**Cloud-Oriented Scheduling Tool Supporting Big Data Processing Workflows**

**Instructor:** Jia Zhang

**Authors:** Claud Wang

**Semester:** Fall 2013

Table of Contents

I. Executive Summary 2

II. Background & Motivation 3

A. NASA HECC Overview 3

1) Front-End Nodes 3

2) Portable Batch System 3

3) Compute Nodes 4

B. Amazon Elastic Compute Cloud 5

C. Vistrails 4

III. Motivation 5

III. Architecture and Design 4

A. Scope 4

B. Concerns 5

C. Principles 5

D. Constraints 5

E. Design Views 6

1) Functional View 6

2) Deployment View 7

3) Development View 9

IV. Prototype Implementation 10

A. VisTrails Plug-In 10

B. Scheduler Server 12

V. Future Work 12

Appendix A: Source Code 13

Appendix B: Installation 13

Appendix C: Module Navigation 13

References 21

# I. Executive Summary

NASA’s High-End Computing Capability (HECC) [1] project is designed to accelerate and enhance scientific discovery and aeronautics research. The mission of this project is to develop a technique that bridges the gap between scientific workflow design tool and NASA’s Supercomputer Center, so that scientists can run their big data experiments at NASA’s Supercomputer Center and Amazon cloud without leaving their design environment. In this technical report, we describe our proposed architecture and the implementation of a working prototype to demonstrate the feasibility of our solution.

# II. Background

The HECC Project, hosted by NASA, is a world-class computing and storage environment for conducting large-scale scientific research experiments to support NASA’s missions. Scientists access the environment by using two-factor (SSH+RSA) logging mechanism to front-end nodes and issue jobs to compute nodes. The NASA Earth Exchange (NEX) [3] facility assists earth scientists in their research by collecting global and high-resolution satellite data and providing high computing systems to share with the geoscience community. Geoscientists use these environmental data sets in conducting their research, such as creating models to predict natural phenomena. Because the information is high-density, comes from all around the world, and encompasses many years, scientists require high computing power to complete their experiments in a reasonable amount of time.

In Fall 2012, CMUSV’s students have started a project to develope a scientific workflow design tool to help scientists run their big data experiments at NASA’s Supercomputer Center and Amazon cloud without leaving their design environment. The work was published in the proceedings of the 2013 IEEE International Workshop on Scientific Workflows. Based on the outstanding previous work, NASA would like to sponsor this project to explore cost-effective workflow scheduling algorithms, in the context of big data processing (including MapReduce) over NASA’s Supercomputer Center and Amazon cloud.

## A. NASA HECC Overview

### 1) Front-End Nodes

The front-end layer of the system contains 14 Pleiades Font-Ends (PFEs) and 2 bridge nodes. These nodes provide environments for users to perform file transfers, file manipulations, and job submissions. Users are required to first log onto secure front-end nodes first with SSH to be able to log on to one of the 14 Pleiades Font-Ends (PFEs) and 2 bridge nodes using RSA authentication.

### 2) Portable Batch System

Pleiades contains the Portable Batch System (PBS) developed by Altair for all compute job submissions, monitoring, and management. PBS adopts job queues to manage pending work and acts as a scheduler. It dispatches jobs to be run on one or more compute nodes, based on factors such as mission shares (a certain percentage of CPU’s on Pleiades are allocated to each NASA mission directorate), job priority, queue priority, and job size. After users log on to the front-end nodes, they are able to issue qsub/qstats command to access the PBS to submit jobs and receive job statuses, respectively.

### 3) Computing Nodes

Four kinds of computing nodes are currently available on Pleiades [1]. Users can specify the node type and process numbers in the PBS script. An overview of the node specifications is as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Node Type** | **Number of Nodes** | **Processors per Node** | **Processor Speed** | **Memory Size per Core** |
| Sandy Bridge | 1,728 | 2 eight-core processors | 2.6 GHz | 2 GB |
| Westmere | 4,608 | 2 six-core processors | 2.93 GHz or 3.06 GHz | 2 GB |
| Nehalem | 1,280 | 2 quad-core processors | 2.93 GHz | 3 GB |
| Harpertown | 4,096 | 2 quad-core processors | 3 GHz | 1 GB |

## B. Amazon Elastic Compute Cloud (Amazon EC2)

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers.

Amazon EC2’s simple web service interface allows the user to obtain and configure capacity with minimal friction. It provides the user with complete control of their computing resources and lets you run on Amazon’s proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing the user to quickly scale capacity, both up and down, as the change of computing requirements.

