6th Parallel Data Storage Workshop, held in conjunciton with SC 11

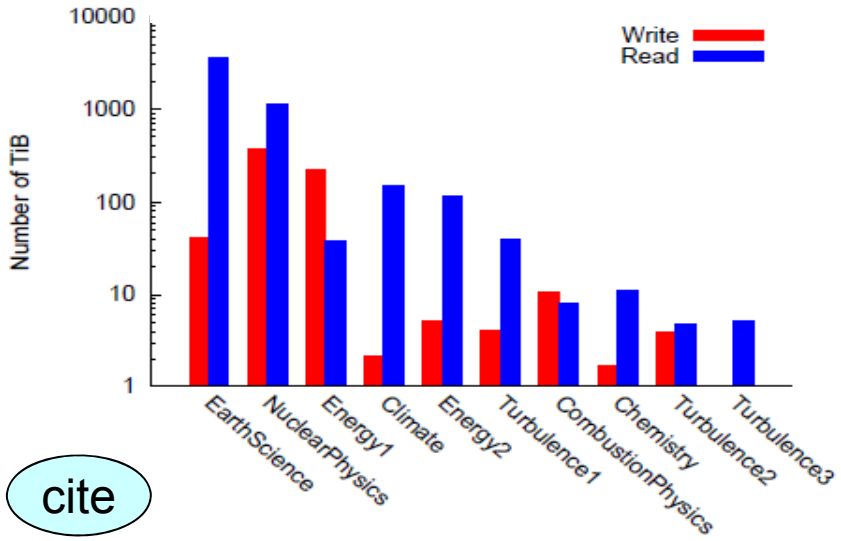
In-Situ I/O Processing: A Case for Location Flexibility

Fang Zheng, Hasan Abbasi, Jianting Cao,
Jai Dayal, Karsten Schwan, Matthew Wolf
College of Computing, Georgia Tech
Scott Klasky, Norbert Podhorszki
Oak Ridge National Laboratory

I/O Bottleneck on High-End Machines

- Scientific simulation and analysis are data-intensive

Appl.	Data size (MB/Core)	Data size (MB/Node)
GTC	180	2880
XGC1	120	920
GTS	220	3520
Chimera	10	160
S3D	14	224
GEM	20	320
M3D-k	14	224



cite

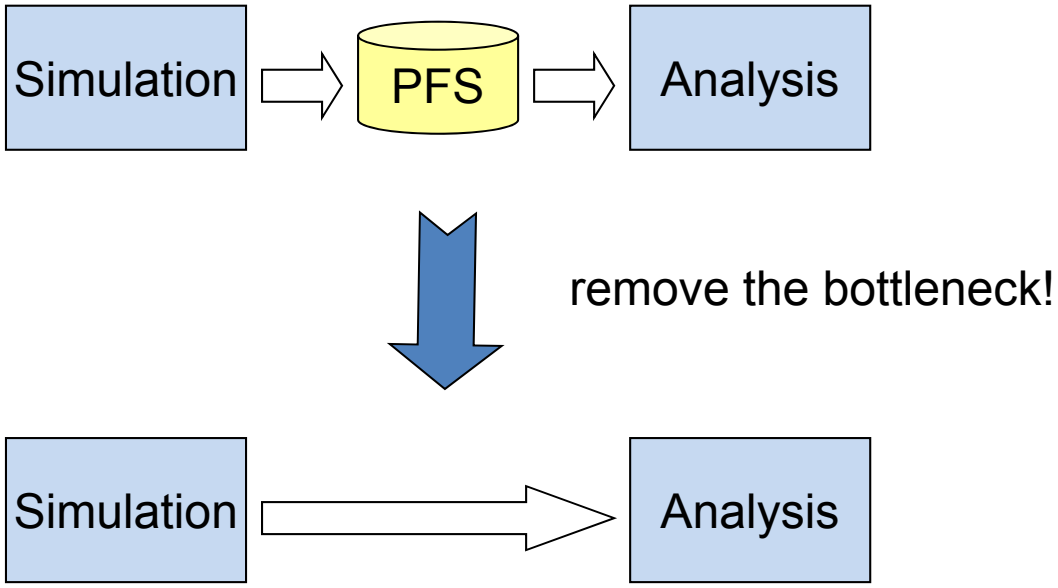
- I/O subsystem is not catching up
  - capacity mismatching between computation vs. I/O
  - complicated I/O pattern
  - shared resource contention

Machine	Peak Flops	Peak I/O bandwidth	Flop/byte
Jaguar Cray XT5	**2.3 Petaflops**	**120GB/sec**	**191666**
Franklin Cray XT4	**352 Teraflops**	**17GB/sec**	**20705**
Hopper Cray XE6	**1.28 Petaflops**	**35GB/sec**	**36571**
Intrepid BG/P	**557 Teraflops**	**78GB/sec**	**7141**

Simulation and analysis spends significant portion of runtime waiting for I/O to finish!

