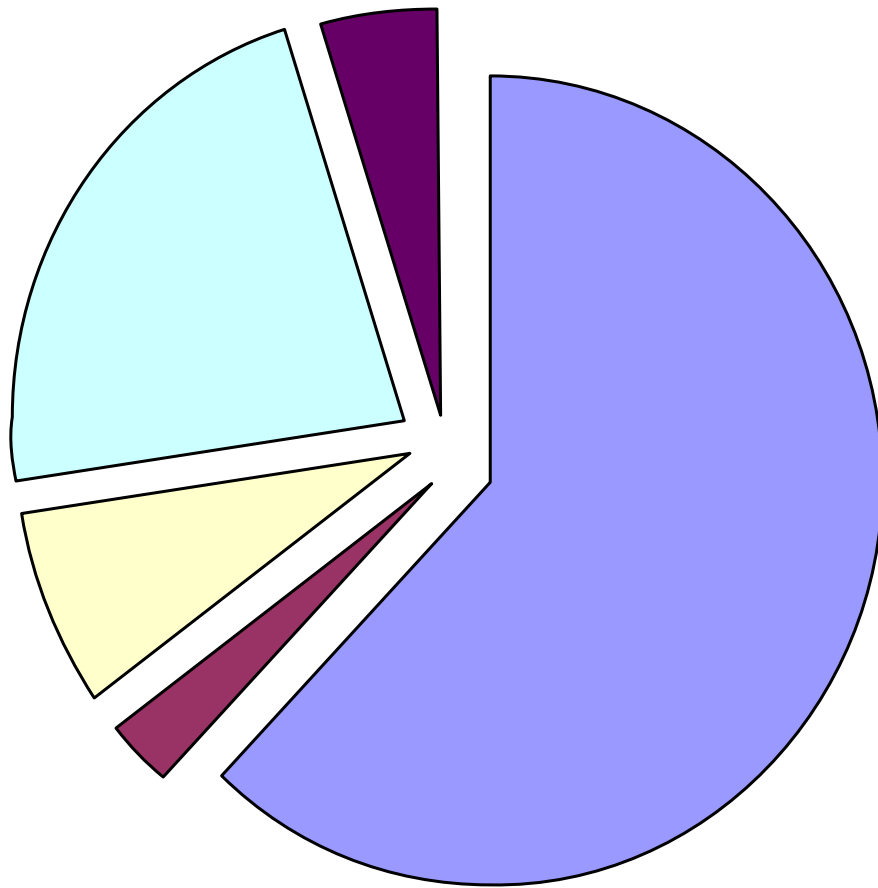
## “Rice Computational Research Cluster”

- ~3 TeraFLOP Cray XD1 Linux cluster\*
  - 336 Dual-Core AMD Opteron™ 275
    - 2.2GHz, 1MB / Core
    - 168 dual socket nodes (4 way SMP)
    - 8 GB DDR 400 / compute SMP
    - 16 GB DDR 333 / system SMP
  - Cray RapidArray (4x Infiniband)
  - 1.4 TB DDR2 400
  - 12 TB Local Disk
  - 6 TB Lustre parallel file system
  - 10 TB NFS file system
  - One XD1 Chassis with FPGA
    - 6 Xilinx Virtex-4/LX160



NSF MRI, Rice, AMD and Cray

# 250+ Active Users



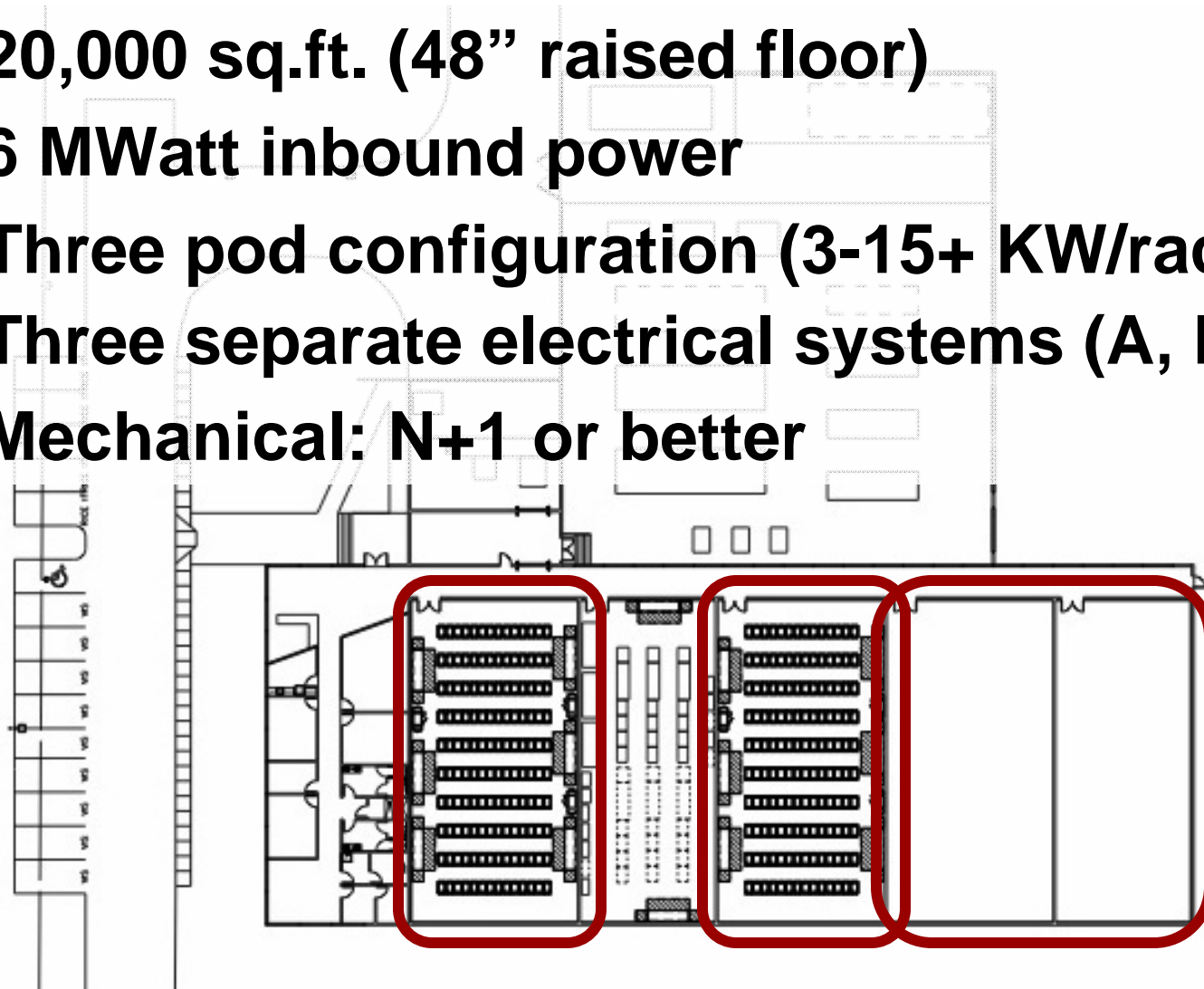
- Engineering
- Management
- CITI
- Natural Sciences
- Social Sciences



# New Datacenter (July 2007)



- **20,000 sq.ft. (48" raised floor)**
- **6 MWatt inbound power**
- **Three pod configuration (3-15+ KW/rack)**
- **Three separate electrical systems (A, B & C)**
- **Mechanical: N+1 or better**





# Datacenter (~12-06)



**CITI: Building communities since 1986**  
Rice University, Houston, Texas



# Datacenter (~12-06)



**CITI: Building communities since 1986**  
Rice University, Houston, Texas



# The Challenge

---

## Getting Science Done

To achieve acceptable (top) application performance scientists and engineers are required to tailor applications to effectively exploit the capabilities of a “bewildering” array of features offered by current and future architectures

# Performance Analysis and Tuning

---

- **Increasingly necessary**
  - gap between typical and peak performance is growing
- **Increasingly hard**
  - complex architectures are harder to program effectively
    - complex processors
      - deeply pipelined, out of order, superscalar
    - complex memory hierarchy
      - non-blocking, multi-level caches, TLB, hw prefetching
  - modern scientific applications pose challenges for tools
    - multi-lingual programs
    - many source files
    - complex build process
    - external libraries in binary-only form



