BOINC development

David P. Anderson

University of California, Berkeley

May 2024

Requests to projects

- Use current server software
  - don’t modify BOINC files!
- Use simplified attach
- Communicate with volunteers
- Internationalize your web site
- Publicize your project
- Publicize BOINC

AI (large neural networks)

- Increasingly relevant to science
- Can use volunteer computing to train large models?
  - NPUs
- More generally: support parallel computing in BOINC

AI startups

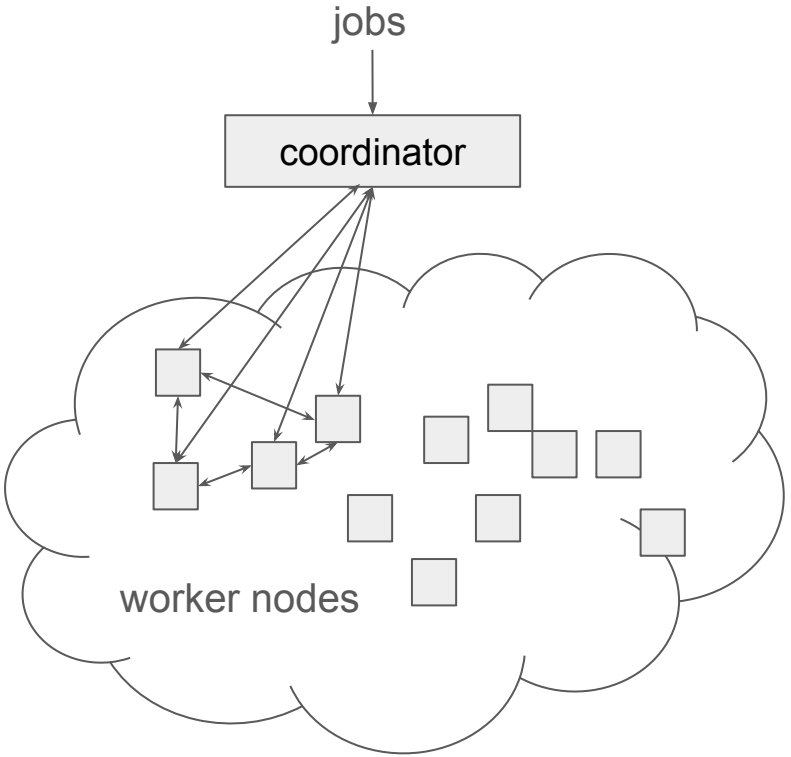
- Face other problems:
  - data privacy
  - financial / incentives
  - competition (e.g. Amazon)

Parallel distributed computing

- Lots of worker nodes run simultaneously
- Nodes can communicate with each other (perhaps indirectly)
- Examples
  - MPI programs (solve differential equations on grids)
  - Map/Reduce algorithms (e.g. web search)
  - Neural net training
- Normally run on data center computers
  - identical, trusted, fast interconnect

Parallel applications on volunteered computers

- Need to deal with
  - heterogeneity
  - intermittent availability
  - possibly slow communication
- General architecture
  - Coordinator
    - track available worker nodes
    - handles job requests
  - For each job
    - identify initial set of workers
    - send commands to workers
    - update worker set as needed

