

Data Throttling for Data-Intensive Workflows

Sang-Min Park and Marty Humphrey

Dept. of Computer Science,
University of Virginia

This work was supported by the National Science Foundation under Grant No. SCI-0426972 and Grant No. SCI-0438263, the DOE Early Career program, and the Microsoft Research eScience Program.



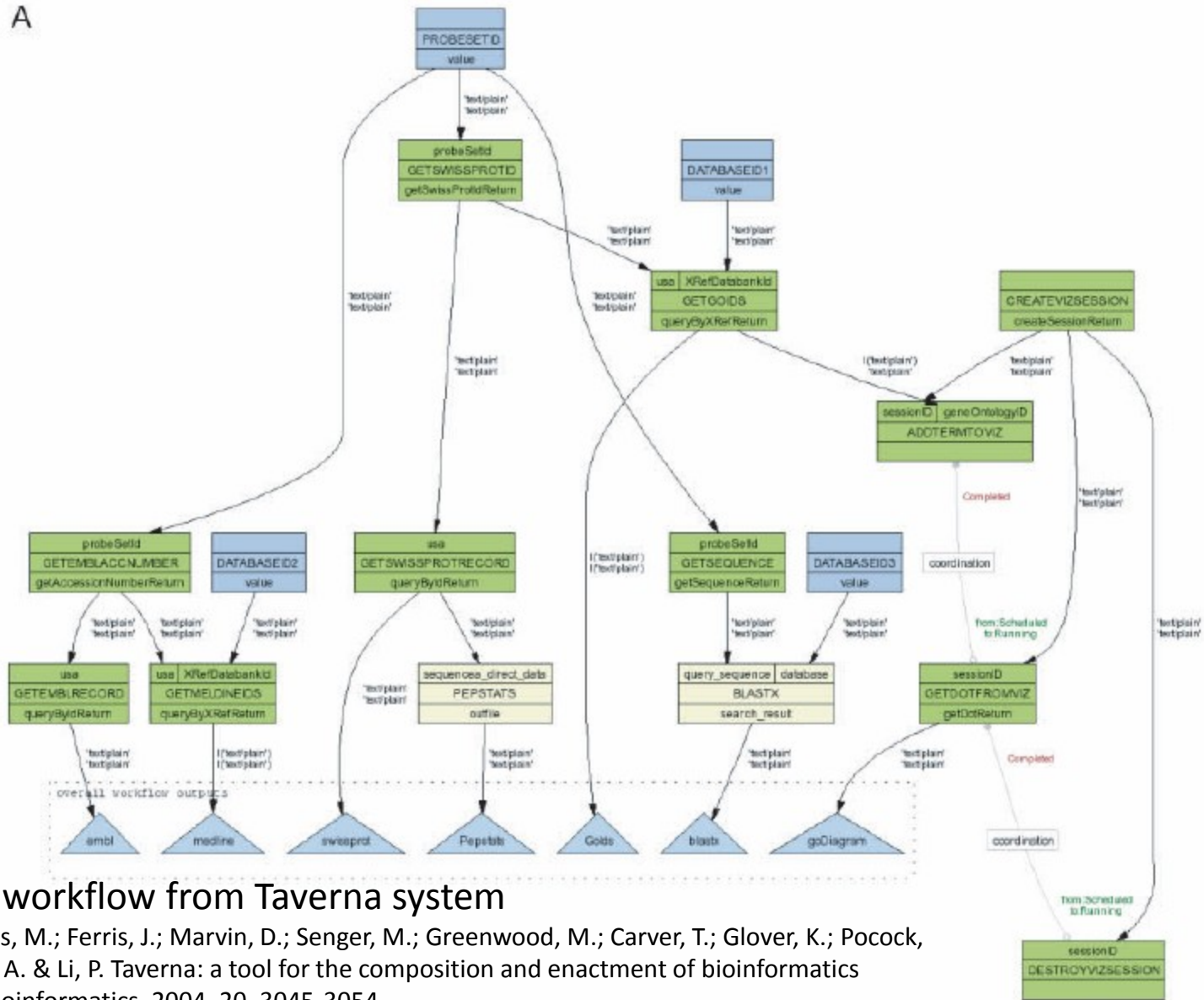
Computer Science
at the UNIVERSITY of VIRGINIA