What is In-Situ I/O Processing?

- Process/analyze simulation output data before data hits disks, during simulation time

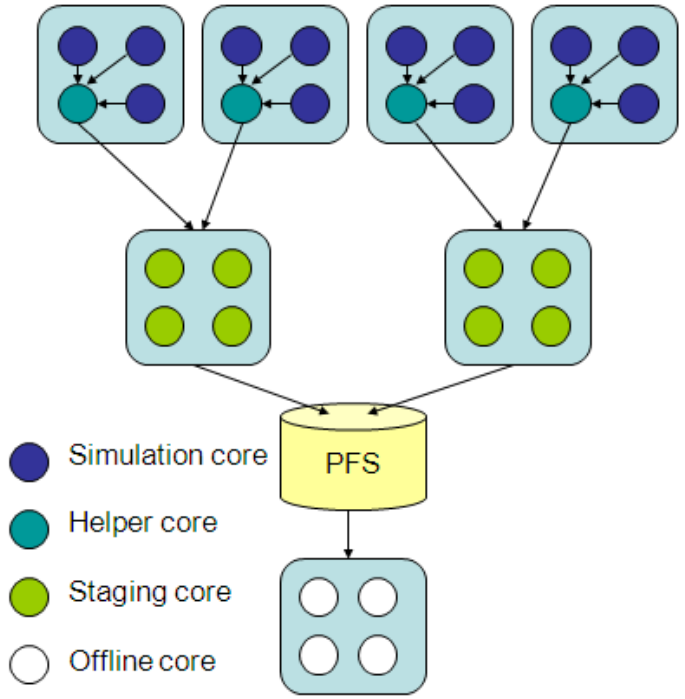


Why In-Situ I/O Processing?

- Get around I/O bottleneck by reducing file I/O
  - Reduce data movement along I/O hierarchy
  - Extract insights from data in a timely manner
  - Prepapre data better for later analysis
  - Better end-to-end performance and cost

Placement of In-Situ Analytics

- Active R&D efforts
  - Active Storage (recently ANL and PNNL)
  - Hercules/Quakeshow (CMU&UCDavis&UTAustin&PSC)
  - ADIOS/DataStager/PreDatA (GT&ORNL)
  - DataSpaces (Rutgers&ORNL)
  - Nessie (Sandia)
  - GLEAN (ANL)
  - Functional partitioning (ORNL&VT&NCSU)
  - HDF5/DSM (ETH&CSCS)
  - ParaView co-processing library (ParaView)
  - VisIt remote visualization (VisIt)
  - In-situ indexing (LBL), compression (NCSU), etc.
- Question: Where should I run In-situ analysis?
  - Inline with simulation?
  - Seperate core?
  - Seprate staging nodes?
  - I/O servers?
  - Offline?



Placement Matters!

- Placement of In-situ I/O processing have significant impact on performance and cost
  - How resource is allocated between simulation and analysis
  - How data is moved between simulation and analysis (interconnect, shared memory, etc.)
  - Resource contention effect

Flexible Placement is Important

- No one place fits everything
  - Diverse characteristics of simulaiton and analytics
  - Machine parameters
  - Resource availability
- Understanding how placement decision affects performance and cost is valuable for end-users

Contributions of This Paper

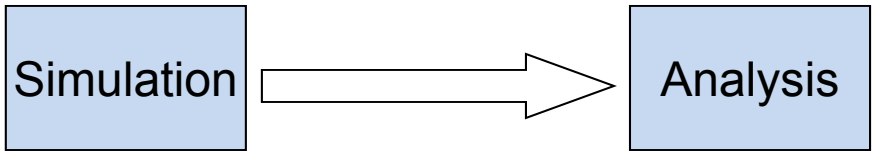
- A (Simple) performance model to reason about placement
  - Capable of comparing performance and cost of different placements
- Application case study-- Pixie3D I/O Pipeline
  - Placement makes huge difference in performance and cost
  - Empirically validate the model

Performance and Cost Metrics

- Performance Metric
  - **Total Execution Time** of both simulation and analysis
- Cost Metric
  - **CPU hours** charged for simulation and analysis

Performance Modeling

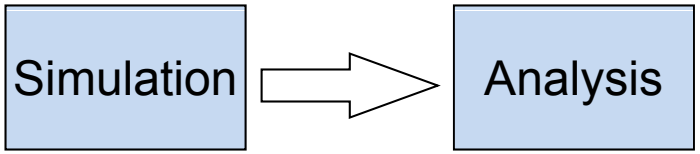
- Scenario:
  - Simulation periodically generate output data and pass to analyis component
  - Analysis process the simulation output data on a per-timestep basis



Performance Modeling

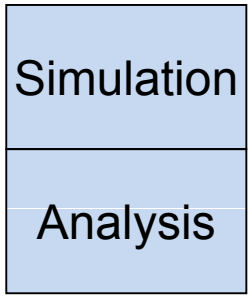
- Place analysis in a staging area vs. inline with simulation?

