Using MyGrid to Run Bag of Tasks Applications on Computational Grids

Walfredo Cirne

Universidade Federal de Campina Grande

Bag-of-Tasks Applications

- Parallel applications whose tasks are independent
  - Data mining
  - Massive search (as search for crypto keys)
  - Parameter sweeps
  - Monte Carlo simulations
  - Fractals (such as Mandelbrot)
  - Image manipulation (such as tomography)
  - And many others…

The Motivation for MyGrid

- Users of loosely-coupled applications could benefit from the Grid now
- However, they don´t run on the Grid today because the Grid Infrastructure is not widely deployed

- What if we build a solution that does not depend upon installed Grid infrastructure?

MyGrid Scope

- MyGrid allows a user to run Bag-of-Tasks parallel applications on whatever resources she has access to
- Bag-of-Tasks applications are those parallel applications formed by independent tasks
- One's grid is all resources one has access to
  - No grid infrastructure software is necessary
  - Grid infrastructure software can be used (whenever available)

What is MyGrid?

- MyGrid is a framework to build and run BoT applications on user-defined grids
- The user provides:
  - A description of her Grid
  - A way to do remote execution and file transfer
  - "The application"
- MyGrid provides:
  - Grid abstractions
  - Scheduling

MyGrid Concepts

- Home machine × Grid machine
  - Attributes make it possible to specify machine constraints
- Grid abstractions enable the user to deal with her grid without knowing details about each machine
  - file transfer
  - playpen
  - mirroring

MyGrid Concepts

- Job = set of independent tasks
- Task = init, grid, final subtasks
  - Subtasks are executed sequentially
  - Init subtask is executed on the home machine and sets up the environment for the task
  - Grid subtask is executed on the grid machine and performs the task computation
  - Final subtask is executed on the home machine, collects results and performs any clean up

Simple MyGrid Example

```
# initial
mg-services mirror $PROC tarefa
mg-services put $PROC ENTRADA.$TASK $PLAYPEN

# grid
tarefa < ENTRADA.$TASK > SAÍDA

# final
mg-services get $PROC $PLAYPEN/SAÍDA
    resultados/SAÍDA.$TASK
```

Defining My Personal Grid

```
proc:
   name = ostra.lsd.ufcg.edu.br
   attributes = lsd, linux
   type = user_agent

proc:
   name = memba.ucsd.edu
   attributes = lsd, solaris
   type = grid_script
   rem_exec = ssh %machine%command
   copy_to = scp %localdir/%file %machine:%remotedir
   copy_from = scp %machine:%remotedir/%file %localdir

[...]
```

Factoring with MyGrid

- Fatora n generates files tasks, init, gridi, and collect, and then invokes mg-addtask tasks
- tasks

```
task:
init= init
grid= grid1
final= collect
task:
init= init
grid= grid2
…
```