# HPCToolkit Goals

---

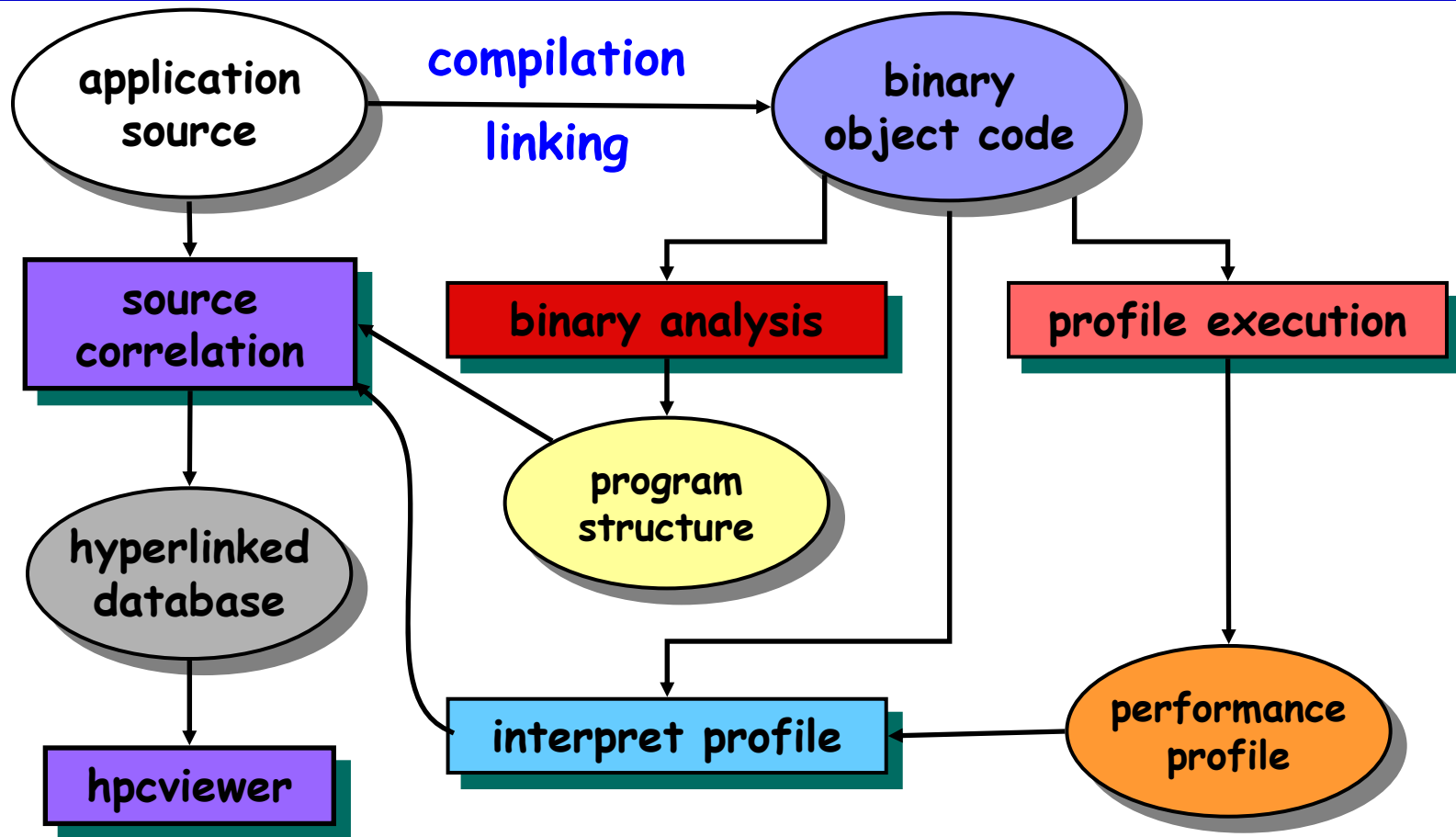
- **Support large, multi-lingual applications**
  - a mix of of Fortran, C, C++
  - multiple programming models (MPI, OpenMP, multi-threading)
  - external libraries
  - hundreds of procedures
  - for ease of use, avoid
    - manual instrumentation
    - significantly altering the build process
    - frequent recompilation
- **Analysis of both serial and parallel codes**
- **Scalable data collection for parallel executions**
- **Effective presentation of analysis results**
  - intuitive enough for scientists and engineers to use
  - detailed enough to meet the needs of compiler writers

# HPCToolkit Design Principles

---

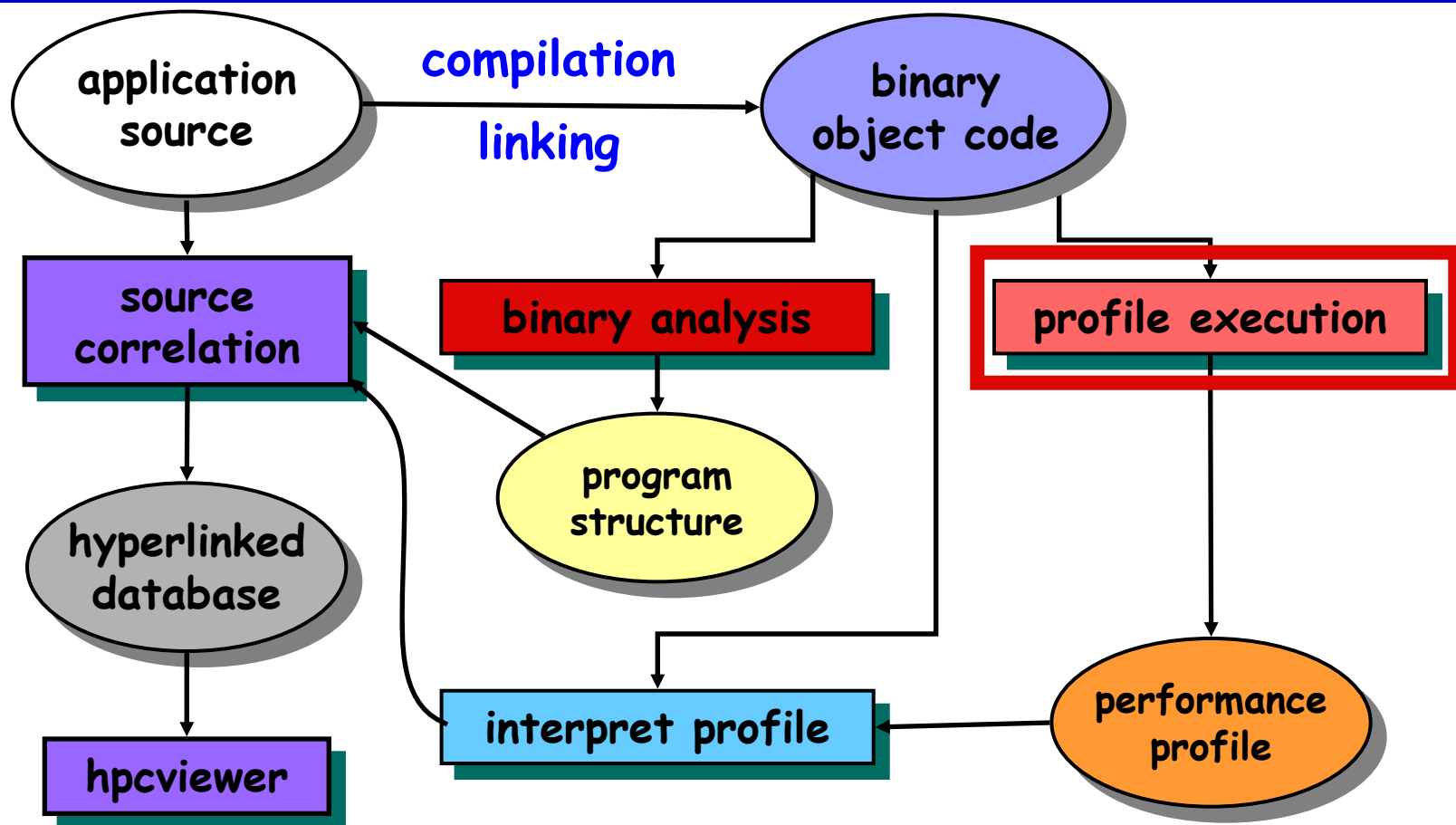
- **Language independence: work at the binary level**
  - supports multi-lingual codes with external binary-only libraries
- **Avoid code instrumentation in each procedure**
  - instrumentation adds overhead and distorts measurements
- **Context is essential for understanding modern software**
  - modular software often depends on layered libraries (e.g. MPI)
- **Any one performance measure produces a myopic view**
  - hard to diagnose a problem with only one species of event
- **Derived metrics are essential for effective analysis**
- **Performance analysis should be top down**
- **Event aggregation for loops and procedures is important**
  - accurate despite approximate event attribution from counters
  - loop-level info is more important than line-level info