Amazon EC2 provides a wide range of pricing in computing instances and offers different on-demand instances based on different needs. More detailed information can be found here: http://aws.amazon.com/ec2/#pricing

## C. Vistrails

VisTrails is a scientific workflow management software package used in data-related research. In our project, value is gained by creating a solution to allow VisTrails to directly submit workflows as jobs to be processed in the NASA’s Pleiades system, since high computing power is necessary in processing the scientists’ data-intensive requests.

# III. Motivation

The motivation for our project is to allow scientists to access the HECC supercomputing environment with minimal knowledge of its operational aspects and modification to their workflows. And in the situation that there is no sufficient computing resource in HECC or there is deadline urgency, they can easily switch to Amazon Elastic Compute Cloud (Amazon EC2).

# III. Architecture and Design

## A. Scope

Our project’s scope is a scheduler that helps earth scientists to have a better use of NASA’s high-performance computing layer of the Pleiades system and Amazon EC2. It achieves this goal by analyzing the workflow submitted by the user and providing recommendations regard to time efficiency or cost efficiency. It also helps the scientists to have a better understanding of the estimated duration and cost.

## B. Design Views

### Macintosh HD:Users:claudwang:Downloads:Screen Shot 2013-12-15 at 2.52.09 PM.png

Figure 1. Flow chart of HECC Scheduler Algorithm

### 1) HECC Scheduler

An overview of HECC Scheduler can be found in Figure 1. The algorithm has two main parts: Partition and Scheduling, which can be understood as a divide-and-conquer algorithm.

We categorize workflow tasks to be either a synchronization task or a simple task. A synchronization task is defined as a task that has more than one parent or child task. Other tasks that have only one parent task and child task are simple tasks. Let a *branch* be a set of interdependent simple tasks that are executed sequentially between two synchronization tasks.

In Partition, we try to combine different simple tasks on a branch to be a general task to simplify the problem. Then in Scheduling, we assign deadlines to different tasks, and assign deadline based on the different workload and time constraint of each task.

The biggest constraint for HECC Scheduler is the limited availability of those computing nodes. So given a certain task, the scheduler needs to communicate with the HECC center back and forth to get the current availability of different computing nodes, and estimate a finished time based on both the current availability and those waiting tasks in the queue. If there is expected to be a long waiting time, and the user’s task cannot be finished as expected by any means, the scheduler has the responsibility to let the user know.

The whole algorithm is implemented based on the research work of *Cost-based Scheduling of Scientific Workflow Applications on Utility Grids [1]*. Detail of the algorithm can be found in that paper.

### 2) Amazon Scheduler

An overview of Amazon Scheduler can be found in Figure 2. The whole idea is to keep minimize the final finished time by improving the time cost on the path which causes the most delay (critical path).

So given a workflow, we will start by assigning the lowest computing node to all the tasks, and calculate the estimated finished time for the final task (final deadline). It probably is going to surpass the expected deadline. So we will start the optimization process. The optimization starts from the final task. Because we already have an estimated deadline for each of the tasks, we can tell among all the tasks the final task depend on, which task causes the most delay. We mark that task as in\_path, and repeat this process for that task, finding the next task that causes the most delay. We repeat until we reach the start task. And we will find the critical path by doing so. We then calculate the whole workload of the critical path and pick the computing node that will finish such a workload in the expected deadline. We update all the task in the critical path with this new computing node, and update the final deadline. We examine to see if the final deadline is smaller than the expected deadline. If yes, we stop. Otherwise repeat the optimization until the expected final deadline is met.

The difference between Amazon and HECC problem is that in Amazon EC2, the resource can be viewed as unlimited. Thus we need not to worry about the availability of the computing nodes. And that’s why we have a different solution for Amazon EC2 Scheduling than the HECC one.

