Data Scheduling with Stork in Distributed Repositories

Mehmet Balman, Ismail Akturk, Tevfik Kosar

Center for Computation and Technology, Louisiana State University

www.storkproject.org

STORK: A Scheduler for Data Placement Activities in Distributed System for Large-Scale Applications

Presented in OSG All Hands Meeting
March 2-5, 2009

Stork: Data Placement Scheduler

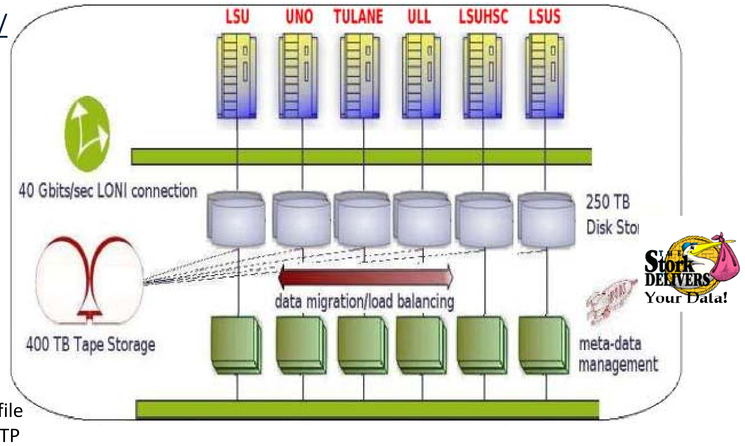
Ex: **submit file**

```
[    dest_url = "gsiftp://eric1.loni.org/scratch/user/";
     arguments = "-p 4 –dbg -vb";
     src_url = "file:///home/user/test/
     dap_type = "transfer";
     verify_checksum = true;
     verify_filesize = true;
     set_permission = "755" ;
     recursive_copy = true;
     network_check = true;
     checkpoint_transfer = true;
     output = "user.out";
     err = "user.err";
     log =  "userjob.log";
]
```

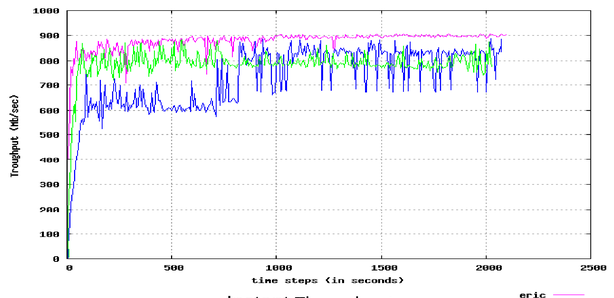
Protocols:

file:/	-> local file
gsiftp://	-> GridFTP
irods://	-> iRODS
Petashare://	-> PetaShare

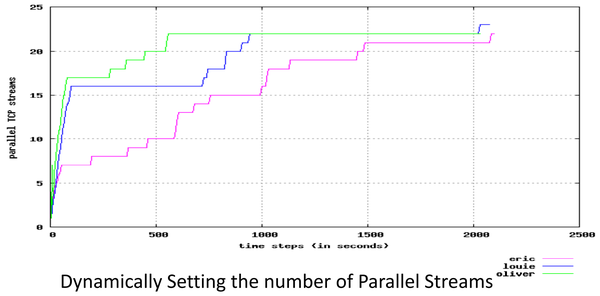
PetaShare Architecture



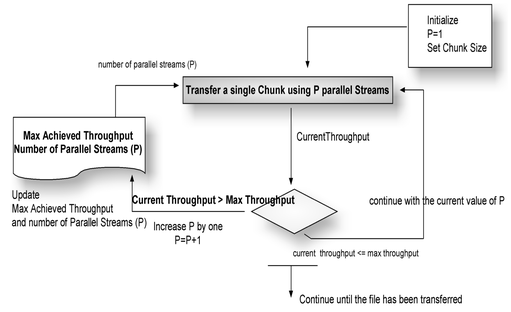
Dynamic Adaptation in Data Transfers



Instant Throughput



Dynamically Setting the number of Parallel Streams



Setting the Parallelism Level inside the Data Transfer Module

A very simple adaptive approach to adjust the level of parallelism on the fly while data transfer is in progress.

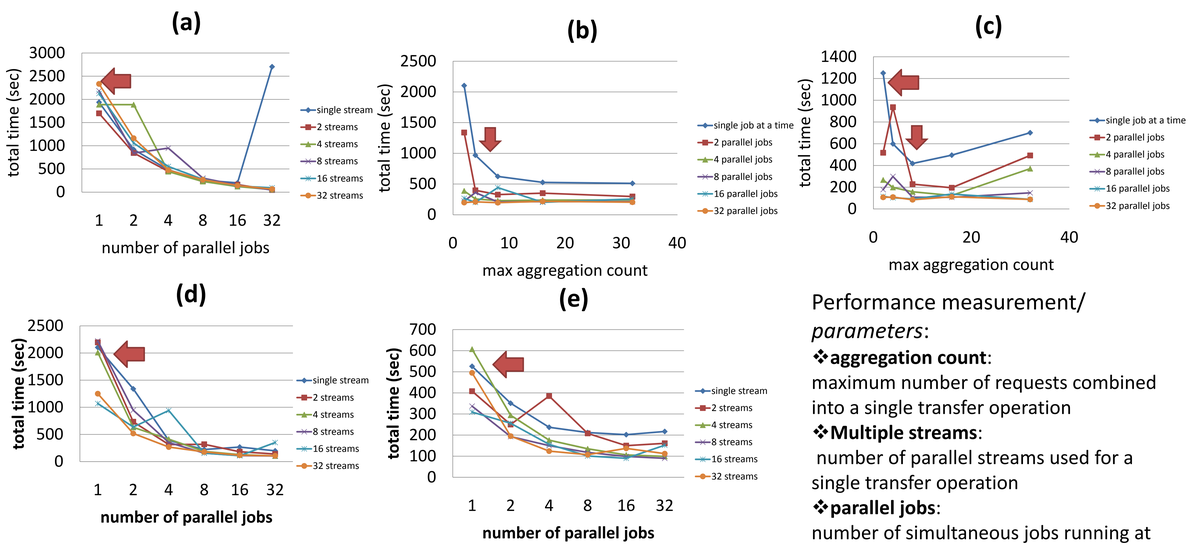
- No external measurement and usage of the historical data to come up with a good estimation for the parallelism
- Reflects the best possible current settings due to the dynamic characteristics of the distributed environment.