# HPCToolkit Workflow



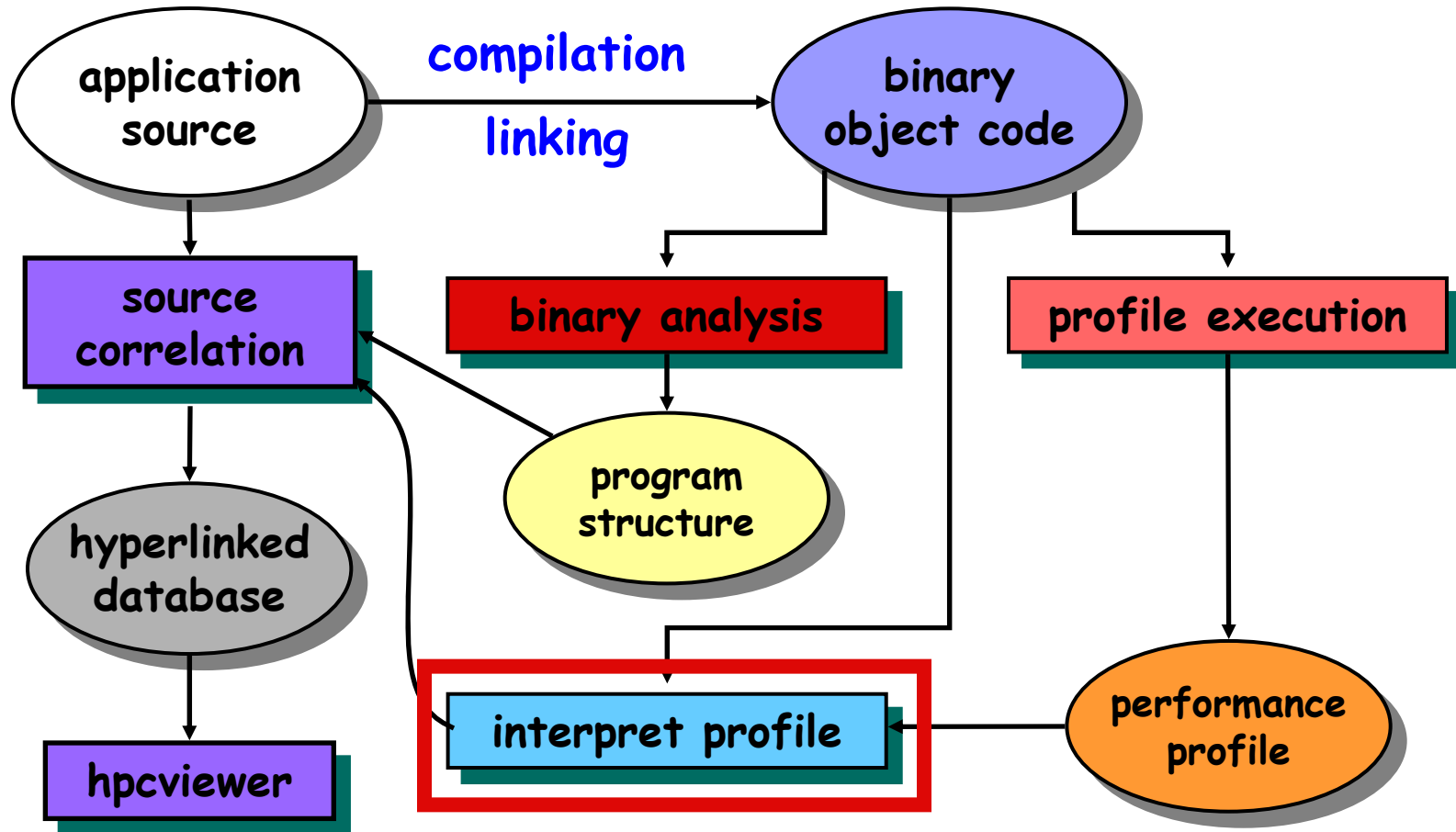


# HPCToolkit Workflow



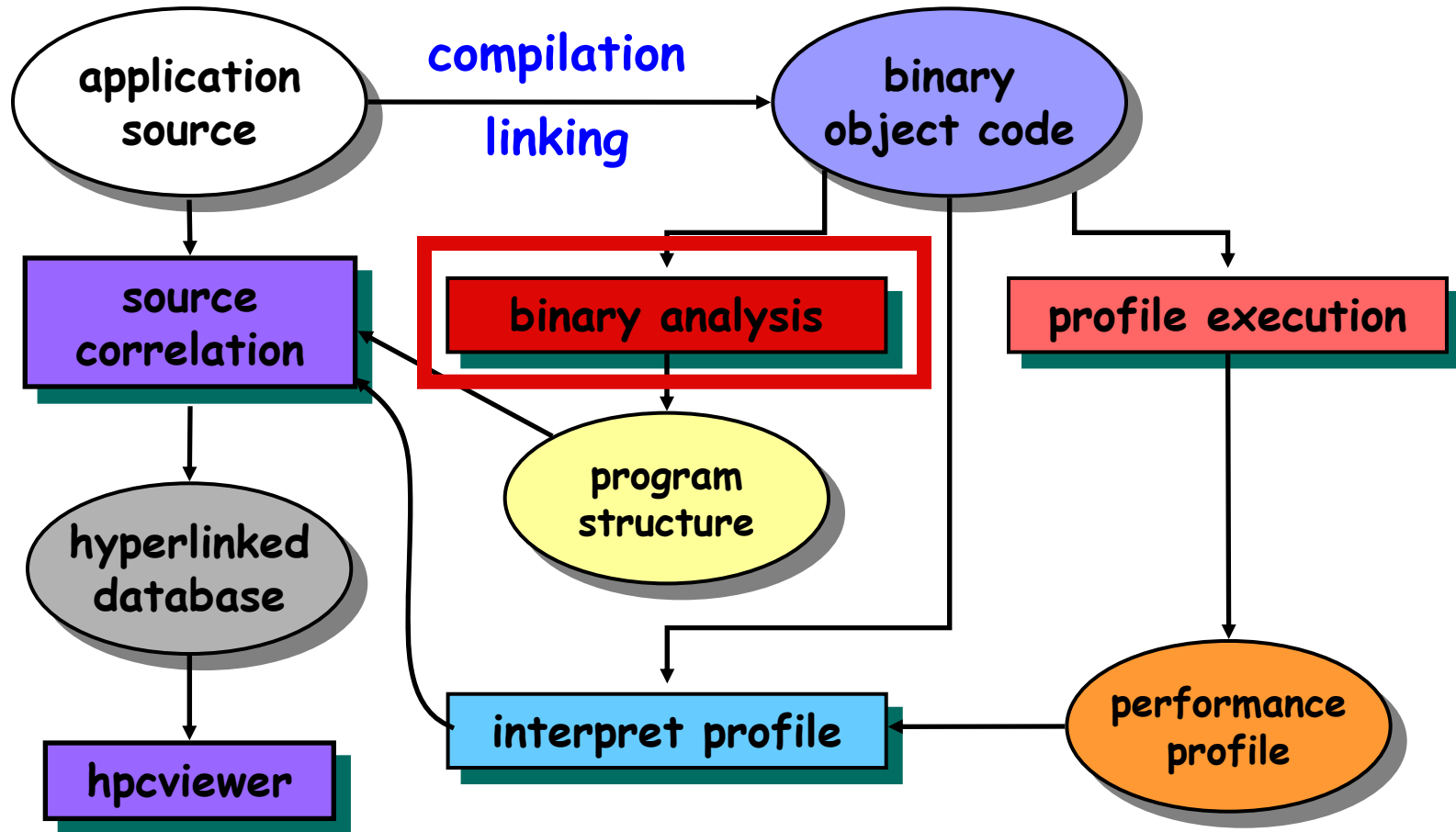
- launch unmodified, optimized application binaries
- collect statistical profiles of events of interest