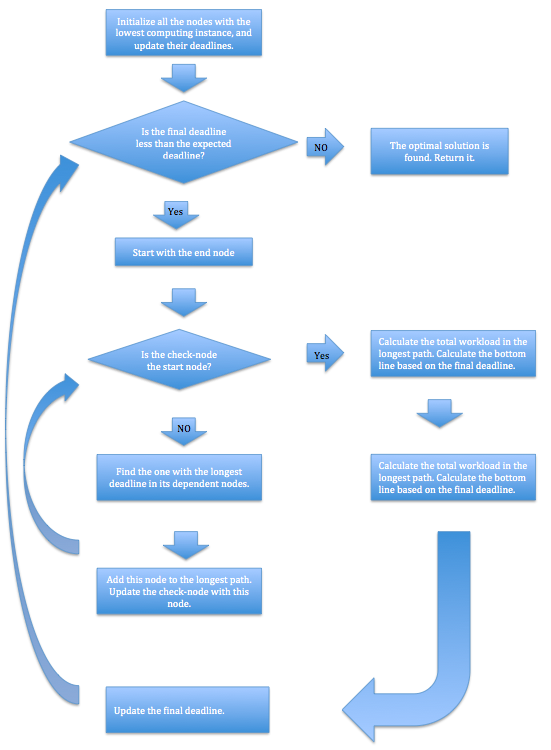


Figure 2. Flow chart of Amazon Scheduler Algorithm

# IV. Prototype Implementation

A prototype of the proposed Schedulers is implemented with Python as a Vistrails plug-in and backend scheduling algorithm. Both for HECC and Amazon EC2.

## A. VisTrails HECC Cost Estimation Plug-In

As described in previous sections, VisTrails is the tool current used by NEX scientists that provides a graphical interface for designing and managing workflows. And you can use the cost estimation by selecting the item in the menu like demonstrated in Figure 3.

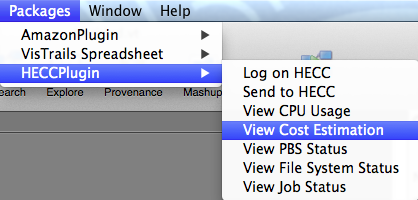


Figure 3. VisTrails Plug-in Menu Items Screenshot

After selection, you will see a node configuration panel as below in Figure 4.

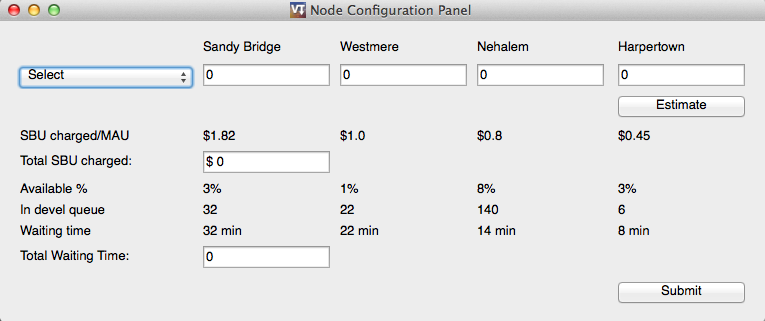


Figure 4. HECC plug-in Node configuration panel

You can select from three kinds of estimation mods: “Performance Efficient”, “Cost Efficient”, or “Manually Set”. “Performance Efficient” will give you the combination of necessary nodes that will finish your task the fastest. “Cost Efficient” will give you the combination of necessary nodes that can finish your task in the expect deadline at the lowest cost. “Manually Set” tells you the total cost given the combination of nodes manually input by you.

## B. VisTrails Amazon Cost Estimation Plug-In

Similarly, we built a plug-in for Amazon EC2, and you can use the cost estimation by selecting the item in the menu like demonstrated in Figure 5.

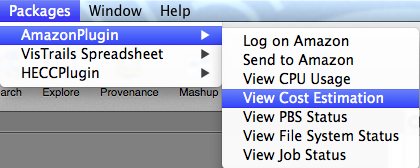


Figure 5. VisTrails Plug-in Menu Items Screenshot

After selection, you will see a node configuration panel as below in Figure 6.

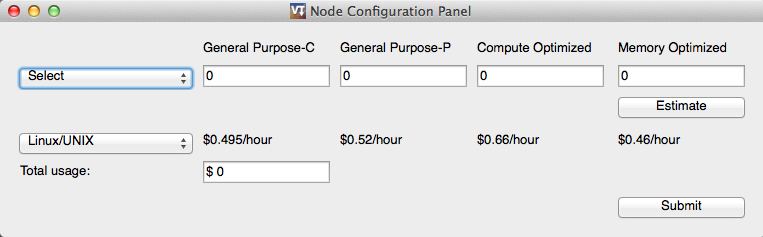


Figure 6. HECC plug-in Node configuration panel

You can select from three kinds of estimation mods: “Performance Efficient”, “Cost Efficient”, or “Manually Set”. “Performance Efficient” will give you the combination of necessary nodes that will finish your task the fastest. “Cost Efficient” will give you the combination of necessary nodes that can finish your task in the expect deadline at the lowest cost. “Manually Set” tells you the total cost given the combination of nodes manually input by you.