In Staging Area:
- Simulation runs on *Psim* nodes
- Analysis runs on another *Pa* nodes
- Space partition (*Psim*+*Pa*) nodes between simulation and analysis
- Pass data through interconnect



Psim nodes *Pa* nodes

Inline with simulation:
- Both simulation and analysis run on the same *Psim* nodes
- Simulation nodes perform analysis inline synchronously on *Psim* nodes
- Simulation and analysis share *Psim* nodes in time



Psim nodes

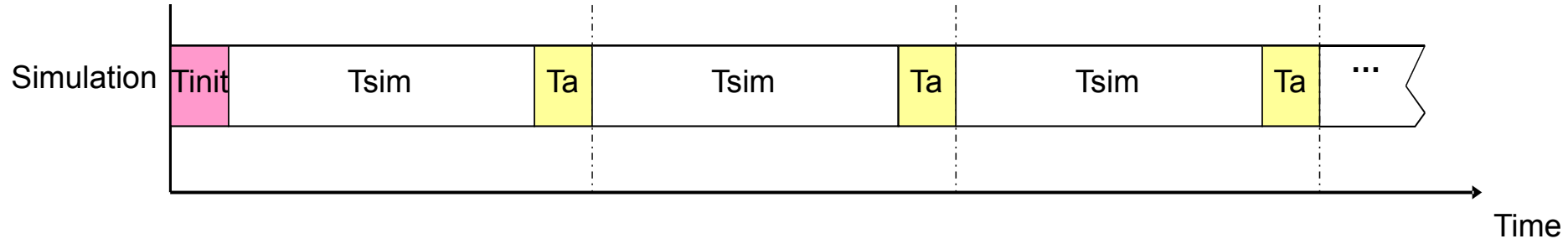
Performance Modeling

- Key parameters

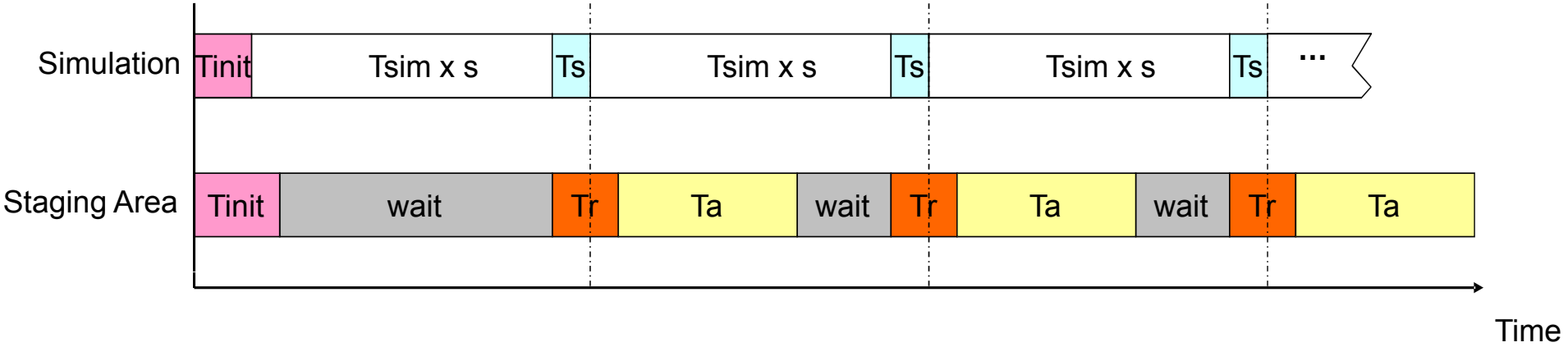
Psim	**Total number of nodes on which simulation is run**
Pa	**Total number of nodes in staging area (if present)**
***Tsim*(*P*)**	**Simulation's wall-clock time between two consecutive I/O actions when running on *P* nodes**
***Ta*(*P*)**	**Analysis' wall-clock time for processing one simulation output step when running on *P* nodes**
K	**Total number of I/O dumps**
Tsend	**Simulation-side visible data movement time**
Trecv	**Staging node-side visible data movement time**
s	**Slowdown factor of simulation**

