A Comparison of Parameter Study Creation and Job Submission Tools

Adrian DeVivo, Maurice Yarrow, and Karen M. McCann

Computer Sciences Corporation, Mail Stop T27A-1
NASA Ames Research Center, Moffett Field, CA 94035
{devivo,yarrow,mccann}@nas.nasa.gov
NAS Technical Number: NAS-01-002

Abstract

We consider the differences between the available general purpose parameter study and job submission tools. These tools necessarily share many features, but frequently with differences in the way they are designed and implemented. For this class of features, we will only briefly outline the essential differences. However, we will focus on the unique features which distinguish the ILab parameter study and job submission tool from other packages, and which make the ILab tool easier and more suitable for use in our research and engineering environment.

Motivation and Background

The generation and submission of physical parametric studies involving simulations with engineering programs is a painstaking task. The manual creation of such studies typically first involves editing a large number of ASCII files containing the physical parameters of interest. Then, the resultant individual simulation job runs must be organized and instantiated. Most often, the required computations will actually occur on a supercomputer distinct from the computer utilized for creating the study. Organizational requirements will usually require the assembly of a special directory structure on the target computer systems, and the importation to this directory structure of the many files required by the individual jobs of the parameter study. In addition, most supercomputer centers require that submitted jobs go through a job scheduler (queuing system), and thus parameter study tools must coordinate with the scheduler. Supporting these tasks are the most important of the functionalities or general requirements to be expected from a parameter study tool. In addition, specific user needs require in certain cases specific features, without which all the other features available may not suffice to make a parameter study tool usable.

We have developed the ILab tool to address the needs of not only general and simple parameter study users, but also the specialized needs found among some of our aeronautical engineers. We will concentrate on a description of these important features in this report.

Tool Descriptions

ILab

"ILab: The Information Power Grid Virtual Laboratory" is a Perl program which provides a Perl/Tk user interface and generates input files and shell scripts for parameter studies being run on the Grid. (Note that the expression "Grid" is now being used to describe the world-wide network of supercomputers that scientists are using to run scientific problems that involve data sets too large to fit on desktop systems.) This program has an extensive user interface; we already have about 23,000 lines of Perl, and we anticipate about 50,000 lines when all options are implemented. We have adopted an object-oriented approach: all data and operations are organized into packages in order to speed up development and debugging. ILab's "container" or "document" object is called an "Experiment": it contains all the

information - data and file paths - that is necessary to define a complex series of repeated and/or sequenced and/or branching processes. This object is serialized to and from disk, for user's convenience and re-use. At run time, ILab's current "Experiment" is used to generate required input files and shell scripts, create directories, copy data files, and then both initiate and monitor the execution of all processes in the Experiment.[1]

Nimrod/Clustor and Nimrod/G

Nimrod, Nimrod/G, and Clustor [2][4] are historically related tools which create parameter studies and handle job submission. Employing the resources of the Globus Grid toolkit, Nimrod/G performs tasks such as: scheduling of a job, resource discovery, remote storage, and a sort of systems passport (user authentication). Nimrod/G employs the functionality of "computational economy"[4]: this is a process that attempts to negotiate for time slots of adequate size to complete the job. The user can also negotiate resources and make a decision as to when the job can be completed and what the cost will be. When creating a parameter study the user may need to program a special internal script in a Clustor designed metalanguage. Clustor does not support visual parameterization: manual editing of data files is necessary to prepare them for parameterization; subsequently, parameterized input files can be generated automatically.

Condor

Condor[5][8], a distributed job launcher, utilizes idle system cycles on systems in the "Condor pool". The Condor pool consists of systems running Condor software on a LAN, WAN, or over the internet (not behind a firewall). Condor monitors systems for specific user defined parameters such as minimum idle time and wall time. When those parameters are met it sends a job from its queue to be processed by the idle system. At regular intervals while the job is running, Condor sets checkpoints to save its progress. In the event that the job is stopped by the host system's user Condor will hopefully have someplace to pick up when the job is sent to another machine. The primary drawback of Condor is that it can not run parallel jobs in order to speed up the completion time of a project. There are also some security issues to be addressed, namely " ...a sufficiently malicious and clever user could cause problems by doing local system calls on the executing machine."[5] Condor is appropriate for simple process execution, where processes are executed serially but use up many computing cycles. Condor does not create parameter studies.