# HPCToolkit Workflow



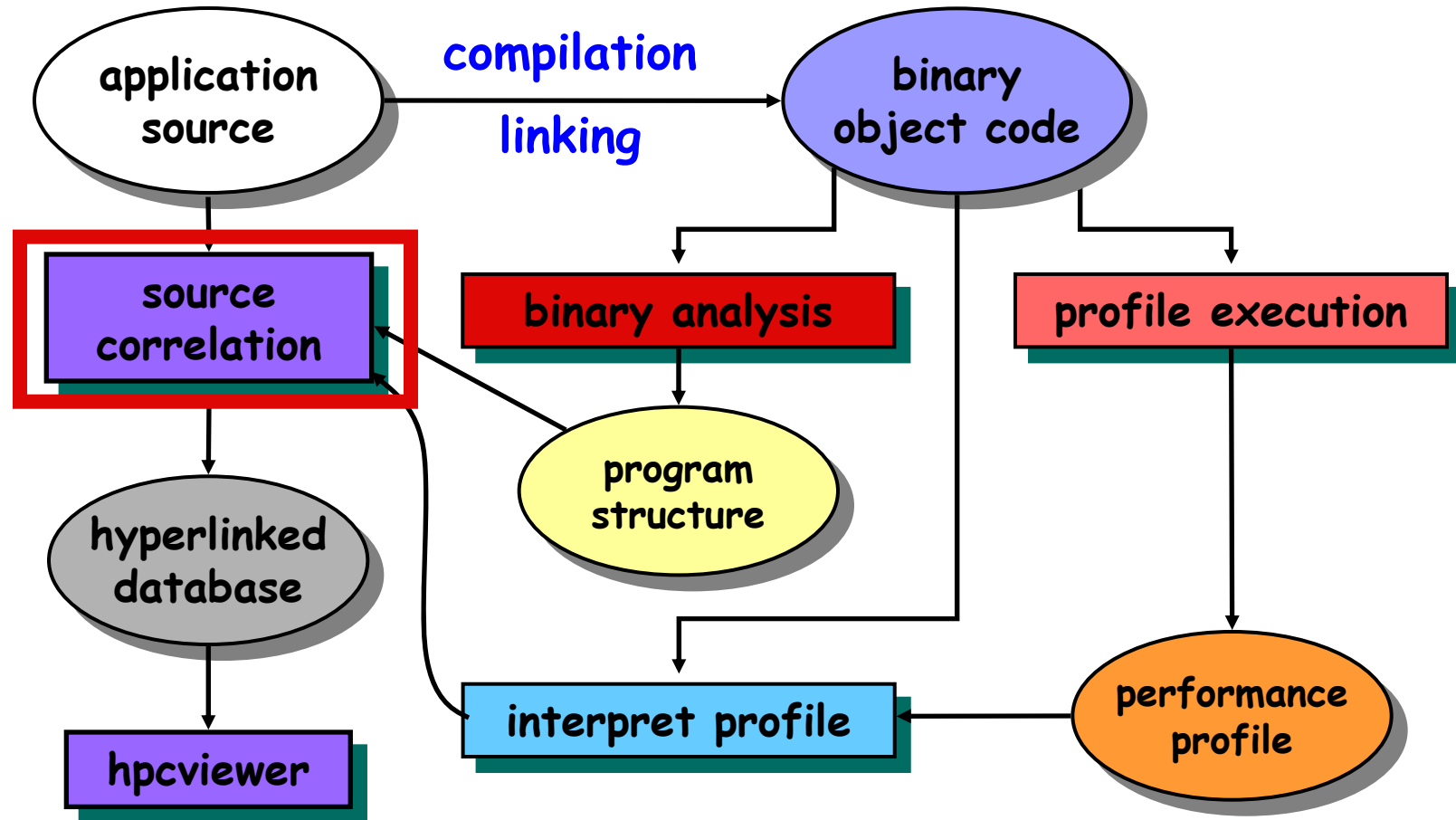
—decode instructions and combine with profile data

# HPCToolkit Workflow



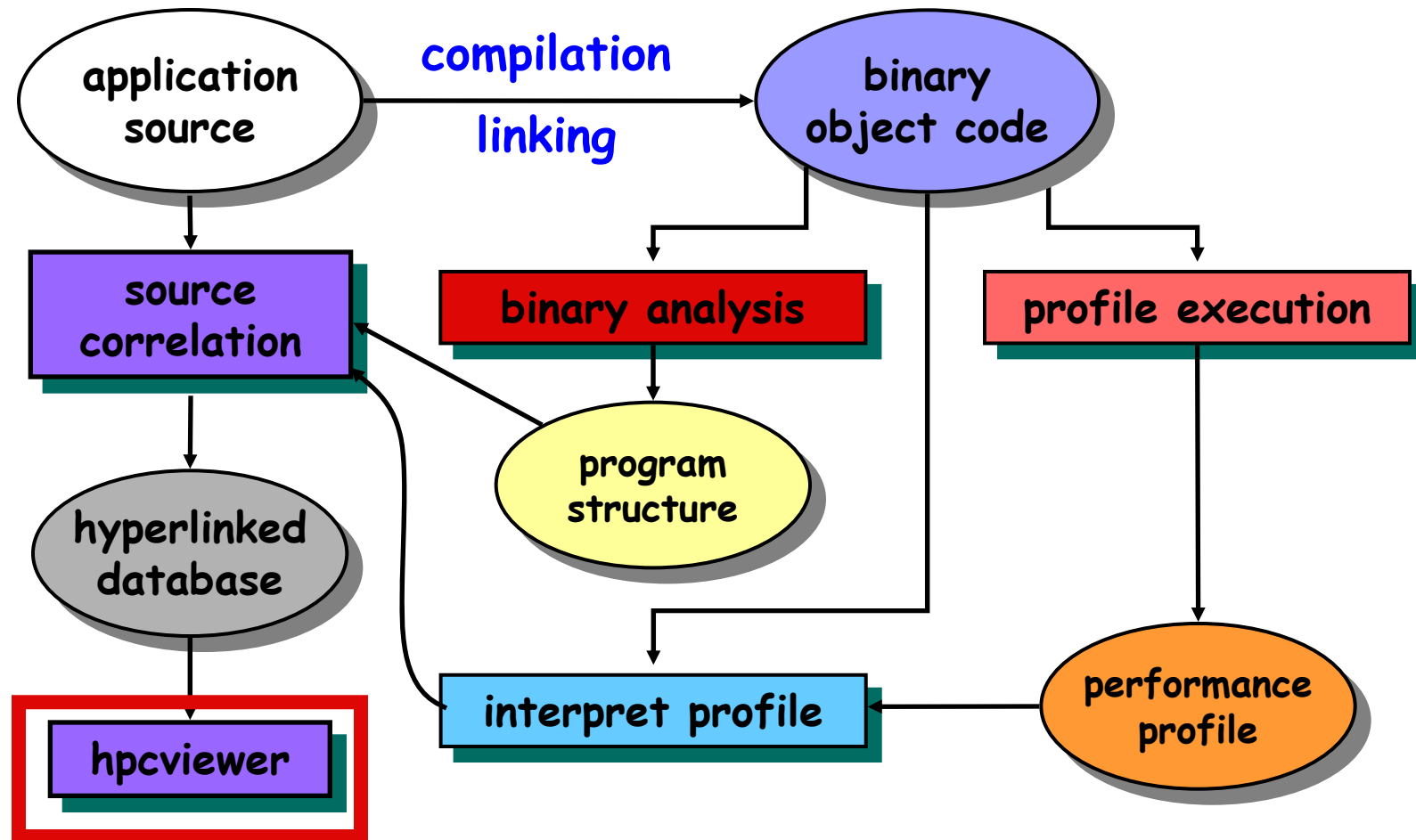
—extract loop nesting & inlining from executables