You can also select the platform of the computing node instance to see different price options.

# V. Future Work

Since right now we only achieved a minimal viable version of the Cloud Scheduler, there are a plenty of things remain to be done in the future:

First, the Cloud Scheduler works well with simple workflows right now. However more complicated workflow cases need to be tested in the future. And we can optimize the scheduling algorithm based on real life workflow and more complicated test cases. Second, the Cloud Scheduler reads in workflow and relevant information from JSON file. To complete the functionality, it needs to be able to read in workflow information from Vistrail. Third, we are looking forward to have the opportunity to gain more accessibility to NASA’s resources, because that way we can provide more support in our implementation.

We hope this prototype is a good start and proof of our architectural design for the project. We believe that our work is worthwhile to expand to another level and we can help NASA increase the number of scientists adopting their computing ecosystem.

# 

# Appendix A: Source Code

# <https://github.com/cmusv-sc/WorkflowScheduling-Claud>

# Appendix B: Installation

Directions to enable VisTrails to run in batch mode:

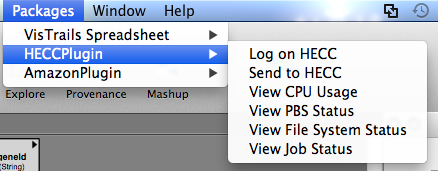
1. sudo apt-get install xvfb python-matplotlib python-suds
2. wget<http://downloads.sourceforge.net/project/vistrails/vistrails/v2.0.1/vistrails-src-2.0.1-5e35e2b83b90.tar.gz>
3. tar -zxvf [vistrails-src-2.0.1-5e35e2b83b90.tar.gz](http://downloads.sourceforge.net/project/vistrails/vistrails/v2.0.1/vistrails-src-2.0.1-5e35e2b83b90.tar.gz)
4. Running a workflow
   1. See scripts/run\_vistrails\_batch\_xvfb.sh for more details
5. Debugging VisTrails installing or runtime issues:
   1. Check VisTrails’s log file: ~/.vistrails/vistrails\_2\_0\_1.log

Installing VisTrails and Modules:

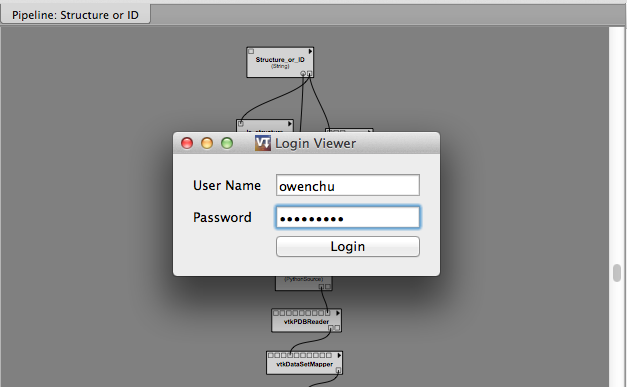
1. <http://www.vistrails.org/index.php/Downloads>
2. Download the HECCPlugin and AmazonPlugin folders from Git
3. Copy the two folders into ~/.vistrails/userpackages
4. On VisTrails, go to VisTrails > Preferences and click on the Module Packages tab
5. Enable the HECCPlugin and AmazonPlugin modules

# Appendix C: Module Navigation

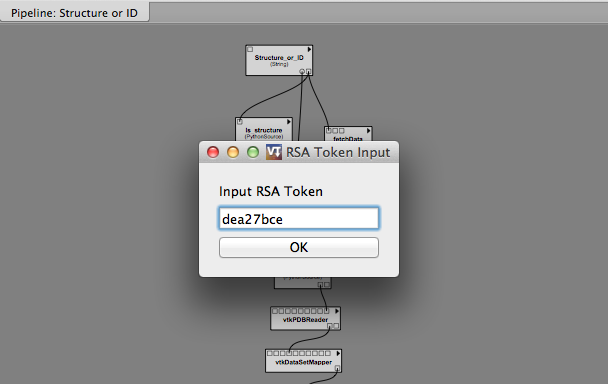
HECC module menu items:



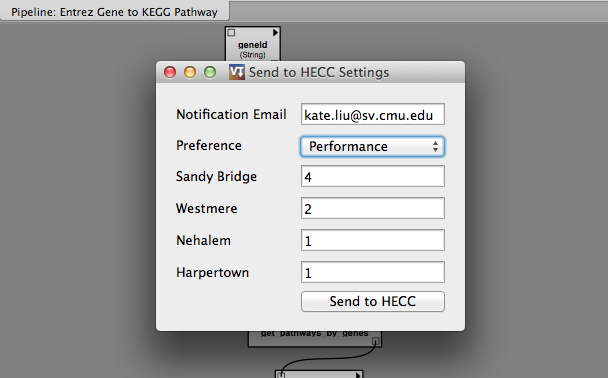
Login window for HECC. This window will show up when the user selects “Log on HECC”.