Performance Modeling

- Total execution time



$$Tinline = K \times [Tsim(Psim) + Ta(Psim)]$$



$$Tstaging = K \times \max\{Tsim(Psim) \times s + Tsend, Trecv + Ta(Pa)\}$$

Pipeline effect of simulation and analysis

Slowdown factor of simulation (s>=1)

Performance Modeling

- Performance comparison of inline vs. staging

$$Speedup = \frac{Tinline}{Tstaging}$$

Let ***α=Pa/Psim***

(size of staging area as percentage of total simulation nodes)

β=Ta(Psim)/ Tsim(Psim)

(analysis time as percentage of simulation time on *Psim* nodes)

$$Speedup = \frac{Tsim(Psim)(1+\beta)}{\max\{Tsim(Psim)\times s + Tsend, Trecv + Ta(Psim\times\alpha)\}}$$

since

$$\max\{Tsim(Psim)\times s + Tsend, Trecv + Ta(Psim\times\alpha)\} > Tsim(Psim)\times s$$

There is a upper bound: $Speedup < (1+\beta)/s$

Performance Modeling

- What does the model say?
  - Total execution time is (1+β) if running analysis inlne with simulation on *Psim* nodes
  - If we can use $\alpha\%$ additional nodes as staging area to offload the analysis to staging area
  - If co-running staging area slows down simulation by a factor of s
  - Then the speedup of such offloading is bounded by

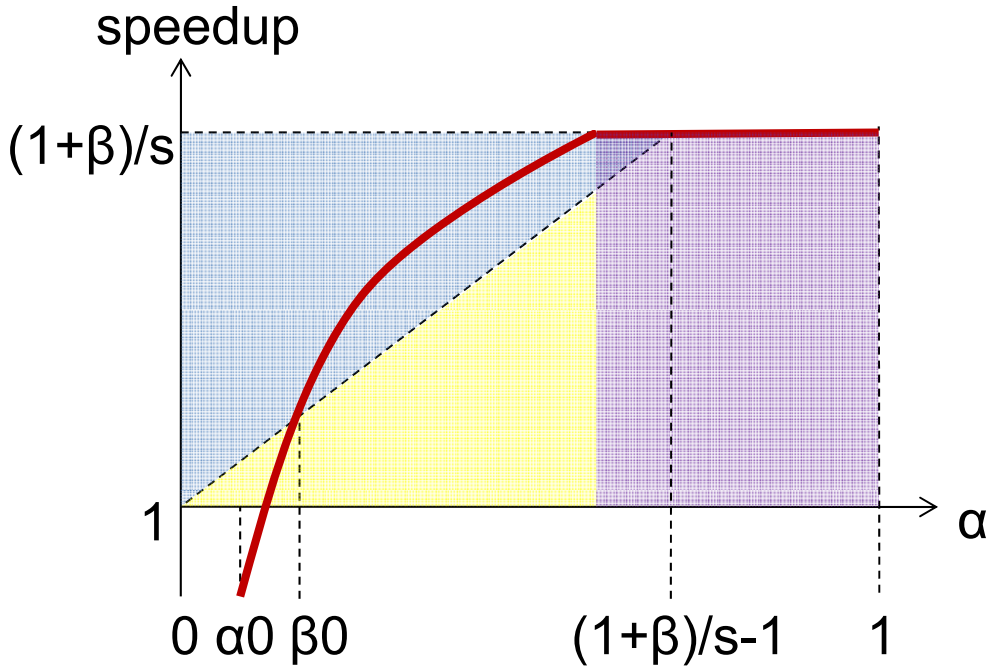
$$Speedup < (1+\beta)/s$$

Performance Modeling

- Comparing Cost of Staging vs. Inline
  - Cost (inline)=Tinline x Psim
  - Cost (staging) = Tstaging x (Psim+Pa)
- We want to know the cost efficiency of using additional staging area to offload analysis
  - Does ***α%*** of additional nodes leads to ***α%*** improvement in Speedup?