# HPCToolkit Workflow



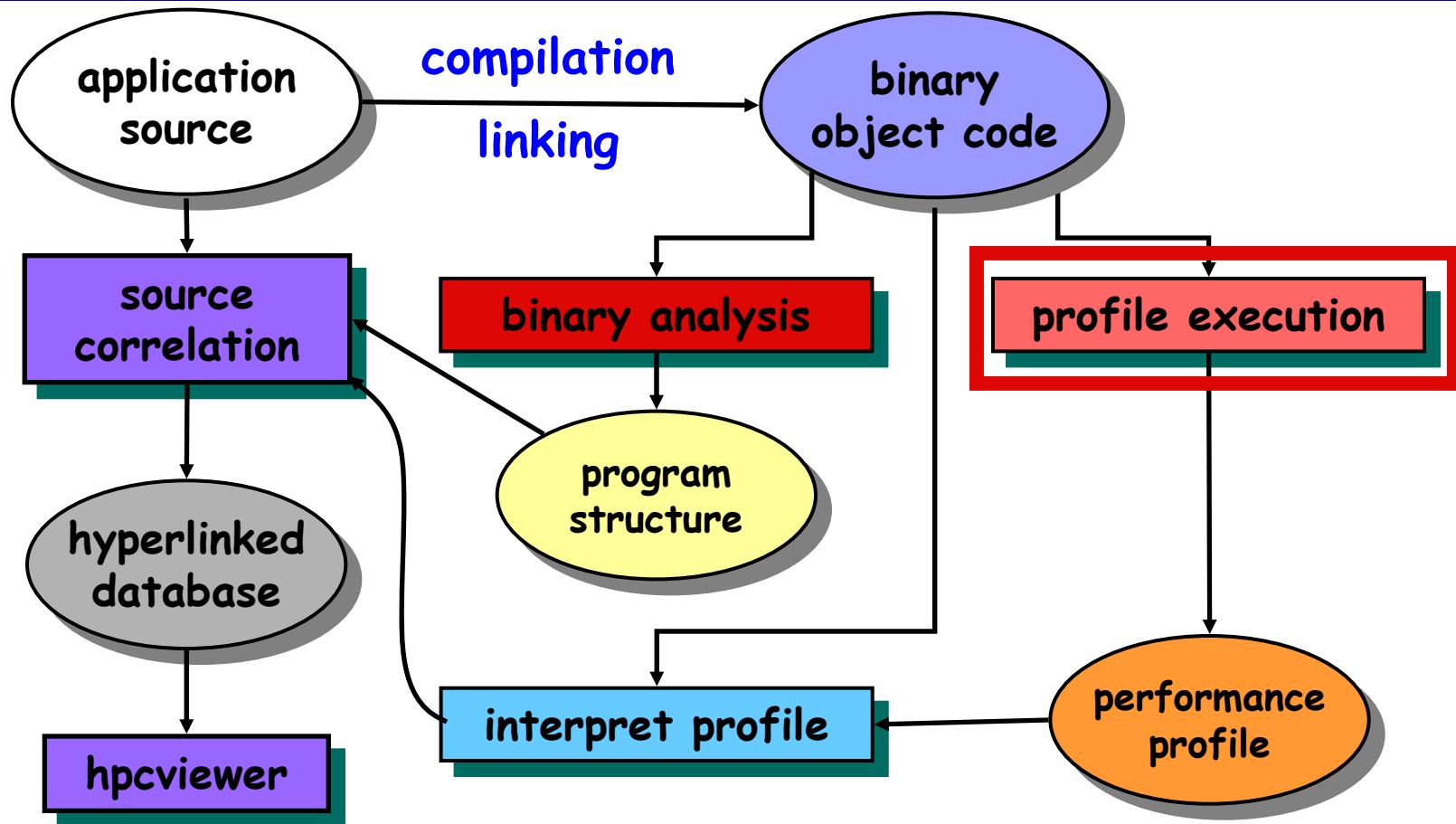
- synthesize new metrics by combining metrics
- relate metrics and structure to program source

# HPCToolkit Workflow



- support top-down analysis with interactive viewer
- analyze results anytime, anywhere

# HPCToolkit System Overview



# Data Collection

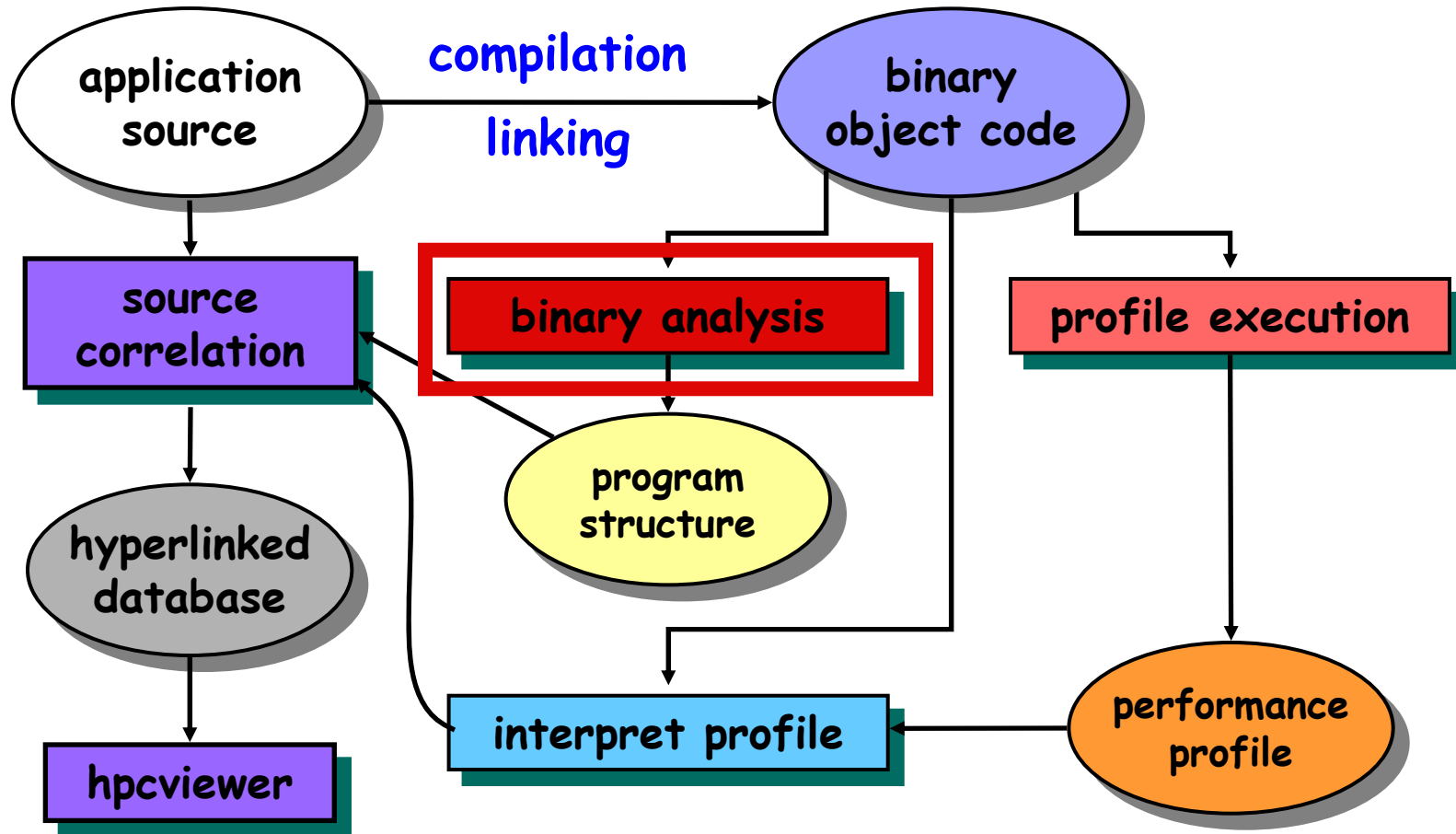
---

## Support analysis of unmodified, optimized binaries

- Use statistical sampling to profile events
  - hardware performance counter overflows
  - interval timer events
- Tools
  - **hpcrun**: flat sampling yields PC histograms
  - **csprof**: attributes samples to calling context