Overview

- E-Science Workflows – Good and Bad
- Data Throttling – Motivation
- Data Throttling – Design and Implementation
- Evaluations
 - Simulation
 - Experiments with Montage workflow
- Conclusions



A



Example workflow from Taverna system

Oinn, T.; Addis, M.; Ferris, J.; Marvin, D.; Senger, M.; Greenwood, M.; Carver, T.; Glover, K.; Pocock, M. R.; Wipat, A. & Li, P. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 2004, 20, 3045-3054



E-Science Workflows - Good

- Natural way to describe scientific logic
 - In terms of computation and data dependency
 - GUI support
 - Easy to map on GRID
- Different aspects of eScience workflows
 - For domain scientist:
 - Problem solving environment
 - Data provenance for reproducible science
 - For computer scientist
 - Separation of concern: single abstraction and multiple implementation
 - Unified application model to apply research ideas



E-Science Workflows - Bad

- Easy to domain scientist = Hard to computer scientist
- Hard problems (research issues)
 - *Reliability*: How to run tens of thousand tasks without failure?
 - *Abstraction*: how to describe scientific logic? Is there a standard, unified way to do it?
 - ***Performance***: How to run fast or how to run it efficiently?



E-Science Workflows – Performance

- Big, complex application
 - Tens of thousands task (e.g., LIGO, Montage, etc)
 - Complex dependencies between tasks
- Dynamic GRID resources
 - Availability of computing nodes change
 - Data bandwidths fluctuate
- Large infrastructure does not guarantee high performance
 - Unbalanced computation
 - Slow communication
 - Inefficient workflow engine and middleware



E-Science Workflows – Performance

- Task scheduling – existing solutions
 - Map tasks (or data) to nodes in a way to minimize workflow's execution time (old and hard problem)
- Scheduling is not sufficient
 - Resource's state changes rapidly
 - Data increasingly dominates computation
 - Often data movement delay is not considered
 - Simplistic approach to predict bandwidth does not solve the problem
 - Prediction is often wrong and too intrusive to be accurate
 - Workflow generates huge amount data flow, breaking the prediction by itself



Problem in data movement

- Conventional practice of data movement
 - Use high performance tool (e.g., GridFTP)
 - Send (or receive) immediately, as fast as possible
- This is often *unnecessary*
 - Data is not consumed by tasks until all predecessor's data is received
- This is *harmful!*
 - Link's capacity is limited
 - Some unnecessarily fast transfer slow down other (more urgent) transfers



