Performance Analysis of Volunteer Computing Traces

Trilce Estrada and Michela Taufer — University of Delaware

Kevin Reed — IBM

Volunteer Computing (VC)

Volunteer Computing (VC) projects deploy resources connected to the Internet and owned by the public. VC projects are throughput-based: performance is measured in terms of number of valid results that are delivered to the scientists within a given interval of time.

BOINC projects are VC projects that:

- Execute longer or shorter work-units (WUs) with different degrees of sensitivity to errors
- Require different levels of redundancy

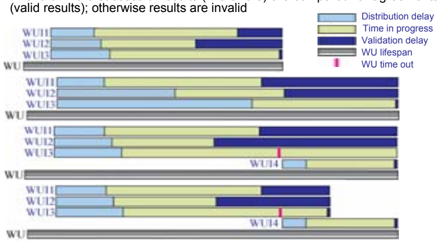
Different worker communities with different levels of availability and reliability can support a VC project.

This heterogeneity across VC projects makes it challenging to find performance bottlenecks as well as common scheduling strategies across projects that are able to tackle these bottlenecks.

Work-Unit Lifespan

The path that leads from the generation of work-units (WUs) to their result validation is error-prone and is characterized by delays hidden in the different phases of a WU's lifetime:

- Generation delay: from WU generation to generation of its replicas or work-unit instances (WUIs) → generated WUs reside in BOINC database
- Distribution delay: from WUI generation to their distribution to workers → new WUIs reside in BOINC DB and are moved to a shared memory for distribution when space is available
- Time in progress: from WUIs distribution to workers to the result collection → in-progress WUIs are distributed in packages and on the worker can be waiting for execution, in-execution, or waiting to be returned
- Validation delay: from the collection of results to the validation → two or more successful results (w/o errors) are compared for agreements (valid results); otherwise results are invalid



Time-out, error, and invalid results can result in new replications of WUIs and ultimately in delays in delivering results to the scientists.

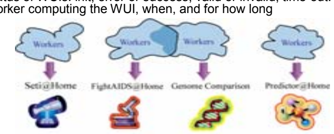
Challenges

The times WUIs wait for distribution, are in progress (waiting on the worker for execution, in execution, or waiting for being returned), or are waiting for validation are difficult to estimate because of the unpredictable behavior of volunteer resources. If project designers and administrators cannot estimate these times, VC projects can be affected by significant delay in delivering results to the scientists.

Volunteer Computing Traces

VC traces describe the overall behavior of VC workers, WU, and WUIs.
Trace data includes:

- When a WU is created,
- When its WUIs are created, distributed, returned, validated.
- Status of WUIs: init, error or success, valid or invalid, time-out.
- Worker computing the WUI, when, and for how long



VC traces are different from one project to another; different worker communities with different levels of availability and reliability can support a VC project.
Worker communities changes with the time and they can overlap partially with other VC projects.

Research Objective

Implement a statistical methodology that provides project designers and administrators with a quantitative approach to predict times in the lifespan of WUs.

Such predictions can be ultimately used for effective scheduling policies:

- Generate new WUs and WUIs
- Assign WUIs to workers
- Estimate delay before to consider a WUI timed out

Methodology

Statistical Distributions

Unknown density functions of WUI times:

- Distribution delay per number of WUIs
- Time in progress per number of WUIs
- Validation delay per number of WUIs

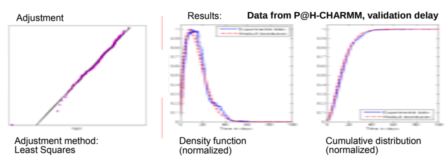
We propose to use statistical methods to predict these density functions and use these density functions to build cumulative distributions useful for scheduling decisions.
Our approach uses statistical methods based on quantiles to produce a $c\%$ confidence upper time bound b on the q^{th} quantile.
Upper time bound $[0, b]$:

- Maximum time a WUI is waiting for distribution (distribution delay)
- Maximum time a WUI is in progress
- Maximum time a WUI is waiting for validation (validation delay)

Adjust Traces with a Single Distribution

Distributions: Weibull, Levy, and Gamma
Adjustment algorithm:

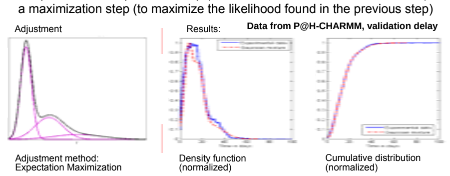
- Apply log(n) to the trace and transform the cumulative distribution function into a linear function
- Adjust the cumulative distribution using Least Squares: stop when sum of residuals is smaller than a given threshold



Adjust Traces with Mixture of Gaussians

Distribution: mixture of Gaussians
Adjustment algorithm:

- Obtain a probability mixture model by using the Expectation Maximization (EM) algorithm.
- EM performs an expectation step (to calculate the likelihood of the model) and a maximization step (to maximize the likelihood found in the previous step)



Training and Testing Windows

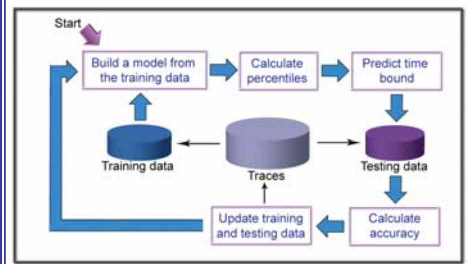
Build the probabilistic model using the data in the training window

Test the model using the data in the testing window

Update the model every 24 hours

Training window: fixed amount of WUIs
Testing window: variable amount of WUIs defined by a fixed interval of time

Overall Methodology

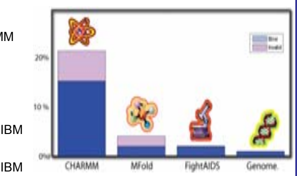


Results

Applications

VC Projects:

- Predictor@Home - CHARMM
  - TSRI
- Predictor@Home - MFold
  - TSRI
- FightAIDS@Home
  - World Community Grid, IBM
- Genome Comparison
  - World Community Grid, IBM

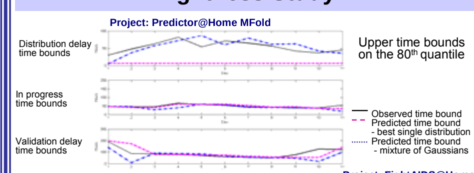


Traces	Number of workers	Number of days	Number of WU	WUI distributed	WUI returned	WU validated	Error rate	Time out rate	Invalid rate
CHARMM	5093	8	85722	199699	167699	43778	15 %	8 %	6 %
MFold	7810	15	162957	447451	440427	118447	4 %	12 %	2 %
FightAIDS	35583	30	136779	348246	344734	96918	2 %	1 %	0.1 %
Genome	30540	30	189182	478362	474459	142867	1 %	1 %	0.01 %

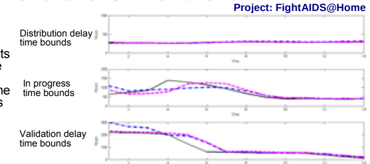
Metrics

Accuracy: percentage of correct time bounds (below the upper time bounds) predicted for all the WUIs. Overall accuracy does not capture a too generous setting of the time bound.
Tightness: fluctuation of the time bounds in testing traces versus predictions. Tightness captures the divergence between predicted and observed time bounds.

Tightness Study

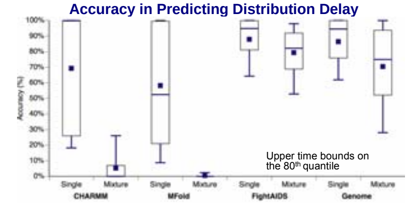


For stable projects such as Genome Comparison and FightAIDS@Home predicted bounds match observed bounds



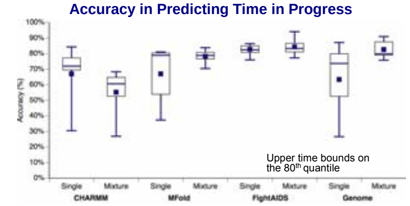
Accuracy Study

Accuracy in Predicting Distribution Delay



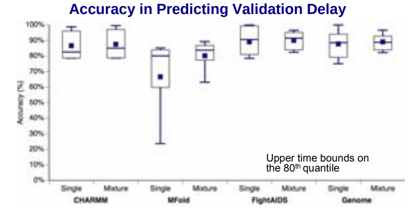
The error-pone nature of CHARMM and MFold makes their distribution delay unpredictable.

Accuracy in Predicting Time in Progress



Our method achieves good accuracy across different communities of VC workers.

Accuracy in Predicting Validation Delay



Our method is able to capture the validation delay across VC projects and communities

Future Work

- Tuning the training and testing intervals
  - Use heuristics and machine learning techniques
- Comparison with Binomial Method Batch Prediction (R. Wolski et al., 2006)
- Integration the method in SimBA
  - SimBA is our Simulator of BOINC Applications (M. Taufer et al., 2006)
- Tuning VC project parameters (e.g., number of WUs generated per hours, WUI time-out) using SimBA and the statistical methods
- Integration in real VC projects (Docking@Home)

Conclusion

- Statistical methods can provide scheduling policies of stable VC projects (with low error and invalid rates) with valuable insight on scheduling decisions
- For error-prone VC projects statistical methods can help to tune applications and scheduling policies a posteriori

Sponsors

NSF InnoDB CONACYT

Contacts: {estrada, taufer} @ udel.edu