Note Taker Checklist Form -*MSRI*

Name : Rob Stapleton

E-mail Address/ Phone #: jrstaple@nsu.edu

Talk Title and Workshop assigned to: Parallel
Interactive Computing in Support of Research in Algebra, Geometry

Lecturer (Full name): Yi Qiang

Date & Time of Event: 11:30 a.m. Jan 31, 2007

Check List:

(✓) Introduce yourself to the lecturer prior to lecture. Tell them that you will be the note taker, and that you will need to make copies of their own notes, if any.

(✓) Obtain all presentation materials from lecturer (i.e. Power Point files, etc). This can be done either before the lecture is to begin or after the lecture; please make arrangements with the lecturer as to when you can do this.

(✓) Take down all notes from media provided (blackboard, overhead, etc.)

(✓) Gather all other lecture materials (i.e. Handouts, etc.)

(✓) Scan all materials on PDF scanner in 2nd floor lab (assistance can be provided by Computing Staff) – Scan this sheet first, then materials. In the subject heading, enter the name of the speaker and date of their talk.

Please do **NOT** use **pencil** or colored pens other than black when taking notes as the scanner has a difficult time scanning pencil and other colors.

Please fill in the following after the lecture is done:

1. List 6-12 lecture keywords: parallel, interactive, SAGE, DSAGE, BOINC, Python, Twisted, ZODB

2. Please summarize the lecture in 5 or less sentences.
Even with a tool as flexible and powerful as SAGE, properly implementing a distributable (parallelised) version to accomplish the same tasks as BOINC is nontrivial

Once the materials on check list above are gathered, please scan ALL materials and send to the Computing Department. Return this form to Larry Patague, Head of Computing (rm 214)

11:30 a.m.
Distributed SAGE
Yi Qiang

Main Objectives: Accessible, scalable, cross-platform (SAGE-dependant), uses local and non-local resources.

Many times, the computing power will be there, but there will be no infrastructure to take use of those resources. "Department of Hidden Resources"

Suited for coarse parallel tasks. NOT "embarassingly" parallelizable problems. Proudly parallelizable.
Should have low requirements for synchronization

A Worker in DSAGE consists of a monitor, which spawns multiple instances of SAGE to work on the tasks.

Technologies:

- Python
- Twisted ("Build the engine of your Internet")
- ZODB ("Zope Object Database")

Why DSAGE?

Xgrid, BOINC, Chainsaw (IPython 1), et al. seem to do the same things.

These packages don't seem to make sense for the types of jobs the casual user (us?) may want to do.

Xgrid knows nothing about math/science, proprietary, and only works on OS X

BOINC is suitable for one massive job.

Chainsaw has no fault tolerance or authentication

Fault Tolerance

Servers, workers, or clients can disappear at any time. These factors must be taken into account in order to keep jobs running smoothly.

Security

- SSL communication by default
- Public key authentication of clients
- Run workers in a virtual machine (in the near future)

Applications

Implemented Distributed Functions (such as Ray-Tracing or Integer Factorization)

Many other possibilities.

Distributed Factorization (qsieve just implemented)

DSAGE (8 processors) factored 11^150+1 in 2 minutes, SAGE factored it in 4 hours

It's actually easy to write this yourself (see slides)

The number of workers is dynamic. Jobs cannot be sent to specific workers--they're just sent to the server, and the server passes them around.

Since SAGE is based upon Python, makes use of Python's pickling to pass information from server to workers and back.

Future Development

Tracking and scoring of users (similar to SETI@home)

Integrity checking schemes (especially an allowance for redundancy)

Run workers SETI@home-style

Has been included in SAGE since version 1.7

Latest dsage is version 0.2

slides

~~dates~~ or schedule page

(black background.