# HPCToolkit System Overview



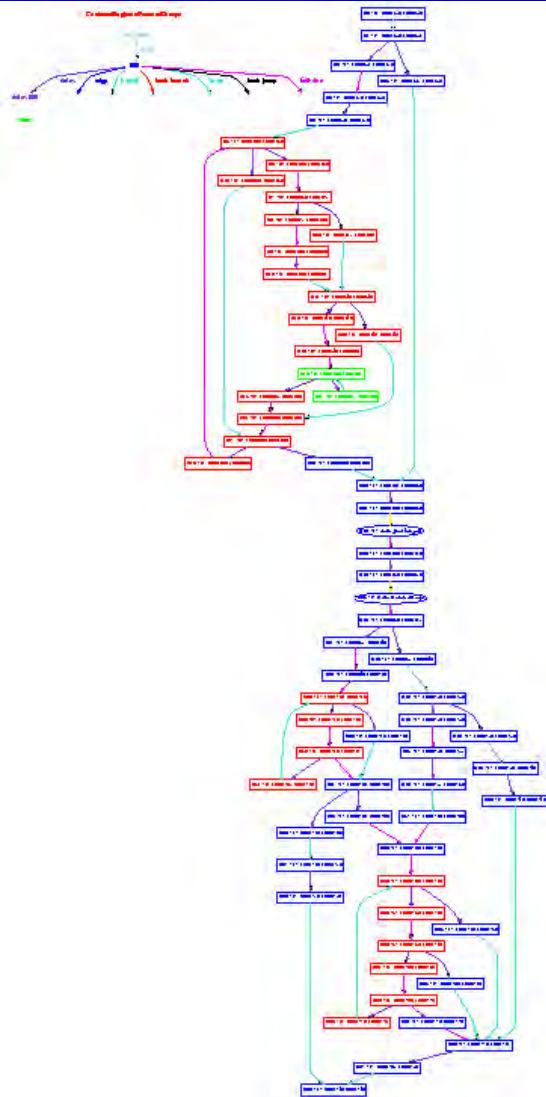
# Program Structure Recovery with **bloop**

---

## Analyze an application binary

- Construct control flow graph from branches
- Identify natural loop nests using interval analysis
- Map instructions to source lines, procedures
  - leverage line map + DWARF debugging information
- Discover inlined code
- Normalize output to recover source-level view

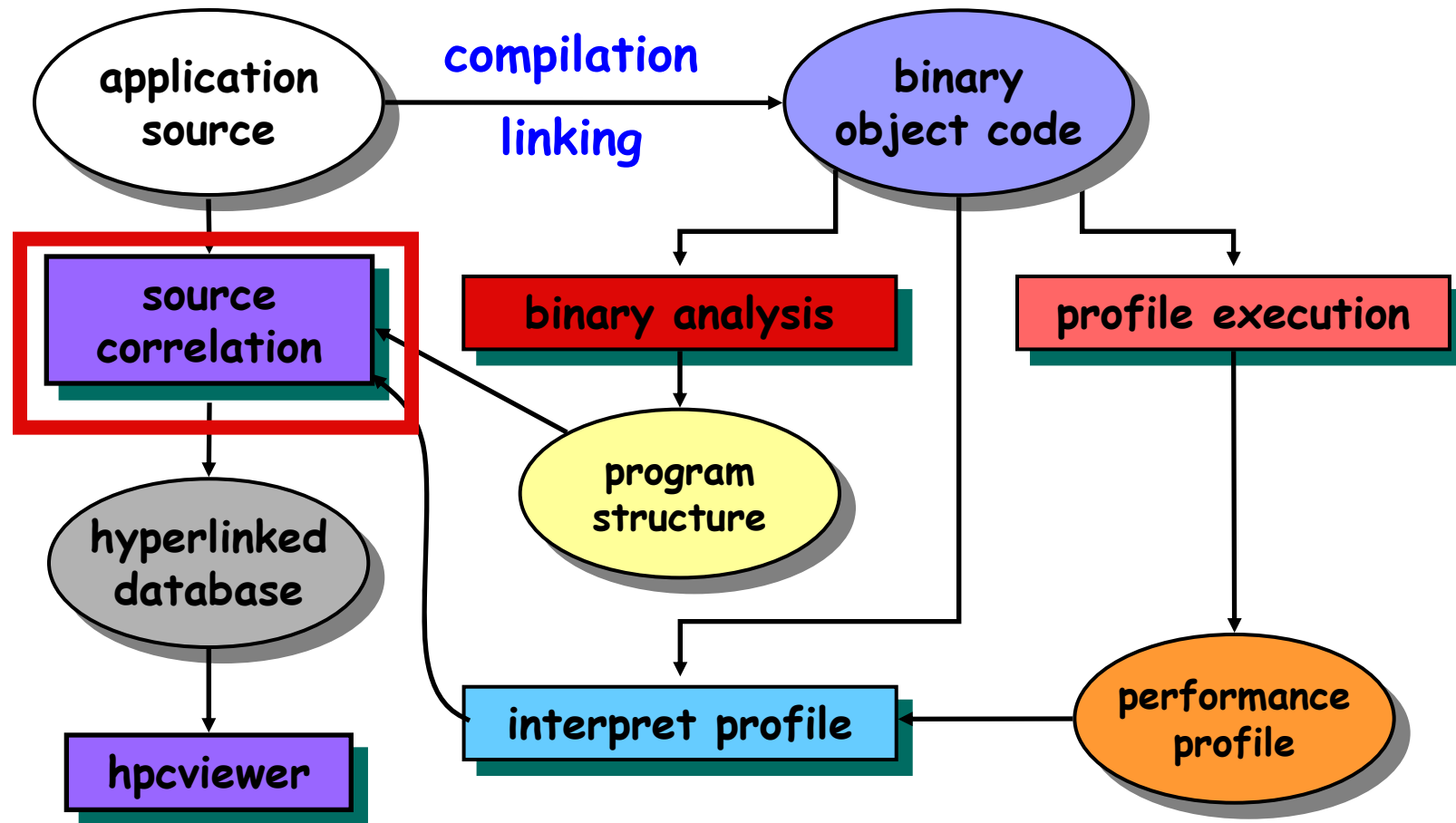
# Sample Flowgraph from an Executable



- Loop nesting structure
- blue: outermost level
- red: loop level 1
- green loop level 2

**Observation**  
**optimization complicates**  
**program structure!**

# HPCToolkit System Overview



# Data Correlation

---

- **Problem**

- any one performance measure provides a myopic view

- some measure potential *causes* (e.g. cache misses)
    - some measure *effects* (e.g. cycles)
    - cache misses not always a problem

- event counter attribution is often inaccurate

- **Approaches**

- multiple metrics for each program line

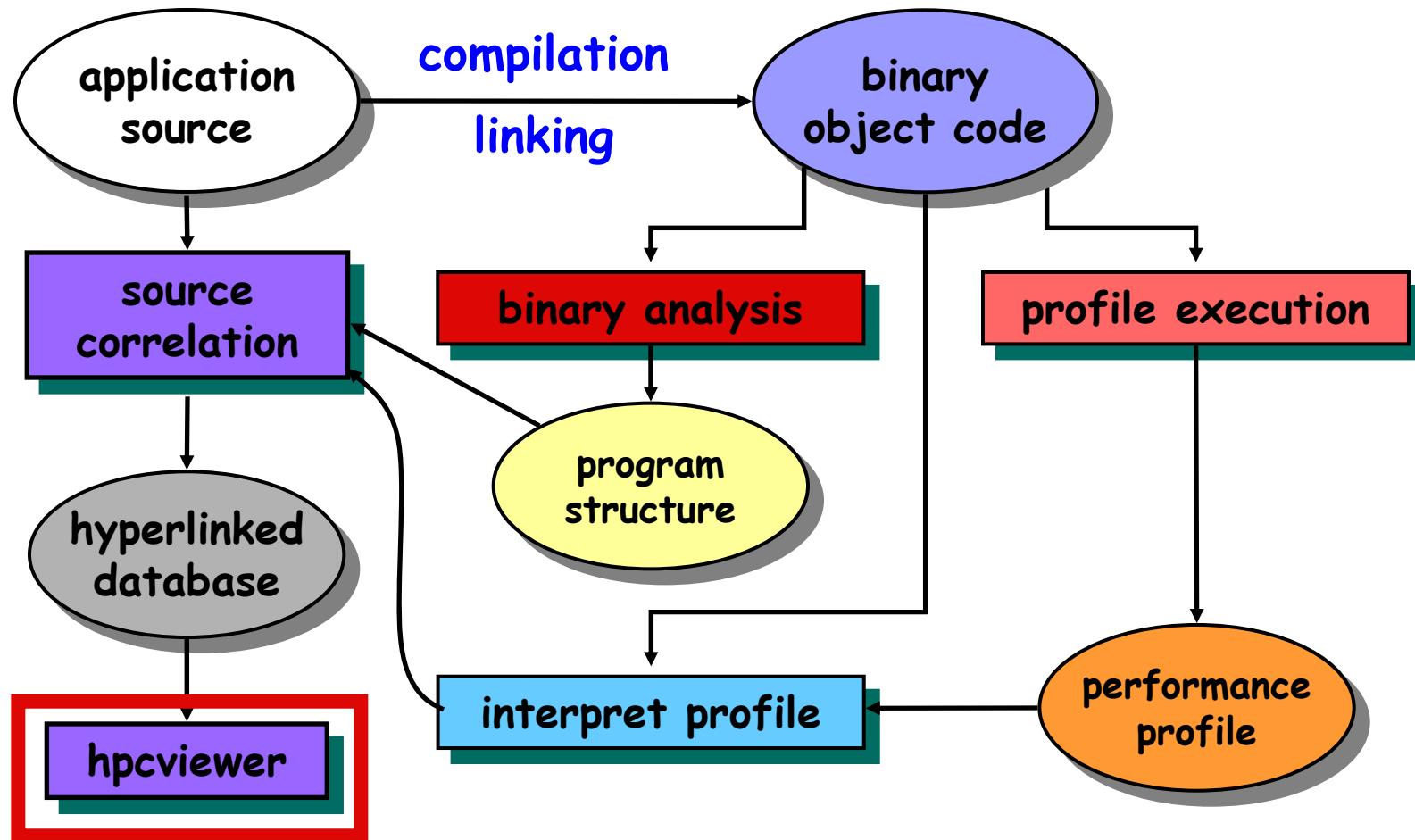
- computed metrics (e.g. waste = peak FLOPs - actual FLOPS)