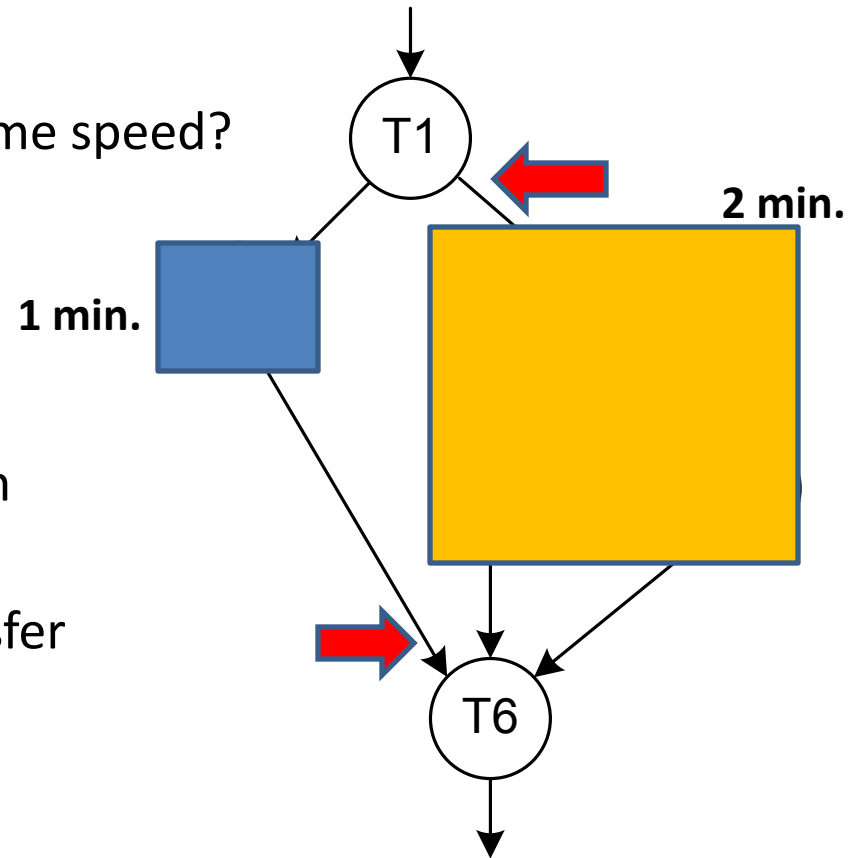
Problem in data movement

- **Outgoing edge**

- Why send data to T2, T3 at the same speed?
- Then, can we send it to T3 faster than T2? (No, currently)

- **Incoming edge**

- Does T6 need to receive data from T2 earlier than T4, T5?
- Then, can we slow down the transfer from T2? (No, currently)



Data Throttling

- Control workflow edge's data movement delay
- Improving performance
 - Throttle up to more urgent sections of workflow
 - Throttle down to less urgent sections
 - Compensate imbalance of workflow execution
- Efficient bandwidth usage
 - Throttle down if early arrival is not necessary
 - Use limited bandwidth with other GRID activities in harmonious way



Data Throttling

- Programmable interface – annotation on workflow graph
- Enforcement mechanism – QoS-enabled GridFTP



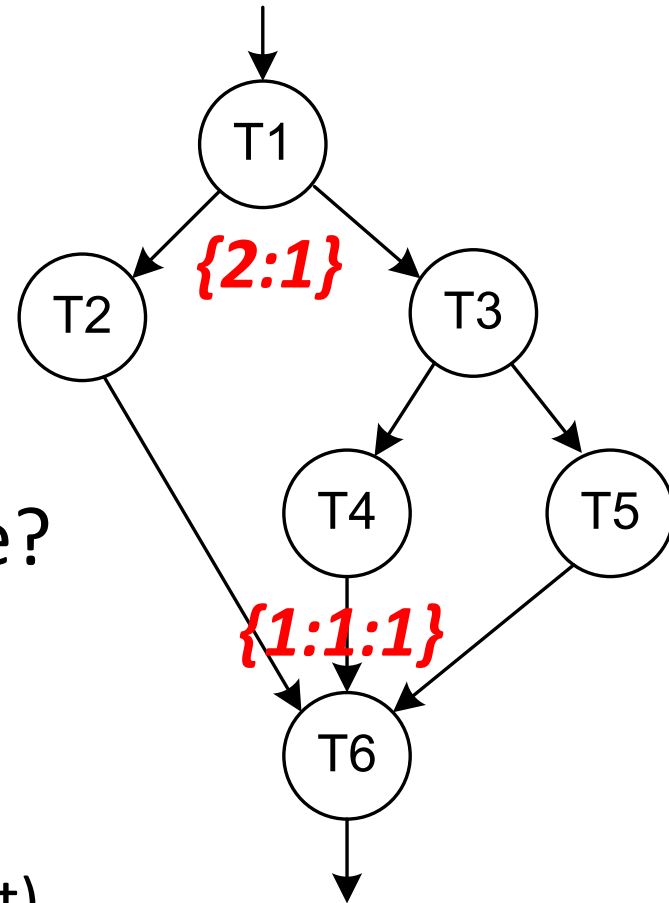
Data Throttling – Interface

- Allow users to control data transfer delay
- Define new workflow programming attribute:
Relative Transfer Time Finished (RTTF)
- RTTF defines relative file transfer time within a group of incoming or outgoing edges
- User or workflow engine annotate workflow graph with RTTF values



Data Throttling – Interface

- RTTF attribute
 - Enforce priority of file transfer delay among edges
 - Throttle down bandwidth if fast movement is not necessary
- Why relative delay as interface?
 - Sufficient to express priority relationship between edges
 - Simpler implementation (w/o end-to-end router QoS support)



Data Throttling – Implementation

- Application-level QoS via rate limiting on GridFTP
 - Congestion in today's network often occurs at end hosts, not broad backbone (e.g., TeraGrid 40Gb/s)
 - So it is often enough to control bandwidth consumption by popular services at end-hosts
 - **Simple** implementation without special hardware support
 - **Sufficient** to enforce relative bandwidth guarantee

