the globus alliance
www.globus.org

FALKON:

A FAST AND LIGHT-WEIGHT TASK EXECUTION FRAMEWORK

Contact: falkon-user@globus.org
http://dev.globus.org/wiki/Incubator/Falkon

Falkon aims to enable the rapid and efficient execution of many tasks on large compute clusters. Falkon integrates multilevel scheduling to separate resource acquisition from task dispatch, and a streamlined dispatcher. As a result, Falkon delivers performance not provided by any other system. Micro-benchmarks show that Falkon throughput (ranging from 100s to 1000s of tasks/sec) and scalability (to 54K executors and 2M queued tasks) are several orders of magnitude better than other systems used in production Grids. Large-scale astronomy and medical applications executed under Falkon by the Swift parallel programming system achieve up to 90% reduction in end-to-end run time, relative to versions that execute tasks via separate scheduler submissions.

Goals:

- Reduce task dispatch time by using a streamlined dispatcher that eliminates support for features such as multiple queues, priorities, accounting, etc.
- Use an adaptive provisioner to acquire and/or release resources as application demand varies.
- Cache data from remote locations to local disk at the computational resource to achieve better application performance and scalability.

Features:

- *Fast:* Throughputs of 487 tasks/sec (stable release), and up to 2500 tasks/sec with the latest optimizations
- *Scalable:* Supports up to 2M queued tasks, and can scale to 54K processors
- *Testbeds:* TeraGrid, TeraPort, Amazon EC2, IBM Blue Gene/L, SiCortex, Workspace Service
- *Dynamic resource provisioning:* (via GRAM4) Allows to trade off resource responsiveness with resource wastage while minimizing the queue wait times
- *Data caching:* Minimizes the use of shared file systems and improve application performance and scalability
- *Web service based architecture*
- *Java-based client API*
- *Support for both polling and notification based events*: Works through firewalls, NATs, and private networks
- *X11 GUI for monitoring system state and automatic graph generation:* Displayable through a web browser

Performance:

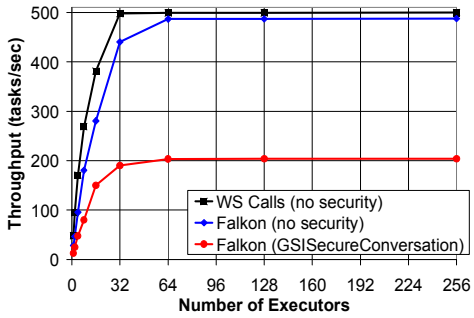

Figure 1: Throughput as function of executors

Table 1: Measured and cited throughput for Falkon, Condor, and PBS

System	Comments	Throughput (tasks/sec)
Falkon (no security)	Dual Xeon 3GHz w/ HT 2GB	487
Falkon (GSISecureConversation)	Dual Xeon 3GHz w/ HT 2GB	204
Condor (v6.7.2)	Dual Xeon 2.4GHz, 4GB	0.49
PBS (v2.1.8)	Dual Xeon 2.4GHz, 4GB	0.45
Condor (v6.7.2) [15]	Quad Xeon 3 GHz, 4GB	2
Condor (v6.8.2) [34]		0.42
Condor (v6.9.3) [34]		11
Condor-J2 [15]	Quad Xeon 3 GHz, 4GB	22
BOINC [19, 20]	Dual Xeon 2.4GHz, 2GB	93

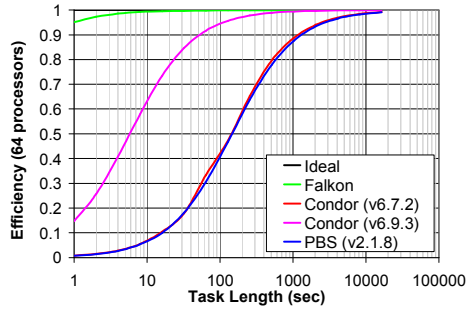

Figure 2: Efficiency of resource usage for varying task lengths on 64 processors comparing Falkon, Condor and PBS

Applications

We have implemented a Falkon provider for the Java CoG Kit 4 abstractions library. This allows software developed against the CoG abstractions library (such as Karajan and Swift) to use Falkon without modifications. We observe reductions in end-to-end run time by as much as 90% when compared to traditional approaches in which applications used batch schedulers directly.

Table 2: Swift applications; all could benefit from Falkon

Application	#Tasks/workflow	#Stages
ATLAS: High Energy Physics Event Simulation	500K	1
fMRI DBIC: AIRSN Image Processing	100s	12
FOAM: Ocean/Atmosphere Model	2000	3
GADU: Genomics	40K	4
HNL: fMRI Aphasia Study	500	4
NVO/NASA: Photorealistic Montage/Morphology	1000s	16
QuarkNet/I2U2: Physics Science Education	10s	3 ~ 6
RadCAD: Radiology Classifier Training	1000s	5
SIDGrid: EEG Wavelet Processing, Gaze Analysis	100s	20
SDSS: Coadd, Cluster Search	40K, 500K	2, 8
SDSS: Stacking, AstroPortal	10Ks ~ 100Ks	2 ~ 4
MolDyn: Molecular Dynamics	1Ks ~ 20Ks	8

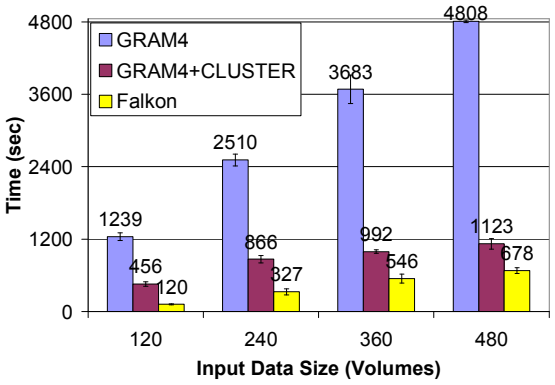

Figure 3: Execution Time for the fMRI Workflow

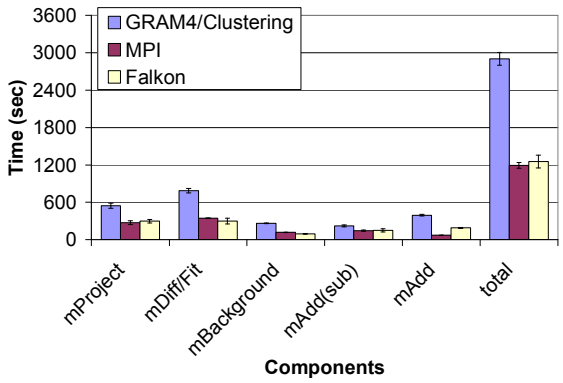

Figure 4: Execution time for Montage application

How You Can Engage and Contribute!

Contribute to the project: To contribute code, documentation, design ideas, and feature requests to the Falkon project, join and post to the several mailing lists available:

- Developer discussion: falkon-dev@globus.org
- User discussion: falkon-user@globus.org
- Commit notifications: falkon-commit@globus.org

Download the most recent version (the current SVN archive will move, check back for new location):

svn co https://svn.ci.uchicago.edu/svn/vdl2/falkon

For more information, please visit: http://dev.globus.org/wiki/Incubator/Falkon

Acknowledgements

This work was supported in part by the NASA Ames Research Center GSRP Grant Number NNA06CB89H and by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Dept. of Energy, under Contract DE-AC02-06CH11357.

Sponsors Include:

Ames Research Center

NASA

TeraGrid EMPOWERING DISCOVERY

THE UNIVERSITY OF CHICAGO

Argonne NATIONAL LABORATORY