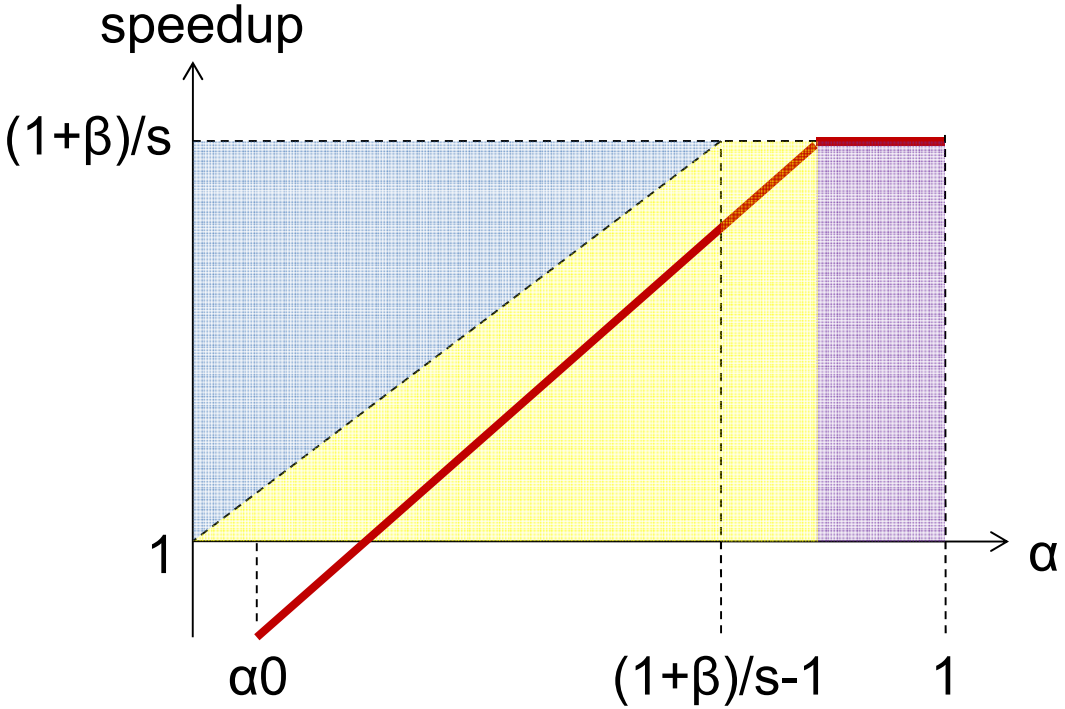
Performance Model

- Key to achieve good speedup and efficiency
  - No slowdown: *s*=1
  - *Tsend*=0
  - *Tsim*(*Psim*)>*Trecv*+*Ta*(*Pa*)
  - *Ta*(*P*) scales sub-linearly with *P* (*Ta*(*P*)x*P* decrease with *P*)



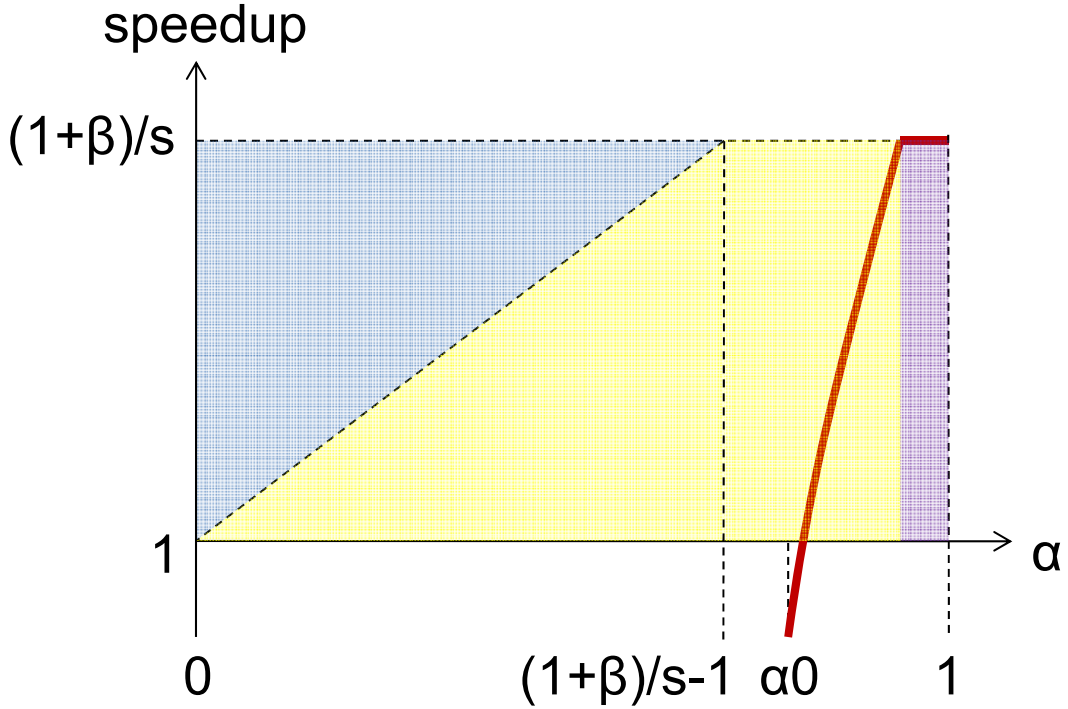
Performance Model

- Not cost-efficient to offload linear-scalable analysis:
  - *Ta*(*P*) x *P* doesn't change
  - Offloading only increase data movement cost