Factoring with MyGrid

- init

    ```
    mg-services put $PROC ./Fat.class $PLAYPEN
    ```

- grid1

    ```
    java Fat 3 18655 34789789798 output-$TASK
    ```

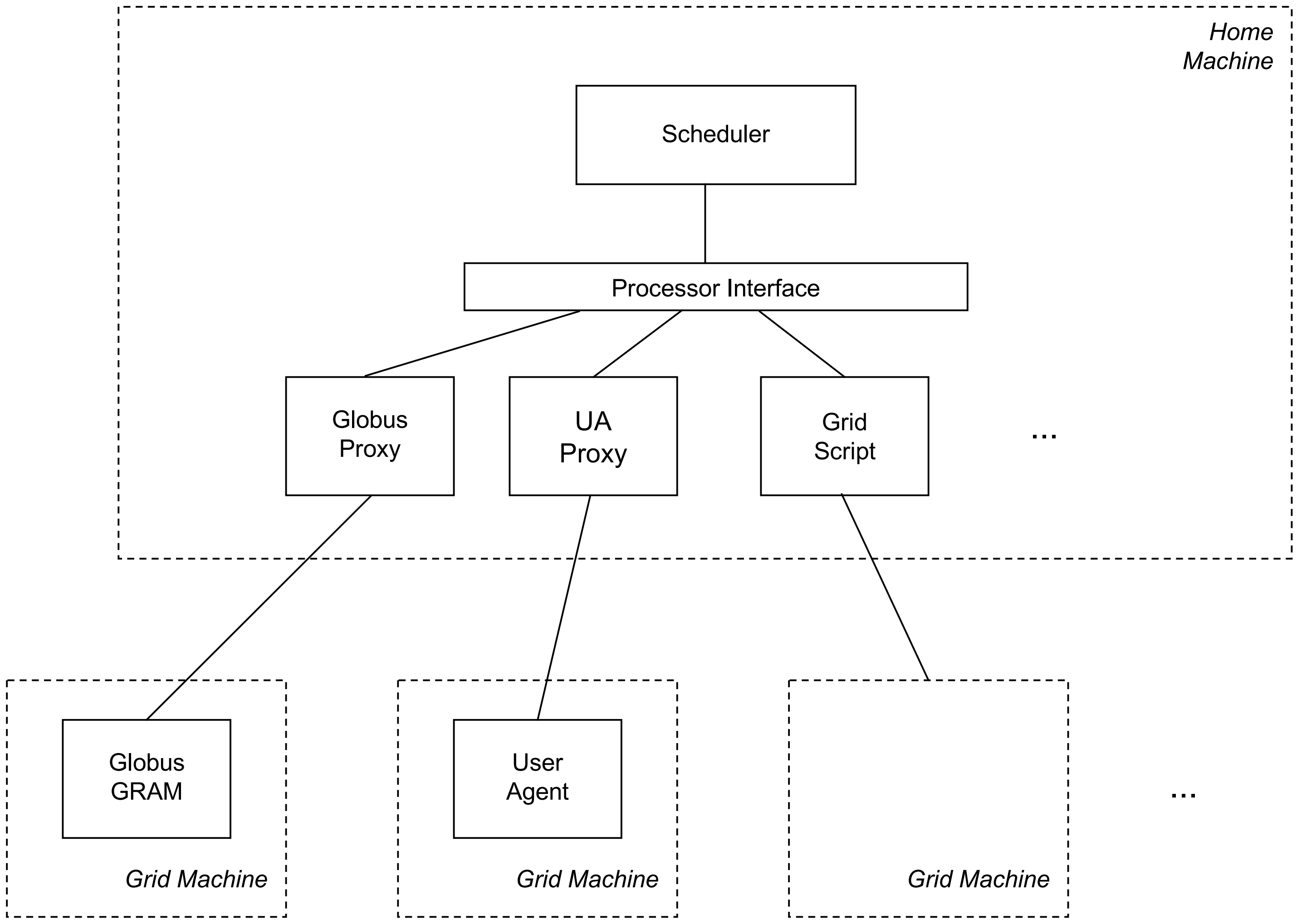
- grid2

    ```
    java Fat 18655 37307 34789789798 output-$TASK
    ```

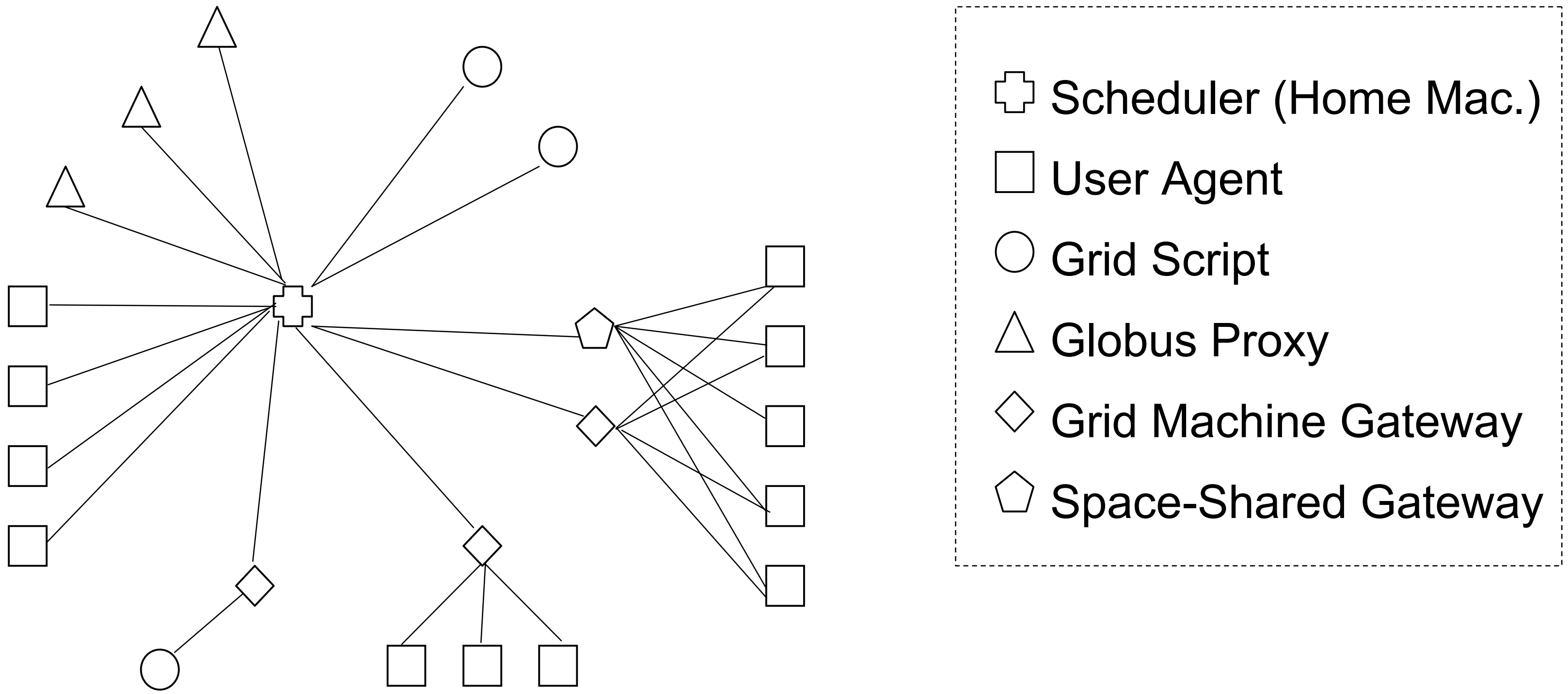
- collect

    ```
    mg-services get $PROC $PLAYPEN/output-$TASK results
    ```

Making MyGrid Encompassing



Dealing with Firewalls, Private IPs, and Space-Shared Machines