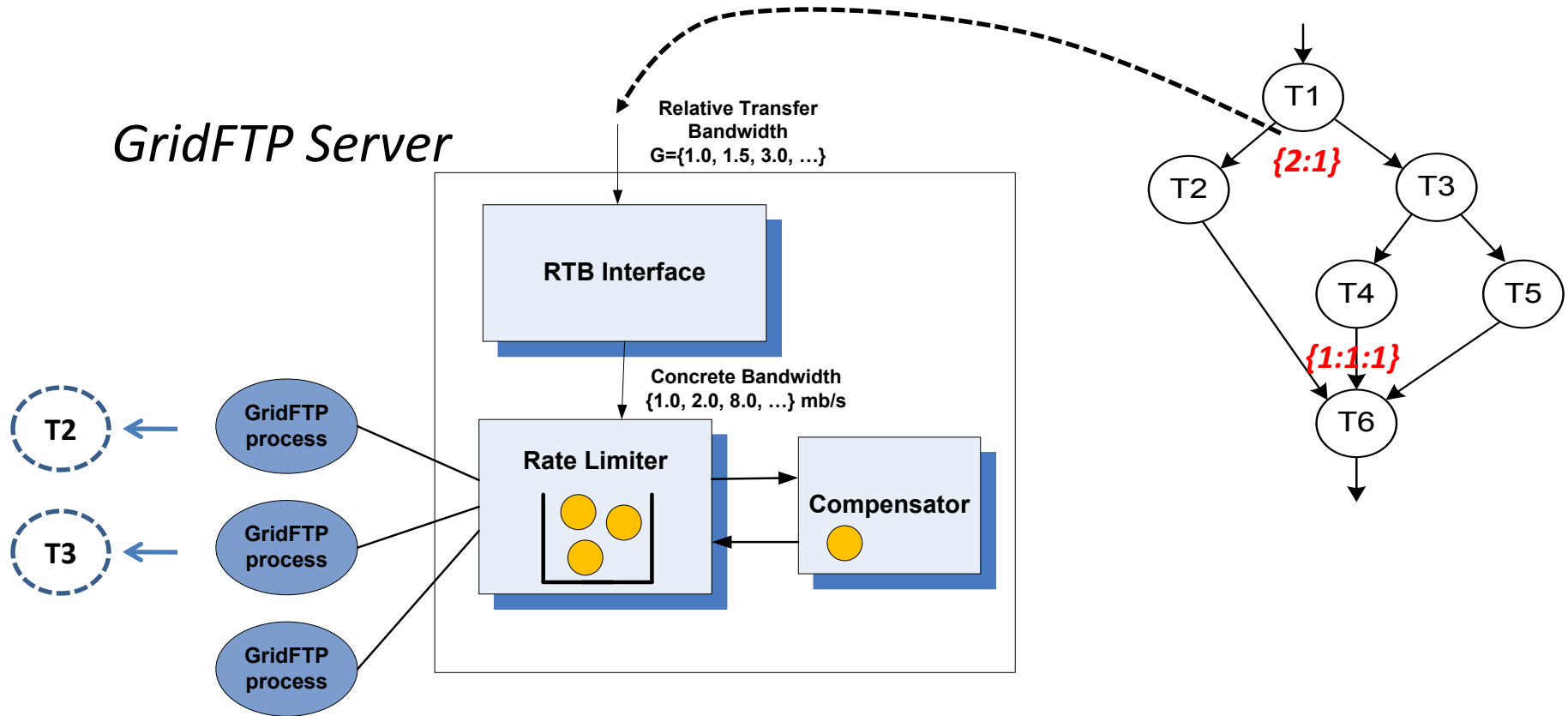


Data Throttling – Implementation

- Token bucket rate limiting
 - Algorithm used in routers
 - GridFTP transfer n bytes only if there are more than n tokens in a bucket
 - Token is filled into bucket at rate R
 - Implemented as XIO drivers in Grid server (no modification to GridFTP client)