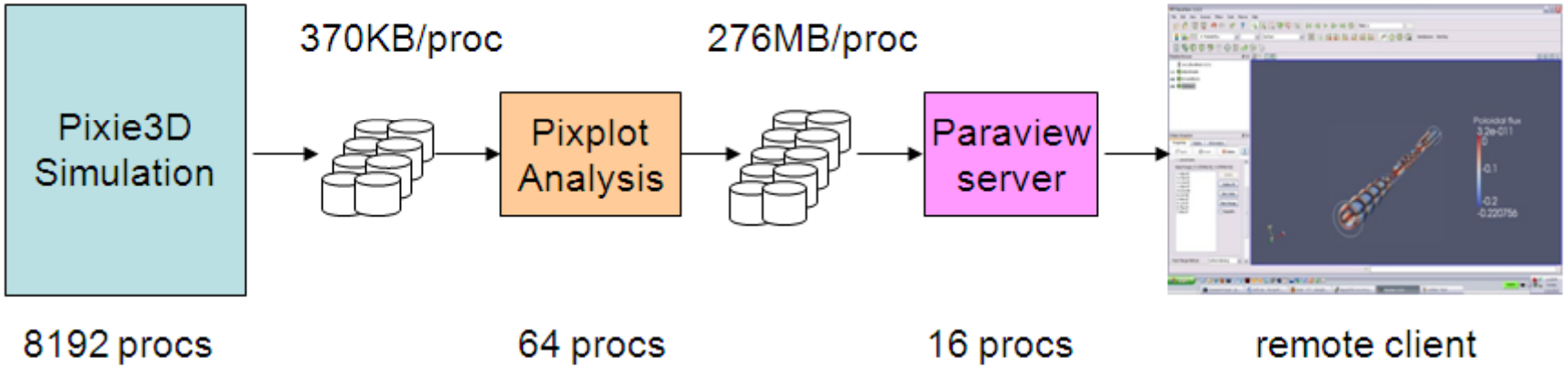
Performance Model

- When the minimum size of the staging area ($\alpha 0$), is larger than $(1+\beta)/s-1$, then offloading is always in-efficient