The Scheduling Challenge

- Grid scheduling typically depends on information about the grid (e.g. machine speed and load) and the application (e.g. task size)
- However, getting grid information makes it harder to build an encompassing system
  - The Grid Machine Interface would have to be richer, and thus harder to implement
- Moreover, getting application information makes the system harder to use and less simple
  - The user would have to provide task size estimates

Scheduling With No Information

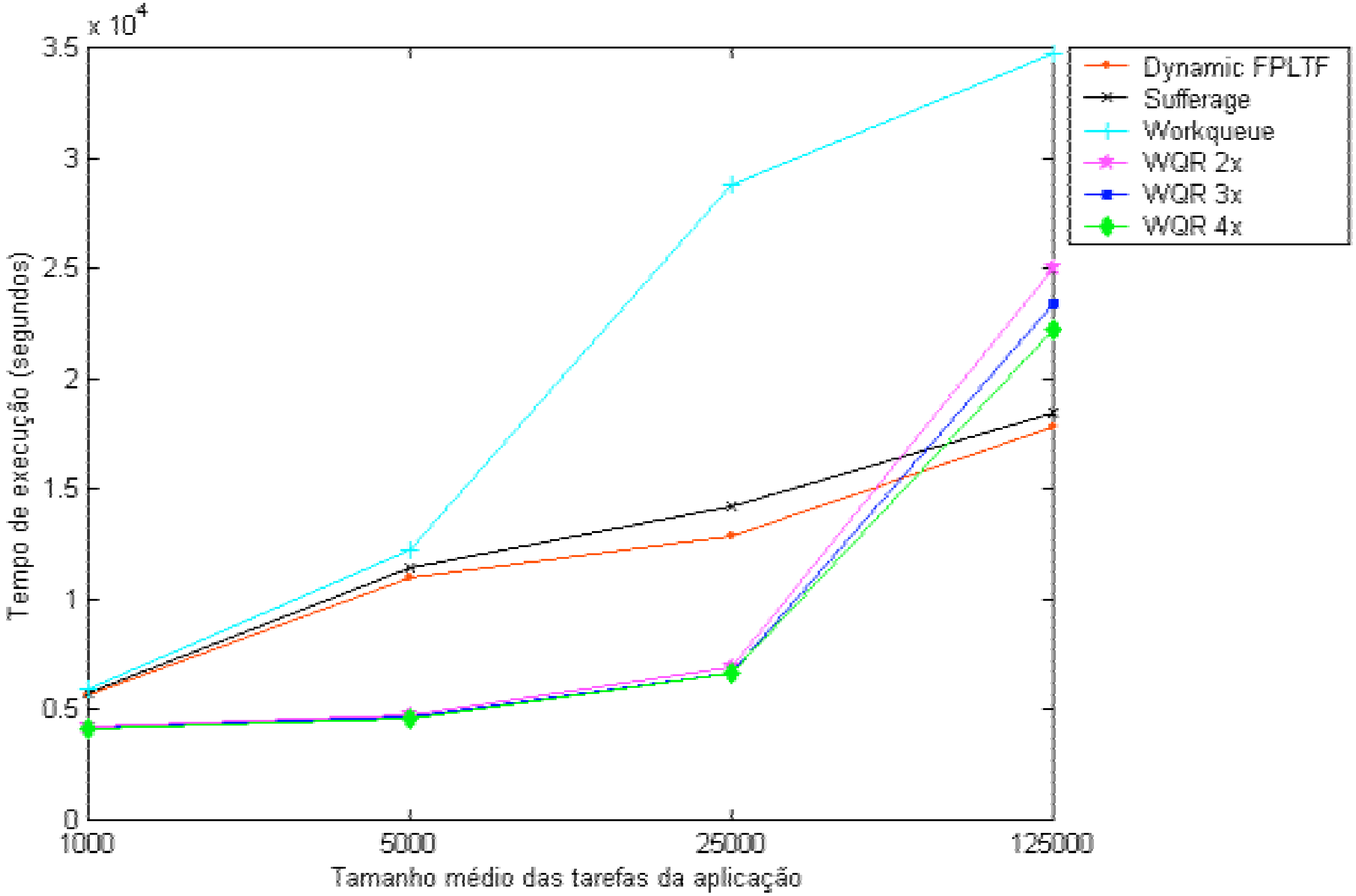
- Work-queue with Replication
  - Tasks are sent to idle processors
  - When there are no more tasks, running tasks are replicated on idle processors
  - The first replica to finish is the official execution
  - Other replicas are cancelled
  - Replication may have a limit
- The key is to avoid having the job waiting for a task that runs in a slow/loaded machine