AppLeS/APST

AppLeS (Application-Level scheduler)[6][7] is a job submission tool which evaluates everything about the system in terms of its impact on the application. This is based upon predictions of the state of the system when the application will be executing. Network status information is received from Network Weather Service (NWS), which can "forecast the performance various network and computational resources can deliver over a given time interval"[6]. AppLeS also does not do parameterization; it is a "performance predictor" tool that also does job submission. APST, "AppLeS Parameter Sweep Template", launches preexisting parameter studies.

Chart/Description

The following chart outlines the functionality of ILab and several other parameter study and scheduler utilities. Clustor, Nimrod, and Nimrod/G are the closest comparison to ILab's capabilities, since they too can create parameter studies in addition to submitting jobs.

Chart Key (i.e., what the terms signify in the context of these comparisons)

Complex Processes : a set of processes where each process depends on the input from the previous process; may or may not involve branching and loops.
Multiple Input Files : more than one file is required for running a process.
Process Looping : the repetition of a process until some condition (typically, a program return value, program internal value, or the existence of an output file) is satisfied.
GUI Parameterization : (Visual parameterization) a graphical screen in which user can choose fields to parameterize by highlighting the fields with a mouse, and then entering a list of values for that field.

	ILab	Nimrod/ Clustor	Nimrod-G	Person. Condor	APST
Complex Processes	Yes*	No	No	N/A	N/A
Multiple Input Files	Yes	Yes	Yes	N/A	N/A
Process Looping	Yes*	No	No	N/A	N/A
GUI Parameterization	Yes	No	Yes	N/A	N/A
Multiple Job Models	Yes	No	Yes	N/A	N/A
GUI	Yes	Yes	Yes	?	?
Programming By User	No	Yes	Yes	N/A	N/A
Built-in Metalanguage	No	Simple	Simple	N/A	N/A
PC/Unix Compatible	Yes	No	Yes	Yes	?
CAD Interface	Yes*	No	No	N/A	N/A
OOP	Yes	No	No	?	?
Plugins	Yes*	No	No	N/A	N/A
Monitor Capacity	Yes	Yes	Yes	Yes	Yes
Single Job capacity	Yes	Yes	Yes	Yes	Yes
Secretarial functions	Yes	No	No	No	No
Graphics/Video screens	Yes	No	No	No	No
Archiving	Yes*	No	No	No	?
Problem-Solving Env	Yes	No	No	No	No
Schedulers	Yes	?	Yes	?	Yes
Metacomputing Env	Yes	No	Yes	Yes	Yes
Parallel Platforms	Yes	?	?	No	N/A
Globus Enabled	Yes	No	Yes	No	Yes
Restart	Yes*	No	No	No	No
* Under Construction					

Comparison Table

Multiple Job Models : the ability to incorporate any of several middleware packages during process execution (e.g., Globus, Condor, Legion, PBS, LSF, etc.)
GUI : Graphical User Interface where all input can be done with mouse clicks and/or keyboard accelerators. Typically, GUIs include on-line HELP and error-checking for all input.
Programming by User : user is required to create some variety of job control language script or "plan file"; actual "programming" (creation of variables and/or control structures) may be required.
Built-in Metalanguage : a set of commands with associated options that can be used to either run a program and/or describe input data.
PC/UNIX Compatibility : program can run without modification on both PC and UNIX systems.
CAD Interface : "Computer-Assisted Design" : special graphical interfaces that allow user to describe data or processes by placing and connecting icons or some form of pictures.
OOP : "Object-Oriented Programming" : a program's data and operations ("methods") are organized into structures (either "classes" or "packages") that allow for easy extensibility and re-use of code.
Plug-Ins : the ability of a program to accept external programmatic tools or capabilities that are not supplied by the original program by default.
Monitor Capacity : the ability to visually indicate the status of jobs during execution.
Single Job Capacity : the ability to create and launch a single process as well as multiple processes.
Secretarial functions : the ability to keep track of sets of files, as well as archive and edit the files. Also, ability to keep a record of user operations performed (audit trail).
Graphics/Video Screens : special screens dedicated to graphical processing of output files.
Archiving : the ability to move files to a mass storage facility, and also to restore from the same facility.
Schedulers : the ability to address specific scheduling programs that are installed on supercomputing systems. (Note that this implies an internal list of scheduler commands and variables.)
Metacomputing Environment : specific middleware packages (such as Globus, Condor, and Legion) that manage job submission, typically onto remote computing resources. (Implies internal lists of middleware commands and/or variables and/or status features.)
Parallel Platforms : the ability to generate specific commands that utilize the parallelizing ability of given supercomputing systems. (Note that these commands usually involve the invocation of some parallel library. e.g., MPI.)
Globus Enabled : issues Globus commands to run processes.
Restart : enables a user to divide a job up by chunks of timesteps, so that one chunk picks up where the previous left off. (See sub-section below "Job Restart Capability")