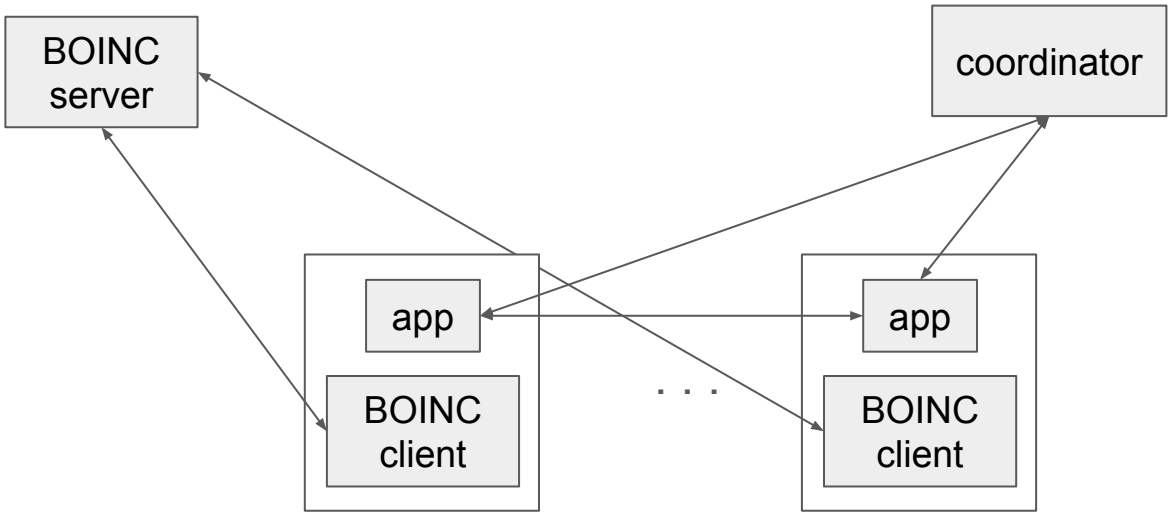


Key features of BOINC

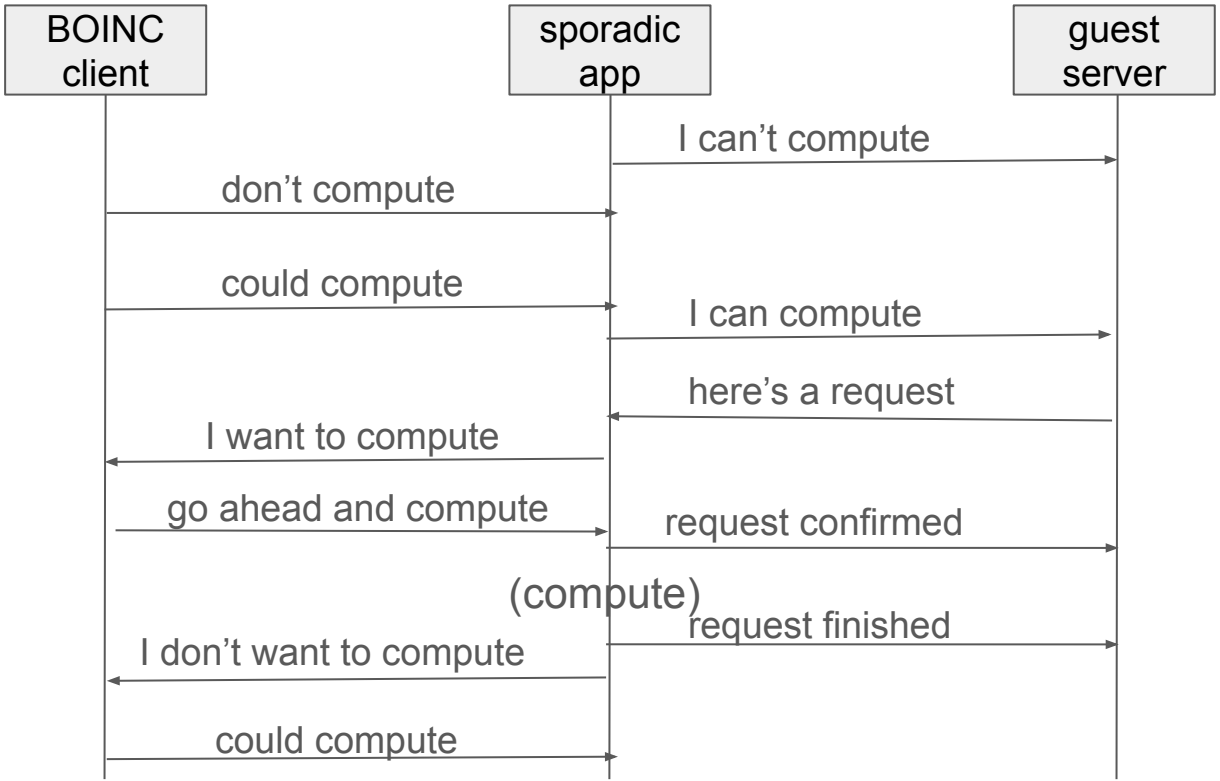
- Manages computing on devices, enforce prefs
  - *needed for parallel computing too*
- Matches app versions to devices (plan classes)
  - *needed for parallel computing too*
- Batch queueing system
  - ***useless for parallel computing- need a new scheduling model***