Work-queue with Replication

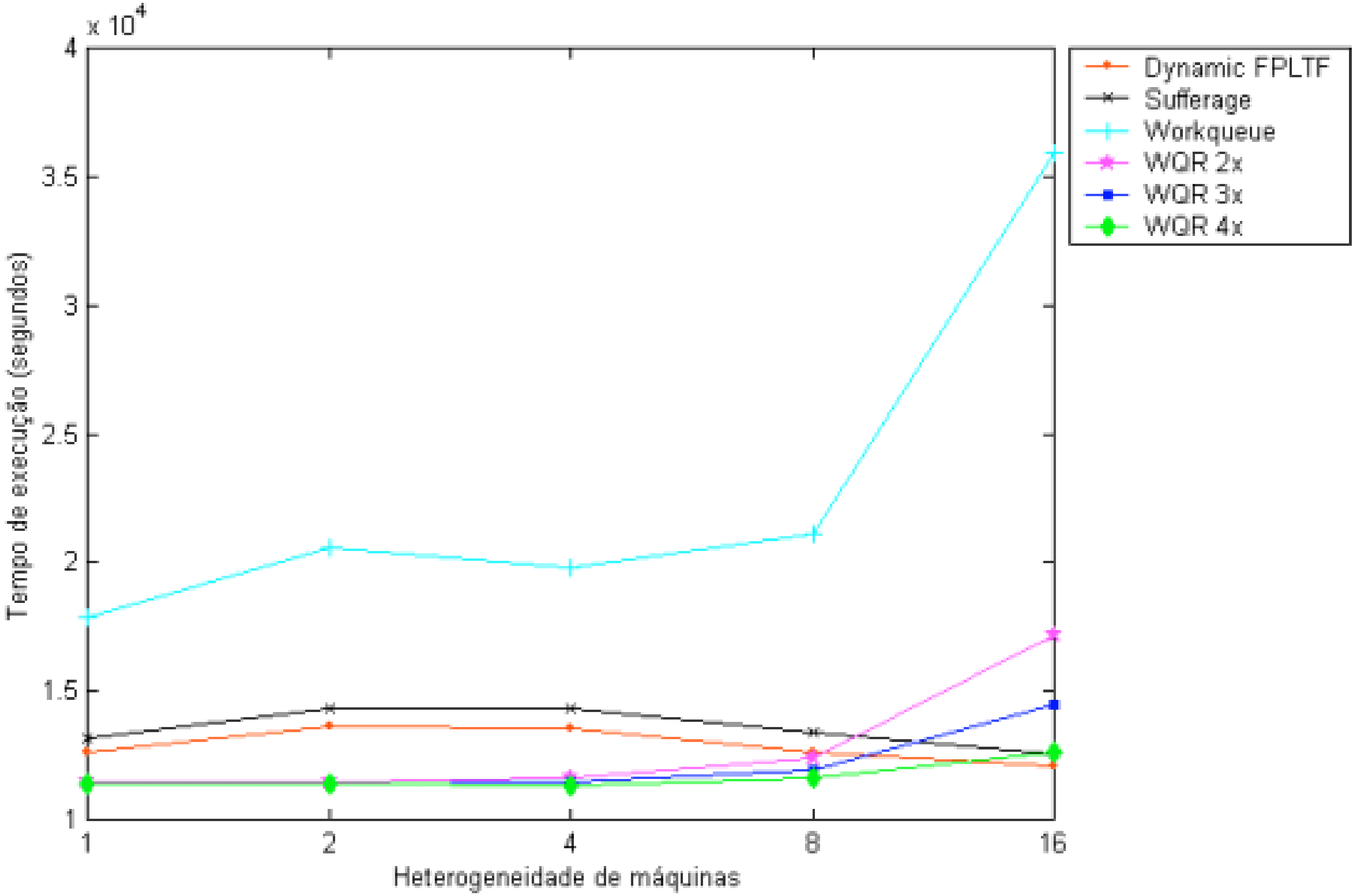
- 8000 experiments
- Experiments varied in
  - grid heterogeneity
  - application hetetogeneity
  - application granularity
- Performance summary:

	Sufferage	DFPLTF	Workqueue	WQR 2x	WQR 3x	WQR 4x
Average	13530.26	12901.78	23066.99	12835.70	12123.66	11652.80
Std. Dev.	9556.55	9714.08	32655.85	10739.50	9434.70	8603.06

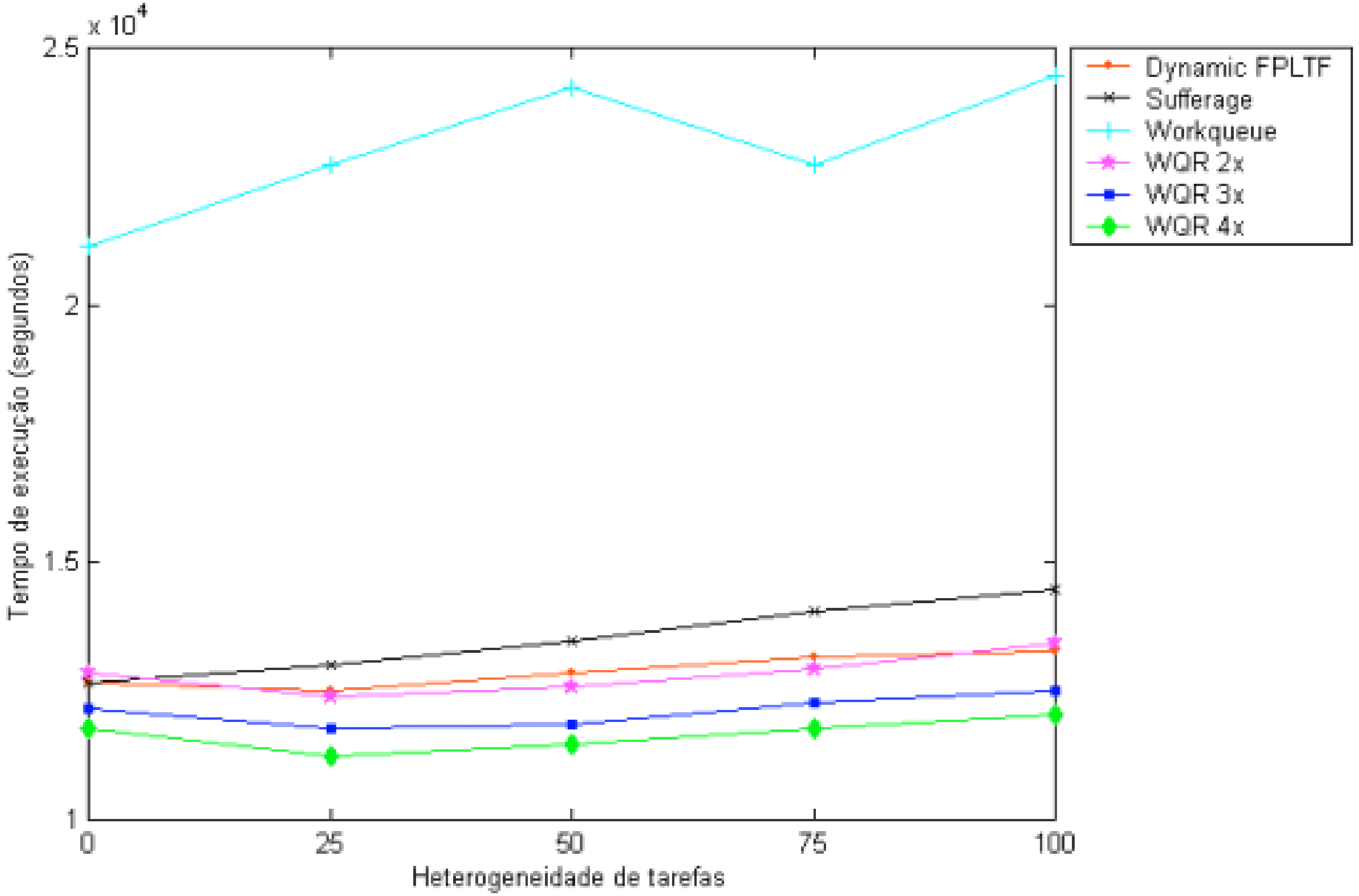
Application Granularity



Grid Heterogeneity



Application Heterogeneity