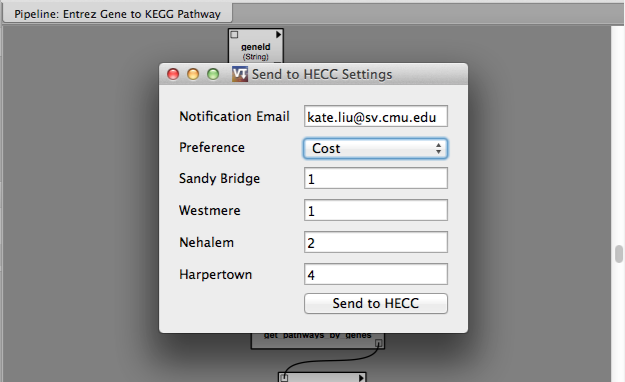
RSA token window. The user enters the code that appears on their RSA token into this window. This window will show up after the user enters account information from the user login window or selects “Send to HECC”.

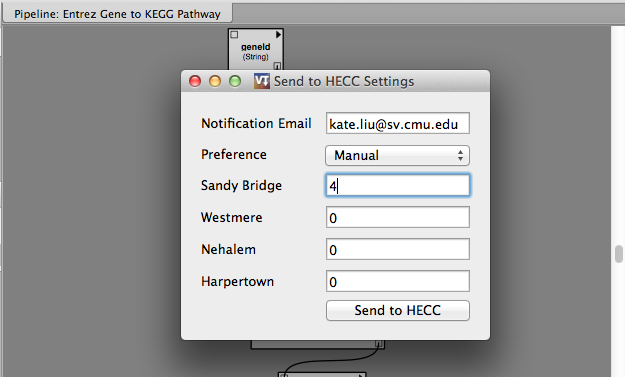


HECC settings window. This window will show up when the user selects “Send to HECC” and puts their RSA token. The user can choose the email address they want to receive the job completion notification. They can also select from 3 different preferences, which will change the number of nodes in the ‘Sandy Bridge’, ‘Westmere’, ‘Nehalem’, and ‘Harpertown’ inputs. The ‘Performance’ preference is selected below. When the user is done, the user can press ‘Send to HECC’ to send the job to be run remotely.

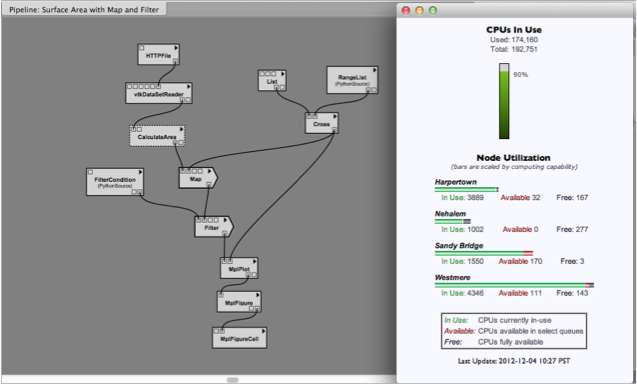


The ‘Cost’ preference is selected below. The functionality is the same as mentioned above.