- eliminates mental arithmetic
    - serves as a key for sorting

- hierarchical structure

- errors with line level attribution still yield good loop-level information

# HPCToolkit System Overview



# hpcviewer User Interface

The screenshot displays the hpcviewer interface with several key components highlighted by red boxes and arrows:

- source pane:** Located at the top, it shows the source code for `hmc.cc`. The `main` function is highlighted in yellow.
- flatten/zoom control:** A red box with an arrow pointing to the zoom controls in the source pane.
- view control:** A red box with an arrow pointing to the view selection buttons: `Calling Context View`, `Callers View`, and `Flat View`.
- navigation pane:** A red box with an arrow pointing to the `Scopes` pane, which shows a tree view of the execution context, including `main`, `void Chroma::doHMC`, and various loops.
- metric pane:** A red box with an arrow pointing to the `metric pane`, which displays a table of performance metrics.

# samples (I)	# samples (E)
5.80e05	97.5%
5.70e05	95.8%
5.65e05	95.0%
5.65e05	95.0%
5.25e05	88.2%
4.25e05	71.4%
3.30e05	55.5%



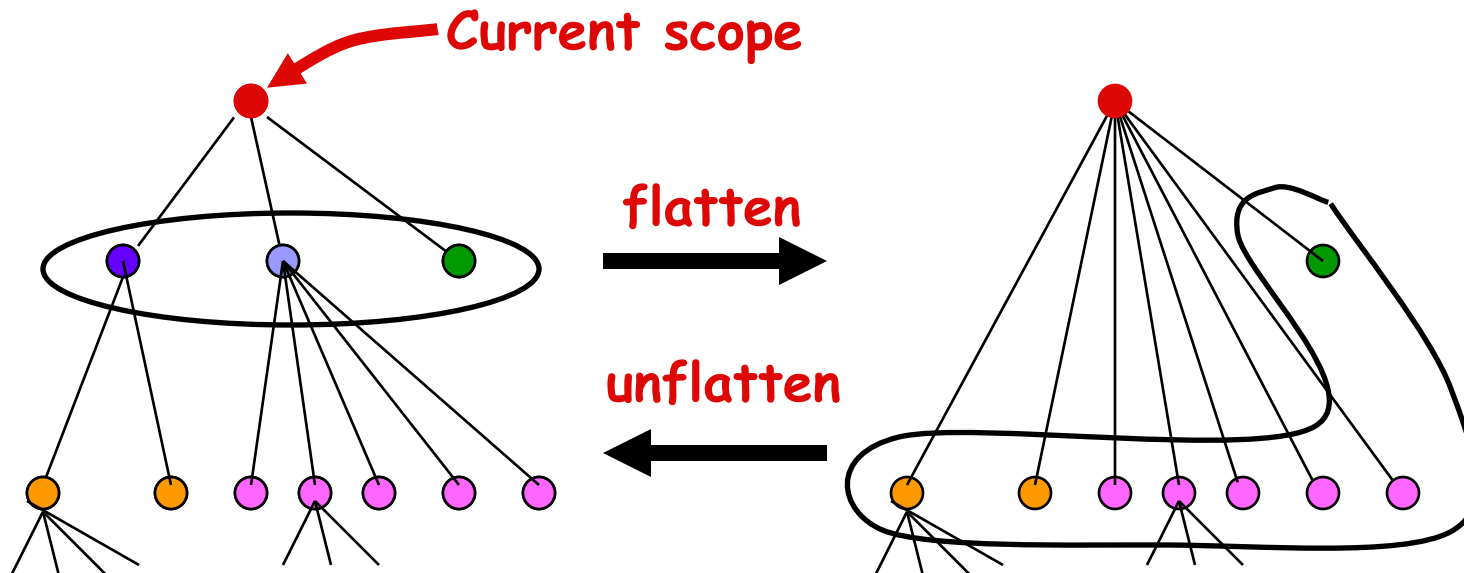
# Principal **hpcviewer** Views

---

- **Calling context tree view**
  - top-down* view shows dynamic *calling contexts* in which costs were incurred
- **Caller's view**
  - bottom-up* view apportions costs incurred in a routine to the routine's dynamic calling contexts
- **Flat view**
  - aggregates all costs incurred by a routine *in any context* and shows the details of where they were incurred within the routine

# Flattening Static Hierarchies

- **Problem**
  - hierarchical view of a program is too rigid
  - sometimes want to compare children of different parents
    - e.g. compare all loops, regardless of the routine they are inside
- **Solution**
  - flattening elides a scope and shows its children instead



# Chroma Lattice QCD Library

calling context view

```
cdp_parscalar_specific.h
77 {
78   QMP_sum_float_array(cest, len);
79 }
80
81 //! Low level hook to QMP_global_sum
82 inline void globalSumArray(double *dest, int len)
83 {
84   QMP_sum_double_array(dest, len);
85 }
86
87 //! Global sum on a multi1d
88 template<class T>
89 inline void globalSumArray(multi1d<T>& dest)
90 {
```

- costs for loops in CCT
- costs for inlined procedure
- inclusive and exclusive costs

Calling Context View Callers View Flat View

Scopes

	# samples (I)	%	# samples (F)	%
Chroma::TwoFlavorExactWilsonTypeFermMonomial<QDP::multi1d<QDP::OLattice<QDP::PScalar<QDP::PColorMatrix>>>	2.30e05	36.7%		
Chroma::MdagMSysSolverCG<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	2.20e05	37.0%		
loop at syssolver_mdagm_cg.h: 66-70	2.20e05	37.0%		
Chroma::SystemSolverResults_t Chroma::InvCG2<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	2.20e05	37.0%		
Chroma::SystemSolverResults_t Chroma::InvCG2_a<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	2.20e05	37.0%	1.00e04	1.7%
loop at invcg2.cc: 147-182	1.85e05	31.1%	5.00e03	0.8%
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	1.05e05	17.6%	1.00e04	1.7%
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	7.00e04	11.8%	1.50e04	2.5%
[I] globalSumArray	5.00e03	0.8%		
[I] vaxpy3	5.00e03	0.8%	5.00e03	0.8%
[I] local_sumsq				
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	2.00e04	3.4%		
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>, 3>>>	5.00e03	0.8%		

# Chroma Lattice QCD Library

caller's view

```
686 /* the basic operations in this routine include loading the halfspinor
687 * from memory, multiplying it by the appropriate gauge field, doing the
688 * spin reconstruction, and summing over directions, and saving the partial
689 * sum over directions */
690
691 void mvv_recons_plus(size_t lo,size_t hi, int id, const void *ptr)
692 {
693     DECL_COMMON_ALIASES_TEMPS;
694
695     const Arg_s *a =(Arg_s *)ptr;
696     int low = (int)lo;
697     int high = (int)hi;
698
699     MY_SPINOR* spinor_field = a->spinfun;
700
701     MY_SSE_VECTOR* chla = a->chlfun; /* a 1-d map of a 2-d array */
```