QoS-enabled GridFTP



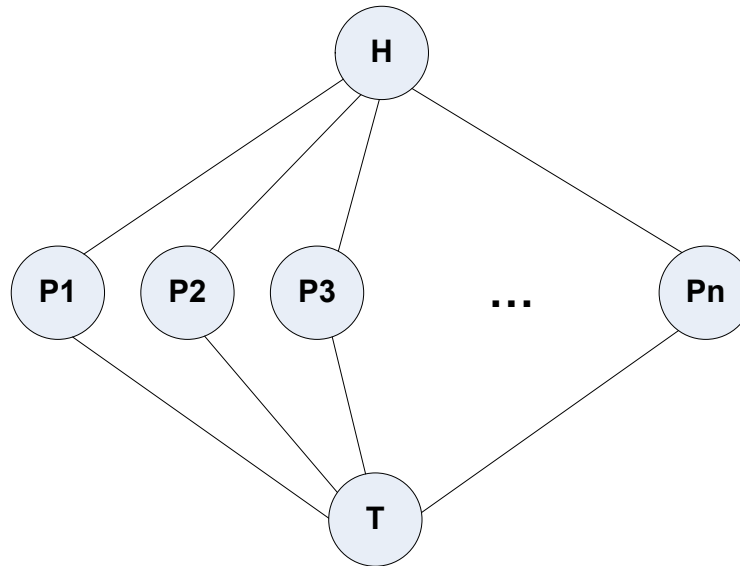
Evaluation – target applications

- Throttling improves performance as it complements task execution imbalance
- But not all applications are applicable
 - Computation \gg Communication: little room to improve
 - Computation \ll Communication: computation's imbalance is not significant
- Find range of computation-communication ratio (CCR) that throttling is worth to apply
- Simulation using GridSim package that simulates QoS network

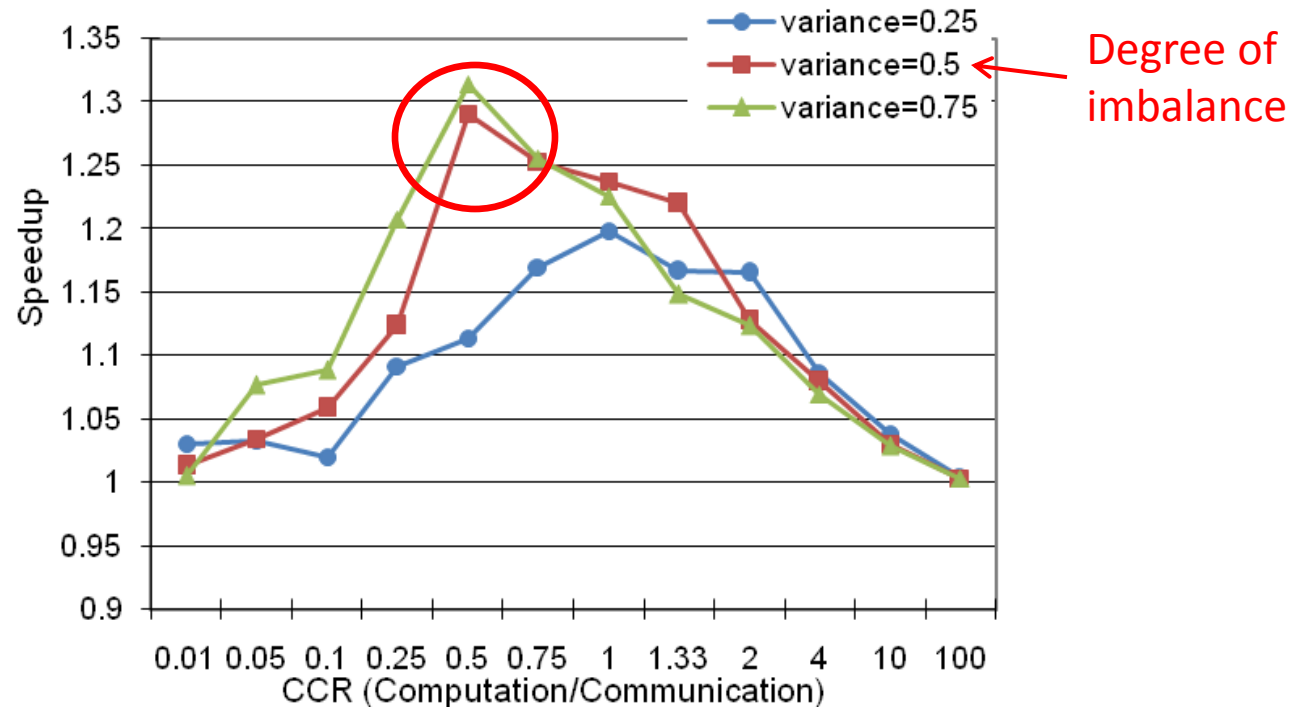


Evaluation – target applications

- Run simple, but frequent workflow pattern with varying CCR
- Users can find at which sections of large workflow throttling is worth