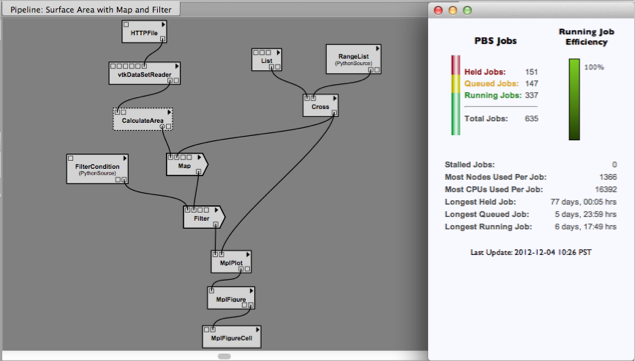




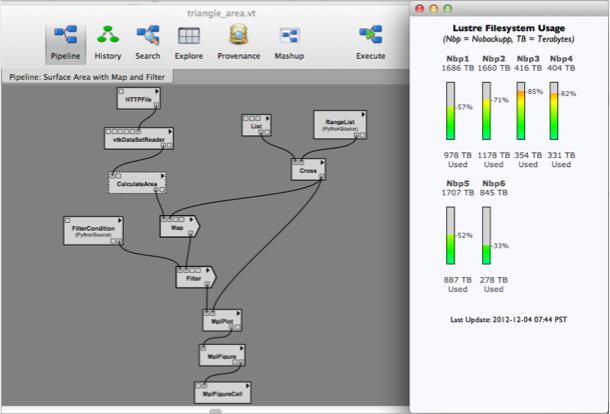
This window will show up when the user selects ‘View CPU Usage’. It shows the usages of the different compute nodes in the Pleiades system.



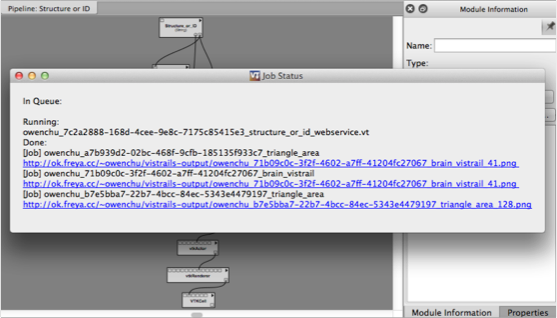
This window will show up when the user selects ‘View PBS Status’. It shows statuses and information of jobs handled by PBS in the Pleiades system.



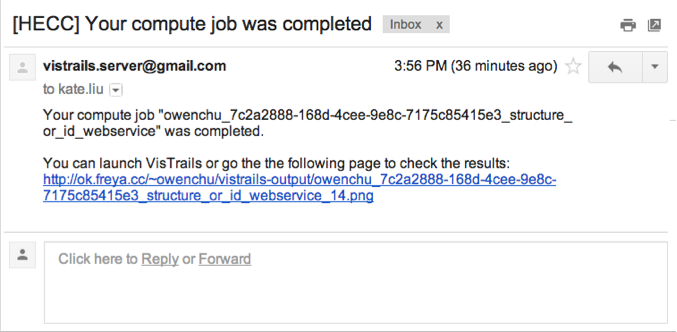
This window will show up when the user selects ‘View File System Status’. It shows the statuses of the file system nodes.



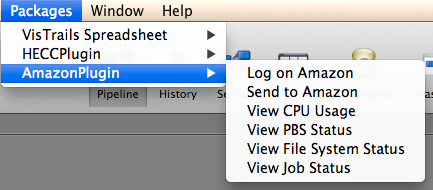
This window shows up when the user selects ‘View Job Status’ after a job has been submitted to HECC. It provides links to results that are viewable online.



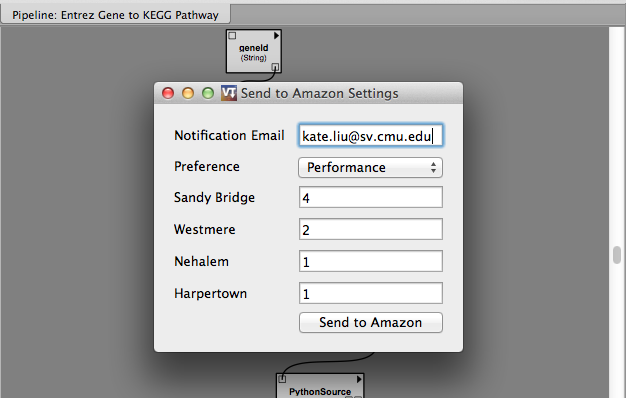
This is an example of the email a user receives when a job is completed.



Amazon module menu items:



This window shows up when the user selects ‘Send to Amazon’. The functionality is the same as ‘Send to HECC’ as mentioned above, but instead of the job running on NASA’s HECC, the job is run on an Amazon EC2 instance. In this case, a RSA token is not required.



# References

[1] NASA High-End Computing Capability. Web. <http://www.nas.nasa.gov/hecc/>.

[2] VisTrails Wiki. Web. <http://www.vistrails.org/index.php/Main\_Page>.

[3] NEX - NASA Earth Exchange. Web. <https://c3.nasa.gov/nex/>.

[4] Writing VisTrails Packages." Web. <<http://www.vistrails.org/usersguide/dev/html/packages.html>>.