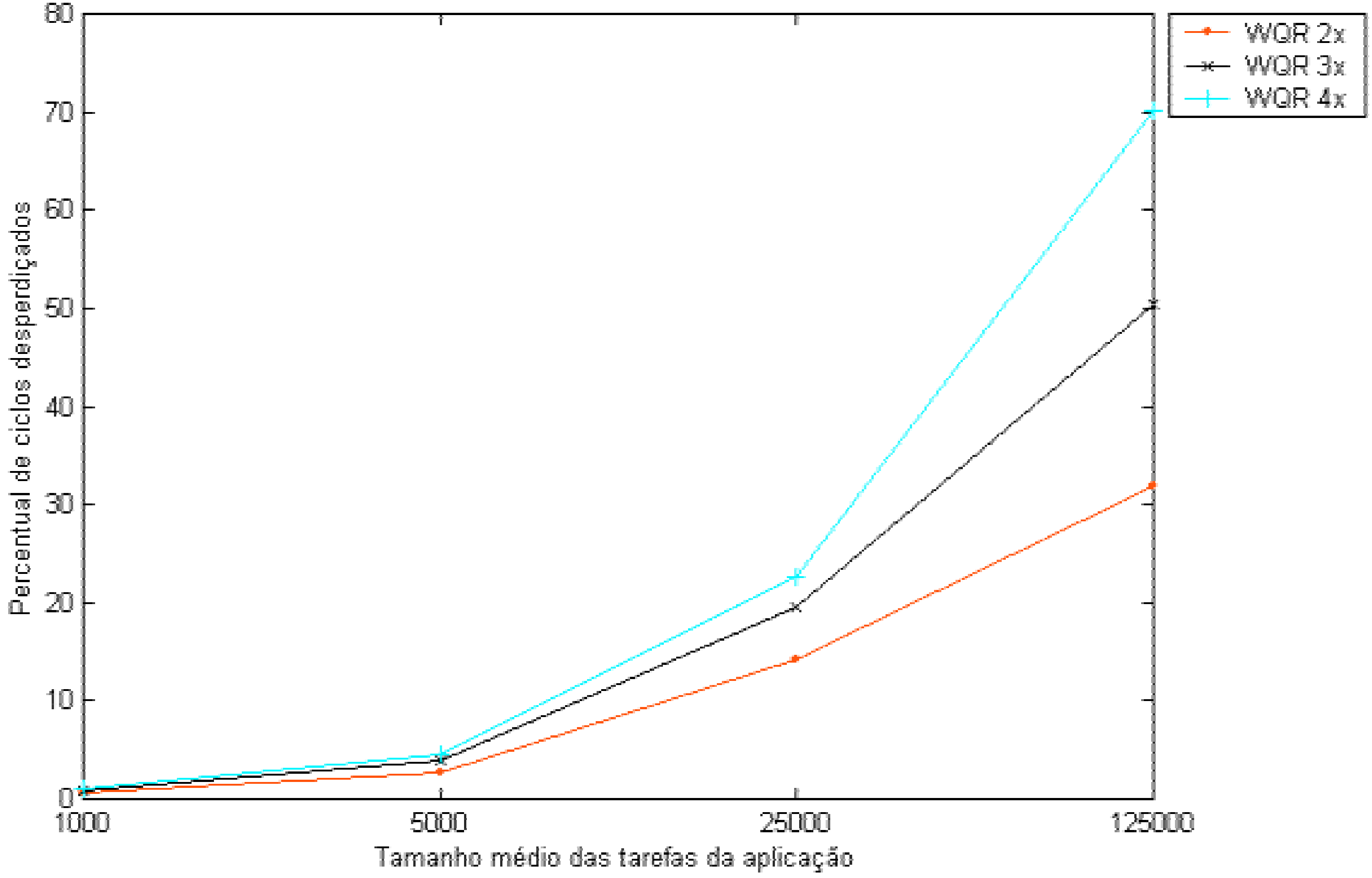


WQR Overhead

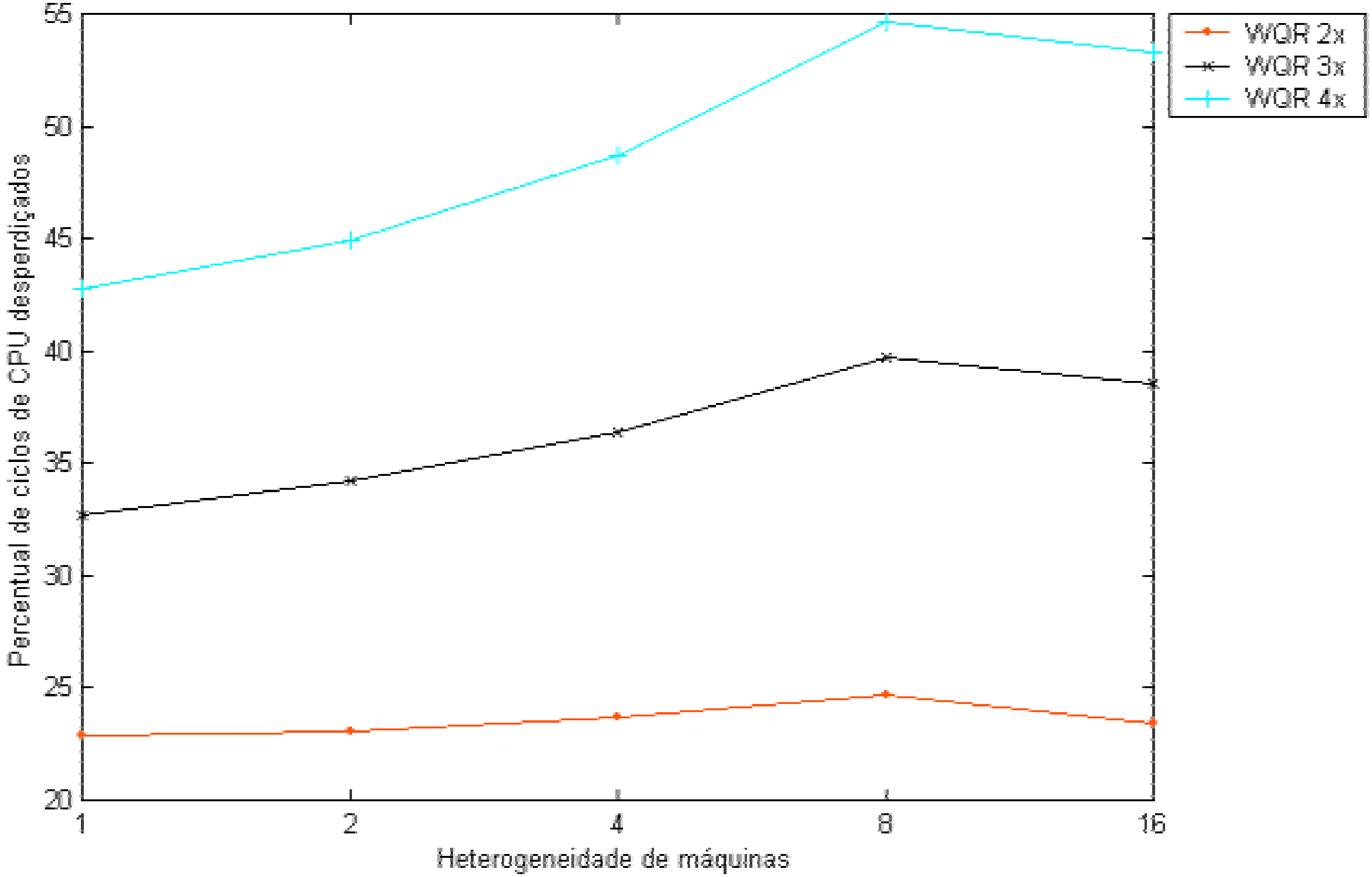
- Obviously, the drawback in WQR is cycles wasted by the cancelled replicas
- Wasted cycles:

	WQR 2x	**WQR 3x**	**WQR 4x**
Average	23.55%	36.32%	48.87%
Std. Dev.	22.29%	34.79%	48.93%