Evaluation – target applications



- Throttling is most effective if $\{0.5 \leq \text{CCR} \leq 2\}$
- More improvement as task imbalance is large (especially when $\text{CCR} < 1$)

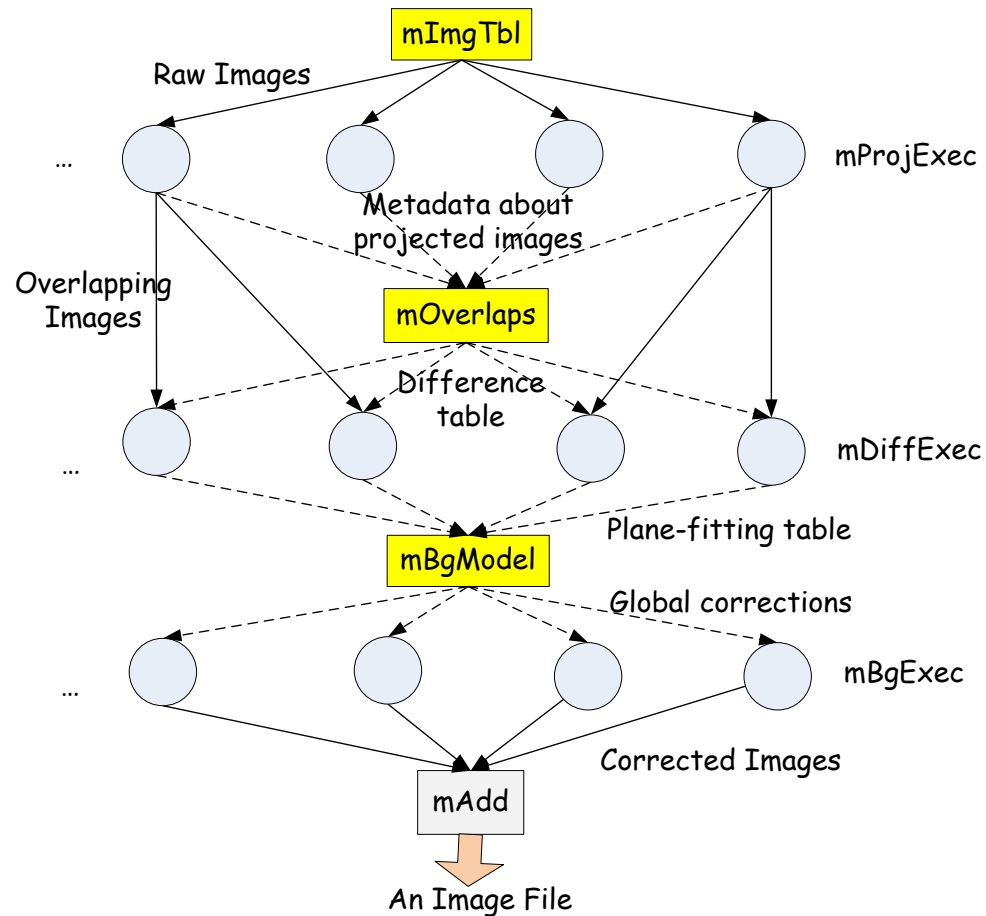


Evaluation – Montage workflow

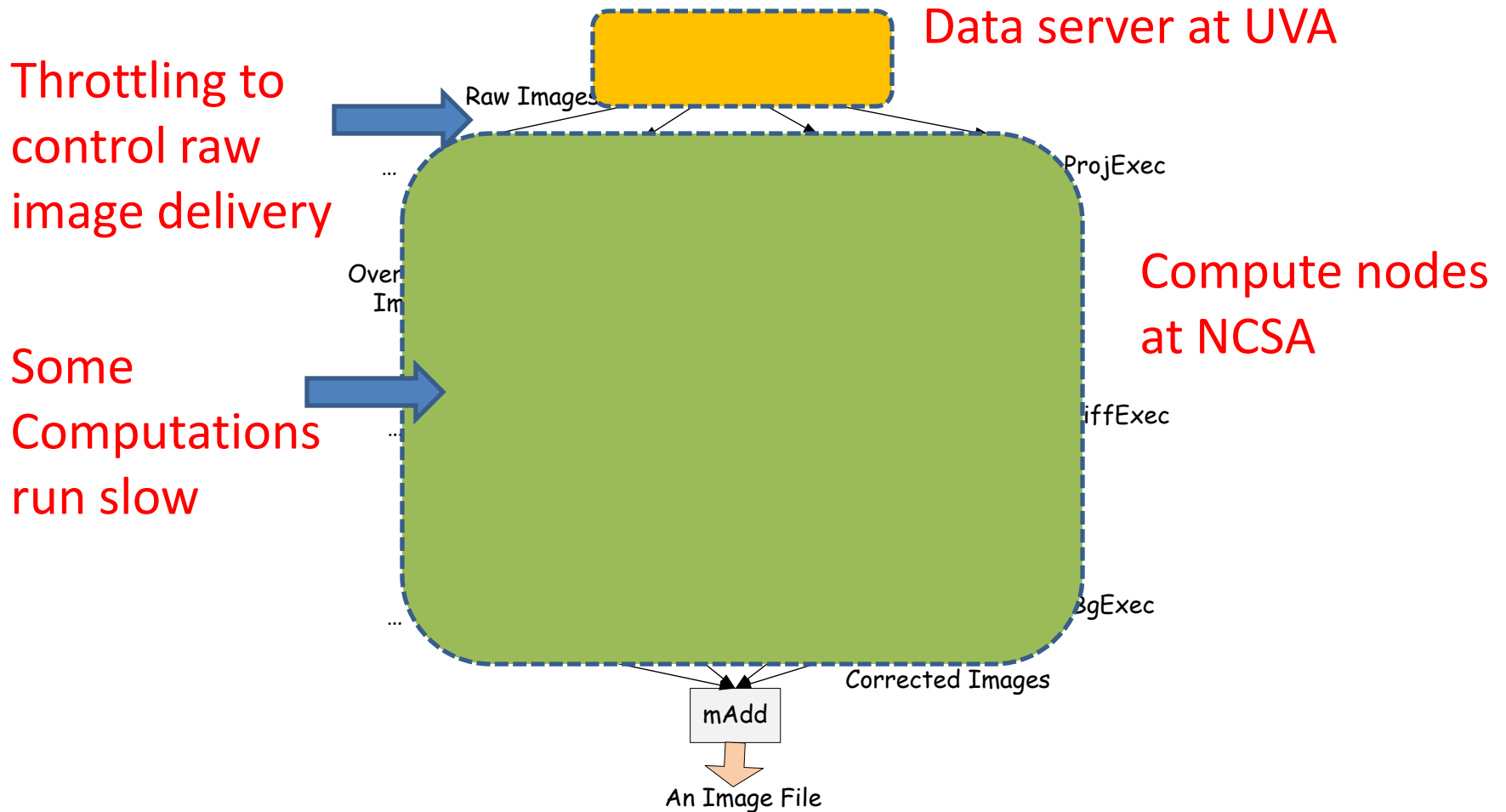
- Montage
 - Image mosaic engine for astronomy research
 - Compute and data intensive application with inherently large parallelism
 - Integrated with Grid workflow systems (e.g., Pegasus)
 - Has range of CCR worth to apply throttling
- Resources
 - UVA runs data server
 - NCSA has compute nodes
 - Internet 2 links between UVA and NCSA



Evaluation – Montage workflow



Evaluation – Montage workflow



Results – Montage workflow

2Mass object	Baseline (sec.)	Rate Throttled (sec.)	Speedup (Baseline/Throttle)
M70	253.3	219.5	1.15
M71	274.3	226.5	1.21
M72	349.5	266.8	1.31
M73	307.5	277.5	1.11
M74	267.6	234.3	1.14
M75	294.0	256.3	1.15
M76	299.5	268.6	1.12
M77	319.7	282.8	1.13
M78	276.4	242.5	1.14
M79	296.1	263.6	1.12
Average	293.8	253.8	1.16



Conclusions

- Current workflow system lacks control over data movement delay
- Relative bandwidth guarantee via QoS-enabled GridFTP can improve workflow performance as well as enhance efficiency in bandwidth consumption



Questions?
(sp2kn@virginia.edu)



Computer Science
at the UNIVERSITY of VIRGINIA