New ‘sporadic app’ feature

- The parallel distributed system runs as a ‘guest’ on BOINC
- When the guest system wants to use a node, it can do so immediately



Negotiation (per processor type)



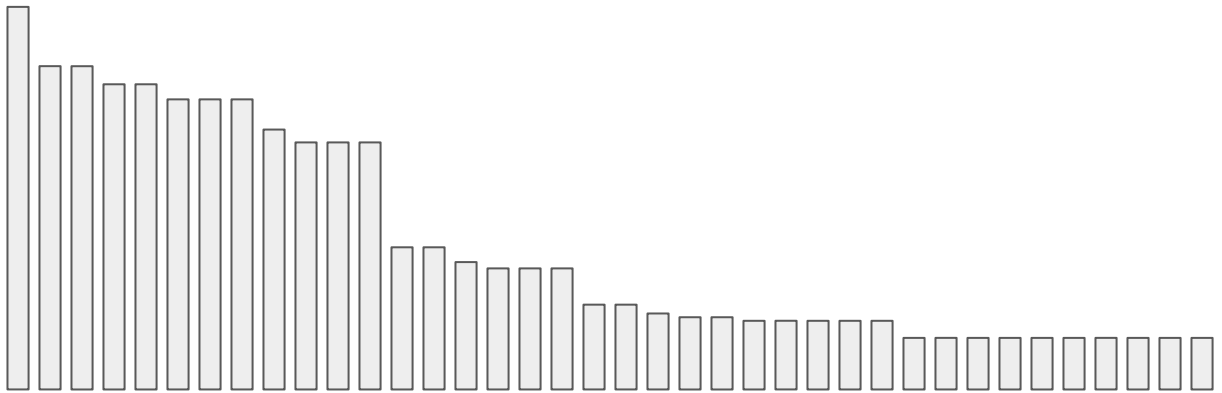
Using heterogeneous resources for synchronous computing

- Divide job into N (synchronous) level 1 sub-jobs
- Divide each level 1 sub-job into some number of (synchronous) level 2 sub-jobs
- To handle a job, the coordinator forms
  - ‘team’: a group of processors with about the same speed; each team handles a level 1 job
  - ‘league’: a set of teams, all with about the same total speed