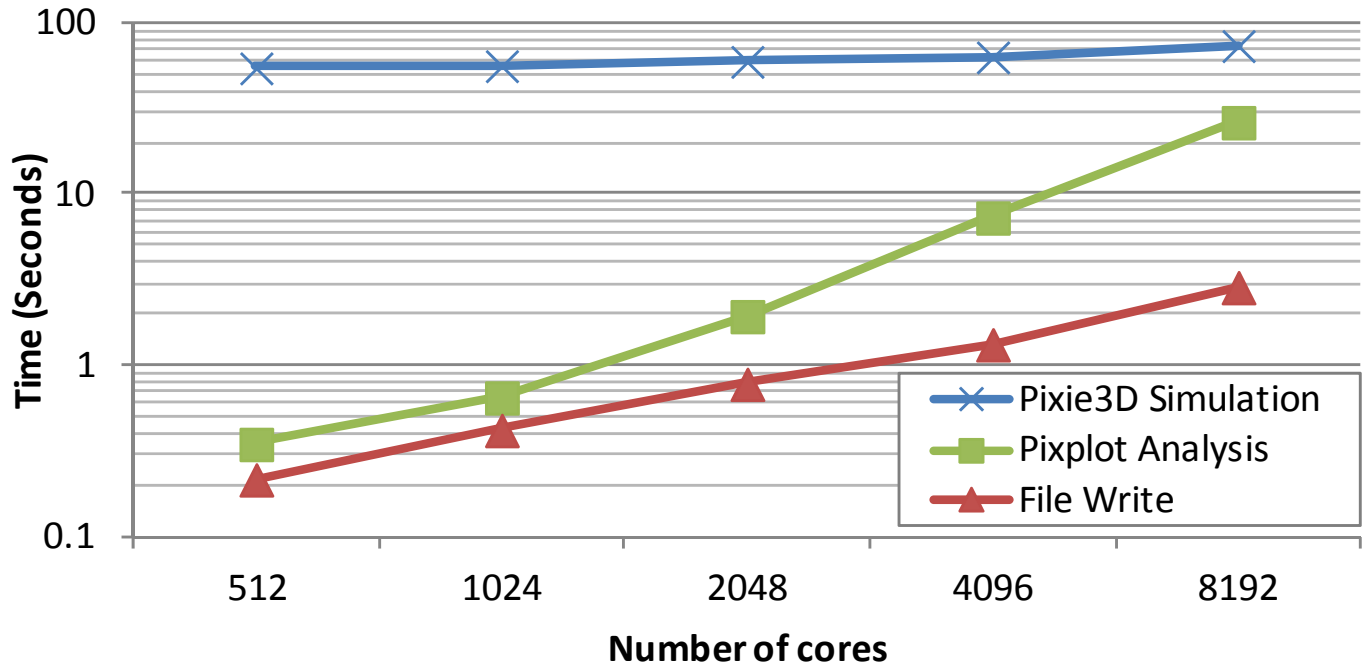
Application Case Study

- Pixie3D In-Situ I/O Pipeline
  - Pixie3D MHD simulation
  - Pixplot: diagnostic analysis
  - Paraview server: contour plotting
- Implement with ADIOS/PreDatA middleware



Pixie3D Performance

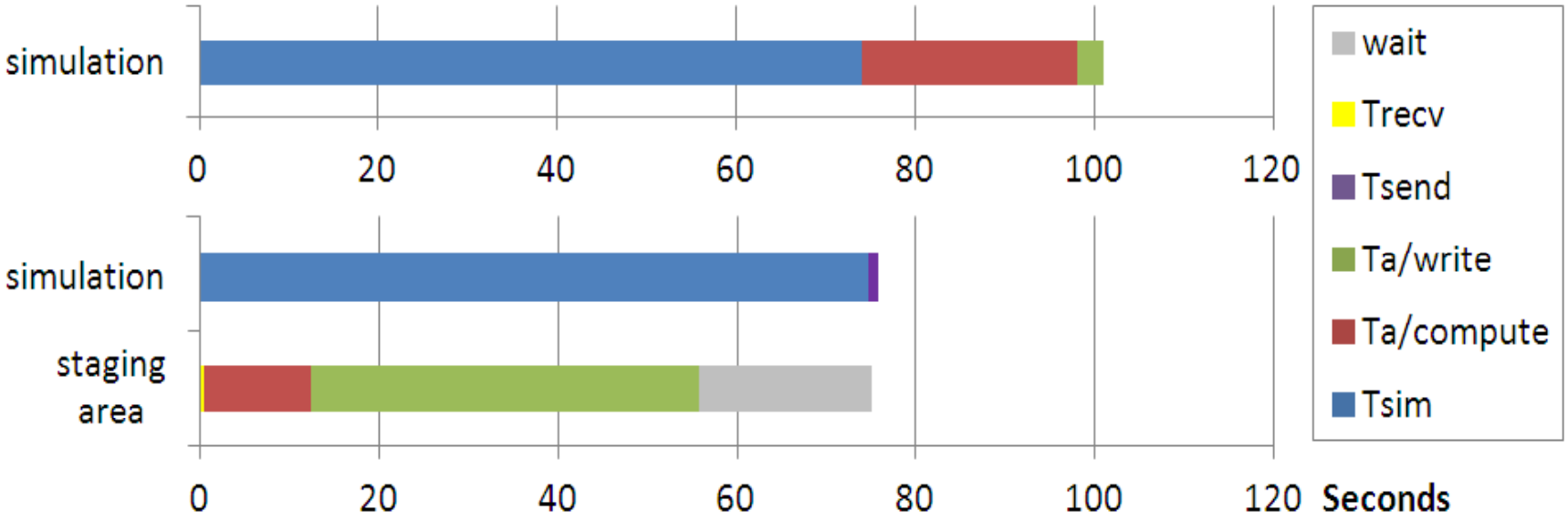
- Scalability



- Pixplot analysis and I/.O scales worse than Pixie3D simulation, so placing inline Would hurt scalability.
- Offloading to a staging area may get good speedup and efficiency