Below are descriptions of features unique to ILab

Ease of Use

In addition to the efficiency with which the study can be performed, the simplicity with which the user can create a parameter study is an issue. Under Nimrod/Clustor and Nimrod/G, job submission may require the user to write a "plan" file. This file contains two sections: a parameter section and a tasks section. Since Condor and AppLeS do not actually do parameter studies they can not be compared in this context. ILab quickly executes a job or jobs, using a series of user-friendly "Wizard" dialog windows. When the parameter study is launched, it then creates the necessary directories and sub-directories to contain the parameter study scripts, data, and output, on the chosen set of systems which can be local and/or remote.

Job Restart Capability

Parameter studies and job scheduling are supported by several other job handler packages, yet ILab includes options not offered by any of them. ILab's set of restart options is one such feature. The ability to restart is different from the act of setting check points to save work accomplished. Rather, a restart allows a user to divide a job up by chunks of timesteps, so that one chunk picks up where the previous left off. Each time that a restart is run, parameters can be changed by the user to aid in a stable execution until a solution is achieved. In addition, a restart capability allows the user to execute long simulation runs on systems controlled by schedulers which restrict time allocations.

A restart capability becomes especially useful in certain circumstances, especially those encountered by some of our aeronautical engineers. There are two primary reasons a researcher would need such a capability:

1) Ramping solver parameters. Unless solver variables/properties are "ramped" or carefully steered, the case may become unstable ("blow up") because of the sensitivity of the flow-solver to such solver parameters as time step and "artificial" dissipation. By allowing for a restart with solver parameter ramping, stable progression to the solution is achieved.
2) Split up a job to run in segments Another frequent problem encountered is the imposition of time restrictions created by system schedulers. If a job cannot reach completion within the scheduler's time restrictions, then data could be lost and also jobs will have to be completed by running them in separate segments. The restart capability allows users with especially large jobs to plan around the time restrictions and complete their jobs, despite limitations enforced by the scheduler. This can be done by setting the number of time steps that will be processed in a given segment. The job is then restarted, beginning again where the previous time steps left off. This process continues until the final iteration is reached. Instead of waiting for a large chunk of time to become available, the user can spread the job out over smaller time slots and set the number of time steps accordingly. This process would lead to a significantly shorter waiting period before restart startup, although the user would have to schedule more frequent restarts. The ILab job restart capability automates this variety of job submission. The Restart feature is still under development.

ILAB's CAD (Computer Assisted Design) screen : Directed Graph

Another noteworthy feature of ILab is its CAD screen. This screen displays a "directed graph", where the graph nodes are processes in the user's Experiment. The icon units are : one Experiment data icon, and for each Process in the Experiment, a Process data icon, a file list icon, and a file handling icon. These icons are connected by arrows which signify the flow of execution within the Experiment.

This screen has a dual purpose : 1) User can generate the directed graph from pre-existing Experiment data, as an organizational aid in Experiment creation, execution, and tracking.
2) User can create the graph by placing icons, and then, after entering logistical data for each process represented by an icon, can generate Experiment data and save it to disk and/or execute the Experiment.

The most important functionality addressed by ILab's CAD screen is the creation and visualization of complicated execution paths, including branching and conditional loops. Branching can be caused by multiple levels of parameterization, or by repeating or splitting one process output to several subsequent processes. Conditional loops are caused by repeated execution of Experiment processes, where the repeat is terminated by either file existence or a process return value. The CAD screen is still under development.