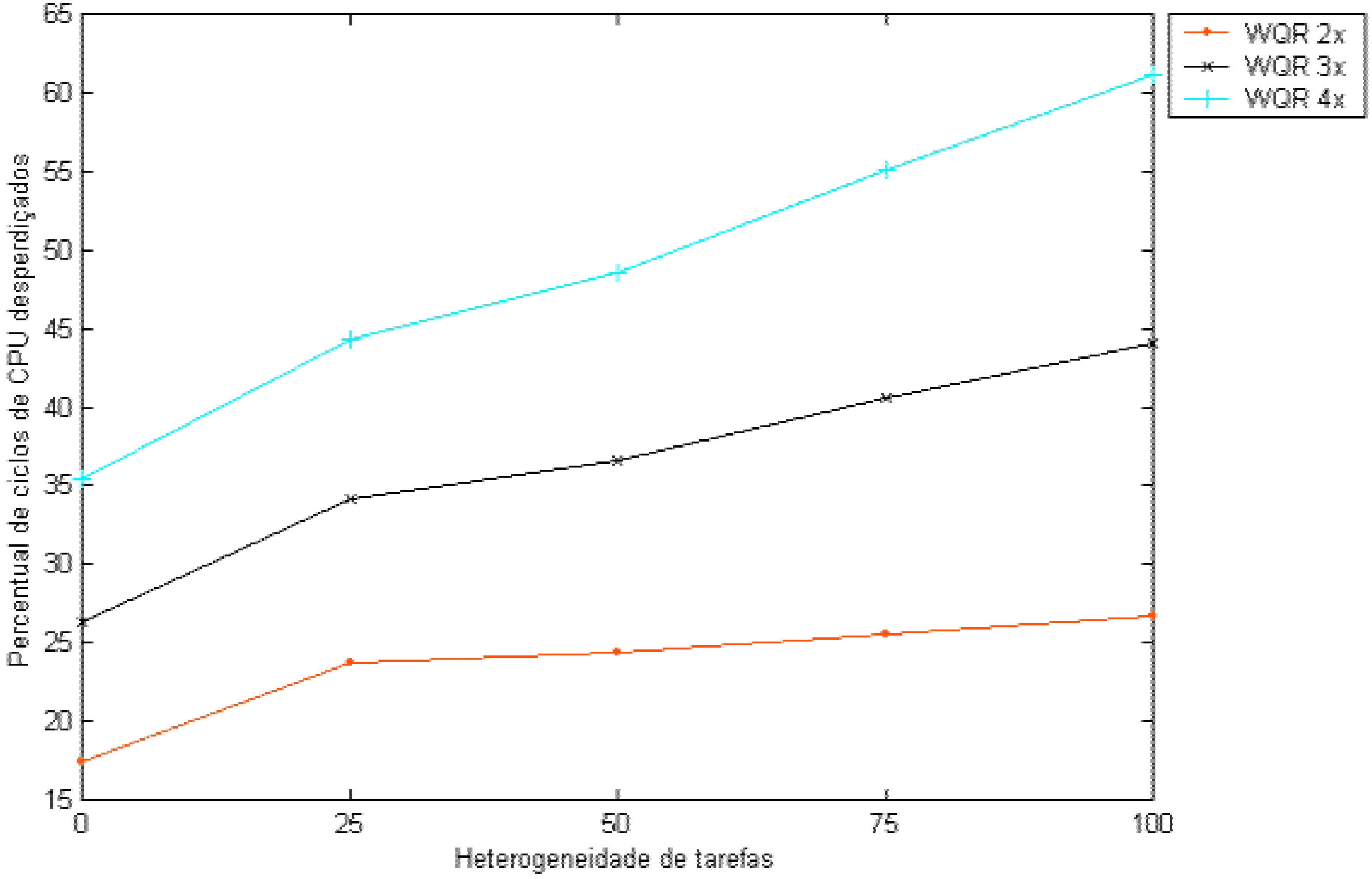
Application Granularity



Grid Heterogeneity



Application Heterogeneity



Self-installation

- Grid Scripts are a source of headaches
  - The interface of human-oriented commands makes scripts hard to port and easy to break with new version
- As an attempt to minimize this problem, MyGrid’s self-installation uses Grid Scripts only to copy and remotely start MyGrid’s own User Agent

Proof of Concept

- During a 40-day period, we ran 600,000 simulations using 178 processors located in 6 different administrative domains widely spread in the USA
- We only had GridScript and WorkQueue
- MyGrid took 16.7 days to run the simulations
- My desktop machine would have taken 5.3 years to do so
- Speed-up is 115.8 for 178 processors

Fighting AIDS

- 55 machines in 6 administrative domains in the US and Brazil
  - The machines were accessed via User Agent, UA + Grid Machine Gateway, UA + ssh tunnel, and Grid Scripts
- Task = 3.3 MB input, 200 KB output, 4 to 33 minutes of dedicated execution
- Ran 60 tasks in 38 minutes
- Speed-up is 29.2 for 55 machines
  - Considering an 18.5-minute average machine

Conclusions

- Bag-of-tasks parallel applications can currently benefit from the Grid
- Running grid applications at the user-level is a viable strategy
  - However, firewalls, private IPs and the such make it much harder than we initially thought
  - Is “upperware” the way to go for new middleware development?

Future Work

- Make MyGrid OGSA-complaint
- Create OurGrid, a community grid for resource sharing
  - Deploy OurGrid in RNP2
- Extend the scheduler for data intensive applications
  - Such a scheduler should try to minimize data movement