Pixie3D Performance

- Time Breakdown
  - Run Pixie3D on 8192 cores, Pixplot on 64 cores

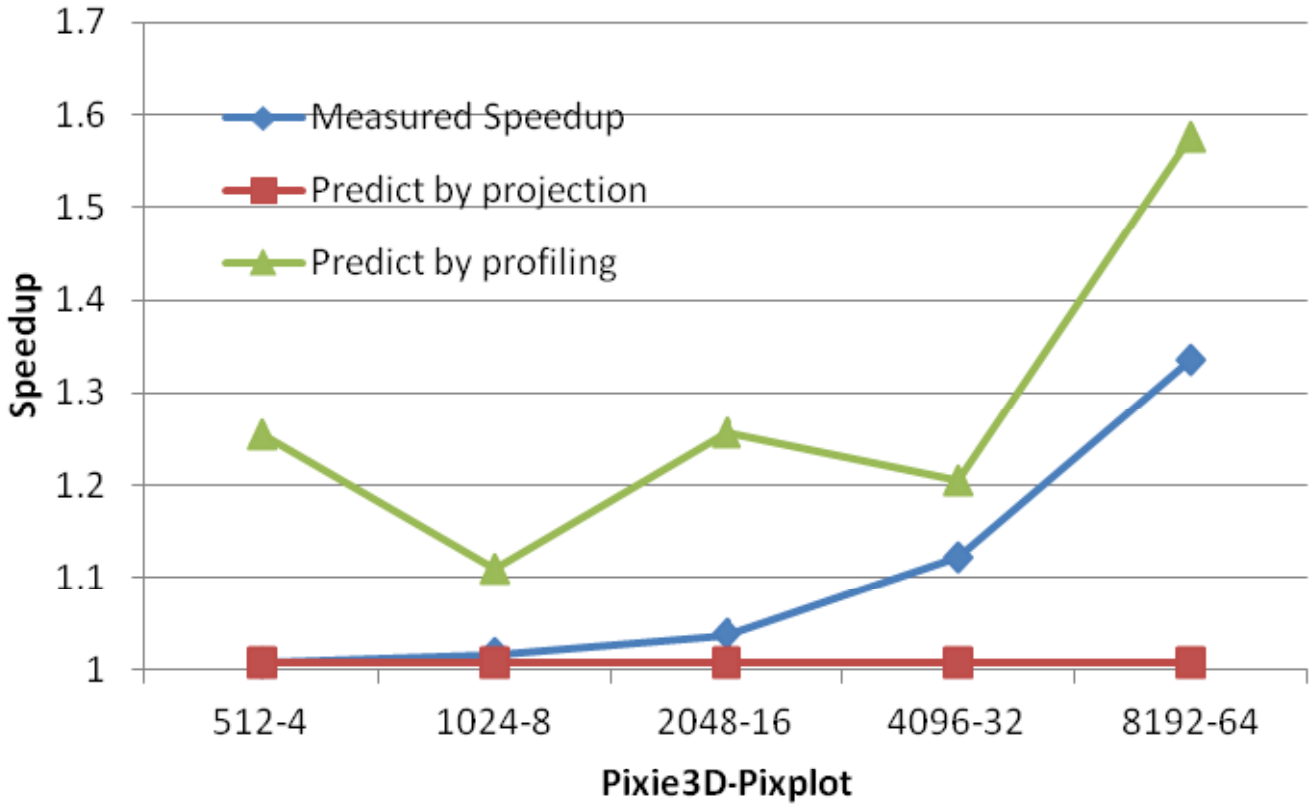


- Using 0.78% additional nodes as staging area, offloading Pixplot and I/O to staging area increases performance by 33%
- The speedup is within 96% of upper bound

Pixie3D Performance

- Predict the speedup using the model
  - Predict by projection: measure actual performance at a small scale and project ot target scale
  - Prediction by profiling: run simulation and analysis inline at Psim nodes, and predict speedup by (1+β)

Pixie3D Performance



- Projection-based approach is too conservative because it doesn't consider analysis' scalability

-Profiling-based approach is too optimistic because it omit slowdown and data copy cost

Summary of Performance Model

- Assume per-timestep, simulation-driven case
- Can be used to compare inline vs. staging
- Can be extended to offline
  - Tsend and Trecv is file read/write time
  - slowdown factor: interconnect, storage server side
- Can also be extended to dedicated core
  - Trecv is shared memory copy
  - slowdown factor: contention on shared cache/memory bandwidth within compute node

Conclusions

- Placement makes measurable difference in performance and cost
- Flexible placement is needed for diverse workloads
  - This paper focus on scalability feature of analysis
- Future work:
  - Make model more predictive
  - Automatic placement

Acknowledgements

- The authors thank Berk Geveci, Sebastien Jourdain, and Pat Marion from Kitware Inc. and Kenneth Moreland from Sandia National Laboratory for integrating ADIOS with ParaView and aid in implementing Pixie3D I/O processing pipeline.
- This work was funded in part by Sandia National Laboratories under contract DE-AC04-94AL85000, by the DOE Office of Science, Advanced Scientific Computing Research, under award number DE-SC0005505, program manager Lucy Nowell, and by the Department of Energy under Contract No. DEAC05- 00OR22725 at Oak Ridge National Laboratory. Additional support came from the resources of the National Center for Computational Sciences at Oak Ridge National Laboratory, a grant from NSF as part of the HECURA program, a grant from the Department of Defense, a grant from the Office of Science through the SciDAC program, and the SDM center in the ASCR office.

Thank you very much!