Calling Context View   Callers View   Flat View

Scopes

	# samples (I)	%	# samples (E)	%
mvv_recons_plus	5.50e04	9.2%	5.50e04	9.2%
sse_su3dslash_wilson	5.50e04	9.2%	5.50e04	9.2%
Chroma::SSEWilsonDslash::apply(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::RComplex<float>>>>>	5.50e04	9.2%	5.50e04	9.2%
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::FComplex<float>>>>>	3.50e04	5.9%	3.50e04	5.9%
Chroma::SystemSolverResults_t Chroma::InvCG2_a<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::FComplex<float>>>>>	3.00e04	5.0%	3.00e04	5.0%
Chroma::SystemSolverResults_t Chroma::InvCG2_a<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::FComplex<float>>>>>	5.00e03	0.8%	5.00e03	0.8%
Chroma::EvenOddPrecWilsonLinOp::operator()(QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::FComplex<float>>>>>	2.00e04	3.4%	2.00e04	3.4%
Chroma::SystemSolverResults_t Chroma::InvCG2_a<QDP::OLattice<QDP::PSpinVector<QDP::PColorVector<QDP::FComplex<float>>>>>	1.50e04	2.5%	1.50e04	2.5%
Chroma::TwoFlavorExactWilsonTypeFermMonomial<QDP::multi1d<QDP::OLattice<QDP::PScalar<QDP::PColorVector<QDP::FComplex<float>>>>>	5.00e03	0.8%	5.00e03	0.8%
decomp_hvv_plus	5.00e04	8.4%	5.00e04	8.4%

show attribution of procedure costs to calling contexts



# S3D Solver for Turbulent, Reacting Flows

flat view

```
734 diffFlux(:,:,:,n_spec,:) = 0.0
735 DIRECTION: do m=1,3
736 SPECIES: do n=1,n_spec-1
737
738 if (baro_switch) then
739 ! driving force includes gradient in mole fraction and baro-diffusion:
740 diffFlux(:,:,:,n,m) = - Ds_mixavg(:,:,:,n) * ( grad_Ys(:,:,:,n,m) &
741 + Ys(:,:,:,n) * ( grad_mixMW(:,:,:,m) &
742 + (1 - molwt(n)*avmolwt) * grad_P(:,:,:,m)/Pre:
743
744 else
745 ! driving force is just the gradient in mole fraction:
746 diffFlux(:,:,:,n,m) = - Ds_mixavg(:,:,:,n) * ( grad_Ys(:,:,:,n,m)
747 + Ys(:,:,:,n) * grad_mixMW(:,:,:,m) )
748
749 endif
750 ! Add thermal diffusion:
751 if (thermDiff_switch) then
752 diffFlux(:,:,:,n,m) = diffFlux(:,:,:,n,m) &
753 - Ds_mixavg(:,:,:,n) * Rs_therm_diff(:,:,:,n) * molwt(n) &
754 * avmolwt * grad_T(:,:,:,m) / Temp
755 endif
```

Calling Context View Callers View Flat View

Scopes

- loop at mixavg\_transport\_m.f90: 735-760
  - loop at mixavg\_transport\_m.f90: 736-758
    - loop at mixavg\_transport\_m.f90: 745
    - loop at mixavg\_transport\_m.f90: 758
    - loop at mixavg\_transport\_m.f90: 740
    - loop at mixavg\_transport\_m.f90: 751

# samples (I)		# samples (E)	
2.17e07	11.3%	2.17e07	11.3
2.17e07	11.3%	2.17e07	11.3
1.54e07	8.0%	1.54e07	8.0
6.32e06	3.3%	6.32e06	3.3

attributes costs to loops  
implicit with F90 vector syntax

fine grain attribution to loops  
within a loop nest

# S3D Solver for Turbulent, Reacting Flows

flat view

```
rhsf.f90
199 ! grad_Y - Species mass traction gradients may be required in transport
200 !     evaluation as well as for boundary conditions.
201 !
202 !notes by ramanan - 01/05/05
203 !The array dimensioning can be misleading
204 !For grad_u, 4th dimension is the direction and 5th dimension is the velocity component
205 !For grad_Ys, 4th dimension is the species and 5th dimension is the direction
206
207 call computeVectorGradient( u, grad_u )
208 call computeScalarGradient( temp, grad_T )
209 do n=1,n_spec
210   call computeScalarGradient( yspecies(:,:,:,n), grad_Ys(:,:,:,n) )
211 enddo
212
213 !Added by Ramanan - 01/05/05
214 !Store the boundary grad values
215 if(vary_in_x==1)then
216   if (xid==0) then
217     grad_u_x0 = grad_u(1,:,:,1,:)
218     grad_Ys_x0 = grad_Ys(1,:,:,1)
219     h_spec_x0 = h_spec(1,:,:)
220   end if
```

highlights costs for an implicit loop that copies non-contiguous 4D slice of 5D data to contiguous storage

Calling Context View Callers View Flat View

Scopes

- Experiment Aggregate Metrics
  - ~~~s3d\_f90.x:<unknown-file>~~~: 0
    - ▶ loop at mixavg\_transport\_m.f90: 735-760
      - ▼ loop at rhsf.f90: 209-210
        - ▶ loop at rhsf.f90: 210
        - ▶ loop at mixavg\_transport\_m.f90: 1004-1011

# samples (I)	# samples (E)
1.91e08 100.0	1.91e08 100.0
2.60e07 13.6%	2.60e07 13.6%
2.17e07 11.3%	2.17e07 11.3%
2.03e07 10.6%	1.91e07 9.4%
2.03e07 10.6%	1.04e07 5.4%
8.94e06 4.7%	8.94e06 4.7%

# S3D Solver for Turbulent, Reacting Flows

flat view

```

737
738     if (baro_switch) then
739         ! driving force includes gradient in mole fraction and baro-diffusion:
740         diffFlux(:,:,:,n,m) = - Ds_mixavg(:,:,:,n) * ( grad_Ys(:,:,:,n,m) &
741             + Ys(:,:,:,n) * ( grad_mixMW(:,:,:,m) &
742             + (1 - molwt(n)*avmolwt) * grad_P(:,:,:,m)/Press))
743     else
744         ! driving force is just the gradient in mole fraction:
745         diffFlux(:,:,:,n,m) = - Ds_mixavg(:,:,:,n) * ( grad_Ys(:,:,:,n,m) &
746             + Ys(:,:,:,n) * grad_mixMW(:,:,:,m) )
747     endif
748
749     ! Add thermal diffusion:
750     if (thermDiff_switch) then
751         diffFlux(:,:,:,n,m) = diffFlux(:,:,:,n,m) &
752         - Ds_mixavg(:,:,:,n) * Rs_therm_diff(:,:,:,n) * molwt(n) &

```

waste metric  
peak FLOPs -  
actual FLOPs

highlights memory  
hierarchy problems here

Flat View

Scopes	PAPI_TOT_CYC	WASTE	PAPI_FP_INS	PAPI_TOT_INS	PAPI_STL_ICY
Experiment Aggregate Metrics	6.73e11	100.0	2.05e11	4.56e11	1.59e10
loop at mixavg_transport_m.f90: 735-760	6.96e10	10.3%	9.00e09	4.06e10	1.32e09
loop at mixavg_transport_m.f90: 736-758	6.96e10	10.3%	9.00e09	4.06e10	1.32e09
loop at mixavg_transport_m.f90: 745	4.85e10	7.2%	6.27e09	2.40e10	1.14e09
loop at mixavg_transport_m.f90: 758	2.11e10	3.1%	2.73e09	1.66e10	1.72e08
mixavg_transport_m.f90: 736	4.00e06	0.0%			
mixavg_transport_m.f90: 743	3.00e06	0.0%			2.00e06
mixavg_transport_m.f90: 747	3.00e06	0.0%		1.00e06	
mixavg_transport_m.f90: 754	2.00e06	0.0%			

# Status

---

- **Research prototype available only on Cray XD1**
  - being refined for broader use
- **Porting to Catamount and CNL for Cray XT3 & XT4**
  - support for statically-linked binaries
- **Adding support for HW counter call path profiling**
- **Adding support for comparative analysis**
  - viewer currently analyzes node programs
  - enhance to analyze processes
    - within executions
    - across executions



# Acknowledgments

---

- **HPCToolkit development is currently being supported by the Department of Energy's Office of Science as part of the SciDAC**
  - Performance Engineering Research Institute (PERI) under Cooperative Agreement No. DE-FC02-06ER25762**
  - Center for Scalable Application Development Software (CScADS) under Cooperative Agreement No. DE-FC02-07ER25800**
  - Experiments performed on Cray XD1 funded by the National Science Foundation under Grant CNS-0421109, and a partnership between Rice University, Advanced Micro Devices, and Cray**