Aggregation of Data Placement Jobs

Data placement jobs are combined and processed as a single transfer job (i.e. based on their *source* or *destination* addresses)
We have seen vast performance improvement, especially with small data files.

✓ **Test-set:** 1024 transfer jobs from Ducky to Queenbee (rtt avg 5.129 ms) - 5MB data file per job

Experiments on LONI (Louisiana Optical Network Initiative)



Performance measurement/ *parameters*:

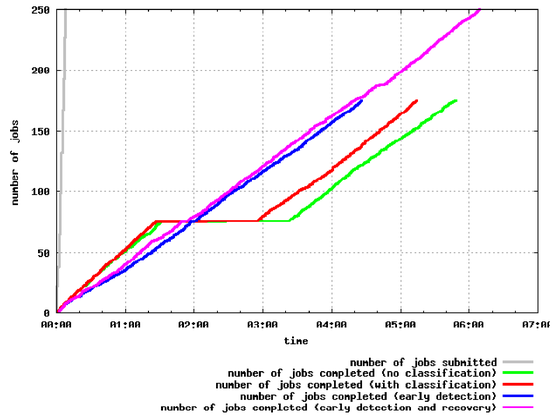
❖**aggregation count**:
maximum number of requests combined into a single transfer operation

❖**Multiple streams**:
number of parallel streams used for a single transfer operation

❖**parallel jobs**:
number of simultaneous jobs running at the same time.

Fig: *Effects of parameters over total transfer time of the test-set*
(a) without job aggregation – number of parallel jobs vs number of multiple streams (b) transfer over single data stream – aggregation count vs number of parallel jobs
(c) transfer over 32 streams – aggregation count vs number of parallel jobs (d) at most 2 requests are aggregated – number of parallel jobs vs multiple streams
(e) at most 16 requests are aggregated – number of parallel jobs vs multiple streams

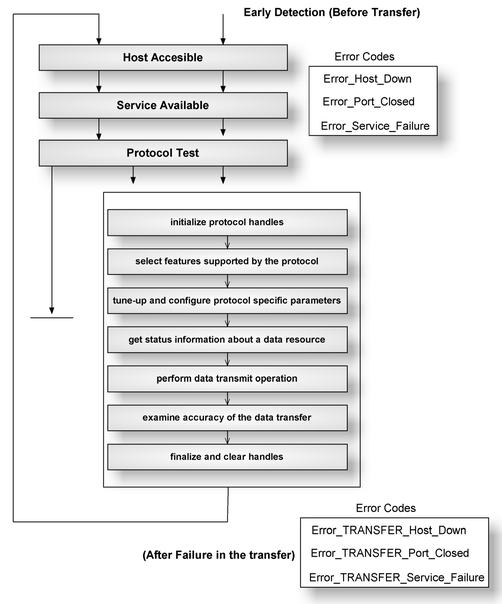
Error Detection and Recovery



stork.globus-url-copy:
In case of a retry from a failure, scheduler informs the transfer module to recover and restart the transfer using the information from a rescue file created by the transfer module.

Stork.globus-url-copy features

`-ckp` | `-checkpoint` -use a rescue file for checkpointing
`-ckpdebug` | `-checkpoint-debug`
`-ckpfile <filename>` | `-checkpoint-file <filename>`
checkpoint filename. Default is "<pid>.rescue"
`-cksm` | `-checksum` >
checksum control after each transfer
`-pchck` | `-port-check`
check network connectivity and availability of the protocol



Ex: rescure file

#-failed_list (1):
gsiftp://dsl-turtle06.csc.lsu.edu/tmp/test/x_ttest2.tar file:///tmp/x_ttest2.tar
#-expanded_url_list (7):
gsiftp://dsl-turtle06.csc.lsu.edu/tmp/test/out file:///tmp/out
gsiftp://dsl-turtle06.csc.lsu.edu/tmp/test/test2/test22/ file:///tmp/test2/test22/
#-transferred_list (2):
gsiftp://dsl-turtle06.csc.lsu.edu/tmp/test/test1/ file:///tmp/test1/
gsiftp://dsl-turtle06.csc.lsu.edu/tmp/test/test2/ file:///tmp/test2/

Acknowledgement: This project is in part sponsored by National Science Foundation, Department of Energy, and Louisiana Board of Regents.