League formation

‘team’: a group of processors with about the same speed

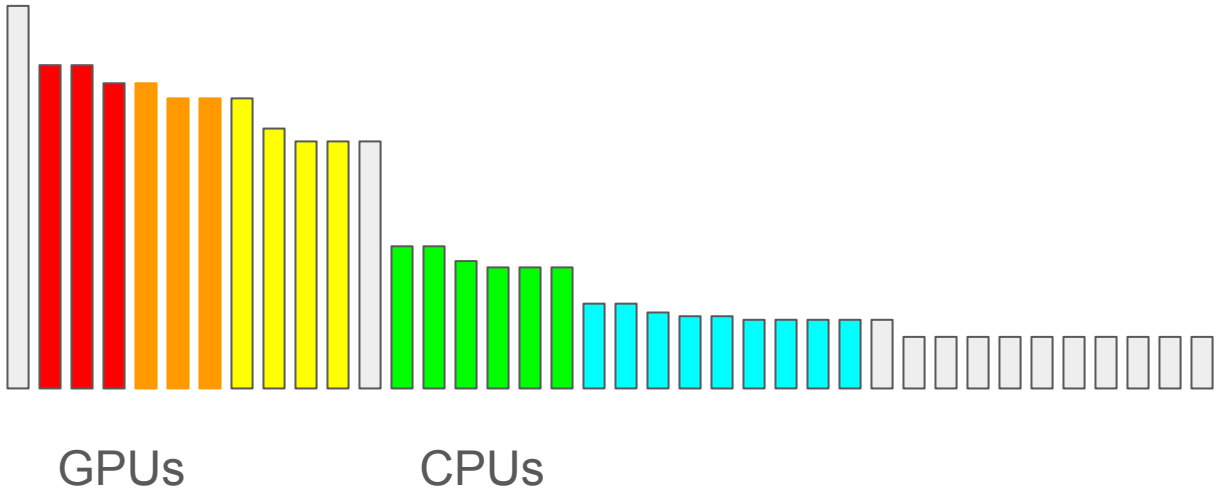
‘league’: a set of teams, all with about the same total speed



League formation

‘team’: a group of processors with about the same speed

‘league’: a set of teams, all with about the same total speed

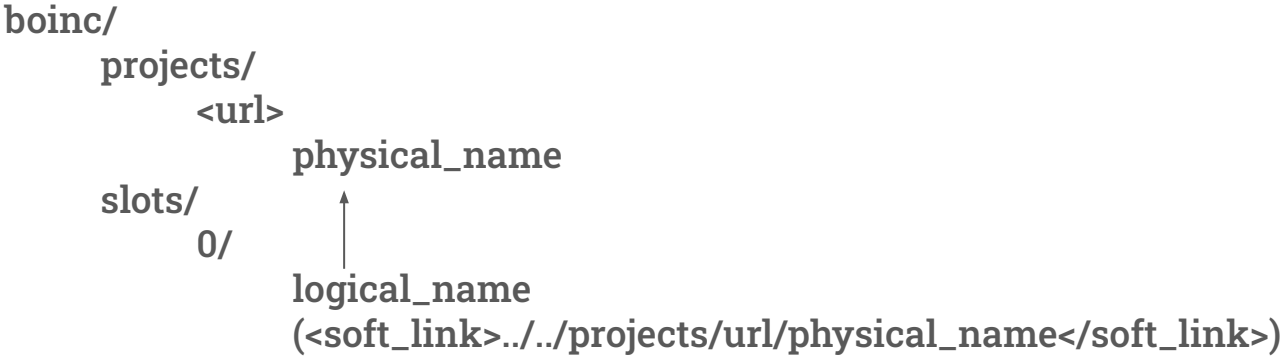


Cool new app test feature

- Lets you test an app version against the BOINC client without a project
- I used this for debugging the sporadic apps feature
- boinc –app_test

New vboxwrapper features

- Multi-attach images
  - multiple jobs can share same VM image file
- Sharing of slot and project directories
  - from within VM, can resolve links and directly access files
  - no copying of app, input/output files



Keeping up with technology

- PHP 8
- MySQL 8
- Python 3
- Mac OS

Support for Apple Silicon GPUs

- Client detects GPU via OpenCL and/or Metal
- Reports as ‘apple_gpu’ with API, version info
- Plan classes, web preferences

Documentation

- Existing
  - combines abstraction and implementation
  - landfill effect
- cookbooks
  - project creation
  - deploy VM app
  - job processing (Python)
  - validation (Python)
  - Needed: native apps; GPU apps; graphics apps
- videos

Future

- Run apps in Docker natively (no VBox)
  - Detect Docker in client, use a wrapper
  - Move boinc2docker into BOINC
  - Windows: conflict with VBox :-(
- Detect and schedule NPUs
- Client packaging
  - Windows Installer (Wix)
- Not done but still important
  - detect movie watching
  - limit fan noise
  - power/environmental preference features