Visual Parameterization Capability

ILab provides a dialog screen, called "Edit Parameters", that gives users a quick, easy, and intuitive way to parameterize an ASCII input file. This screen contains a Perl/Tk "Text" widget, which displays the input file to be parameterized, and a hierarchical list box, which lists the names of all entered parameters, and their entered values. First, user will select and mark a field that is to be parameterized inside the Text widget; for example, in the input file line "reynum=1.5e6", user would highlight the characters "1.5e6" by holding down the left mouse button and dragging; this indicates that the characters "1.5e6" represent a field that will be parameterized. Then, user hits the "Parameterize" button : this pops up an entry dialog allowing user to enter either a list of values, or a minimum/maximum/increment (three numbers separated by slashes; when this is entered, ILab creates the list of values.) For example, user might enter "1.2e6 1.3e6 1.5e6", indicating that variable "reynum" is to take on these three values, in three separate copies of the input file. When user hits "OK" from this dialog, the field is added to the ILab's parameter list, the values entered appear in the left-hand side hierarchical list (along with a default name for the parameter, that user can change), and the text chosen is then highlighted in a contrasting color. User can then right-mouse on the characters "1.5e6" (which now appear in red on grey) and get a pop-up menu allowing user to edit the values or delete the parameterization for this particular set of characters. Hitting the "Generate" button then generates n copies of the displayed input file, with the appropriately sequenced and replaced parameter values, where "n" represents the product of the number of values for each parameter. For instance, if user parameterizes two fields, and enters three values for each field, then a total of 9 file copies will be generated, and 2 lists of 3 values each appear in the hierarchical list. User can also edit, delete, and name parameter values from buttons located underneath the hierarchical list.

Users find that this visual parameterization capability greatly simplifies and expedites the process of creating multi-dimensional parametric studies.

Summary

Several features of ILab make it superior to other parameter study utilities.

First, ILab is "quick and easy" to use. There is no programming required to create or run a parameter study: all the scripts, input files, and directories are generated and handled by ILab, and then ILab submits and monitors the scripts in order to run user's sequence of jobs.

Second, in terms of job submission and execution, ILab is a high-throughput distributed launcher that can take advantage of multiple processors, so that all Experiment jobs can be completed significantly faster.

Third, ILab has an advanced on-line help system, which has a display, an index, and a search function; this help is available from all ILab screens.

Fourth, ILab's restart functionality tends to get a job started more quickly, by accepting shorter time-slots; user can also parameterize especially large jobs which would have exceeded scheduler maximum time limits. Additionally, restarting can lead to more stable runs when solver parameters are gently incremented for a smooth run.

Fifth, ILab's CAD screen is helpful in organizing user's sequence of jobs, since flow execution is displayed in a directed graph that can be printed and saved, and interpreted into the appropriate sequence of operations.

Finally, ILab's OOP organization of all data into a serializable Experiment package makes it very easy for users to edit previously created Experiments and run them again.

References

[1] Maurice Yarrow, Karen McCann, Rupak Biswas, and Rob F. Van der Wijngaart; An Advanced User Interface Approach for Complex Parameter Study Process Specification on the Information Power Grid p.146 Grid Computing - GRID 2000, First IEEE/ACM International Workshop Bangalore, India, December 17, 2000 Proceedings, ISBN 3-540-41403-7 Springer-Verlag Berlin Heidelberg New York.

[2] David Abramson, Jon Giddy, and Lew Kotler; "High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?" (2000). http://citeseer.nj.nec.com/abramson00high.html

[3] Henri Casanova, Graziano Obertelli, Francine Berman, Richard Wolski; "The AppLeS Parameter Sweep Template: User-Level Middleware for the Grid" in Proceedings of the 9th Heterogeneous Computing workshop (HCW'2000).

[4] Rajkkumar Buyya, David Abramson, and Jonathan Giddy; Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computation Grid, HPC ASIA '2000, the 4th International Conference on High Performance Computing in Asia-Pacific Region, Beijing, China, IEEEE Computer Society Press, USA 2000.

[5] Allan Bricker, Michael Litzkow, and Miron Livny; "Condor Technical Summary" (1991), http://citeseer.nj.nec.com/briker91condor.html

[6] Francine Berman, Richard Wolski; The AppLeS Project: A Status Report (1997), http://citeseer.nj.nec.com/berman97apple.html

[7] Network Weather Service, http://nws.npaci.edu/NWS/

[8] Condor Team, University of Wisconsin-Madison 8/22/2000, Condor Version 6.1.15 Manual.