Workflow Management in a Protein Clustering Application

J.L. Vázquez-Poletti

E. Huedo

R.S. Montero

I.M. Llorente

Universidad Complutense de Madrid

Spain

Overview

CCGrid 07

- The Workflow Management Concept
- Available Workflow Management Systems
- GridWay's Workflow Management
- The Bioinformatics Application
- Implementation
- Experimental Results
- Conclusions and Future Work

The Workflow Management Concept

CCGrid 07

- Grid Computing is extending and consolidating
- Growing complexity of problems: Workflows
- Workflows: Nodes + Dependencies
  - Node = Job
- Workflow Management System: Defines, manages and executes workflows
  - “Ordem e Progresso” :-)

Available Workflow Management Systems

CCGrid 07

- **DAGMan (Condor)**
  - Rescue DAG Generation
  - Definable number of job retries
- **Pegasus (extends DAGMan)**
  - A.I. planning techniques
  - Access to various Grid information services (MDS, RLS, MCS, ...)
  - Resource selection randomly and performance prediction infrastructure
  - Pluggable task scheduling strategies
  - Just in-time scheduling
- **Triana**
  - Code can run locally or distributedly
  - Parallel or P2P policy
  - Fault Tolerance Mechanisms based on GAT from GridLab

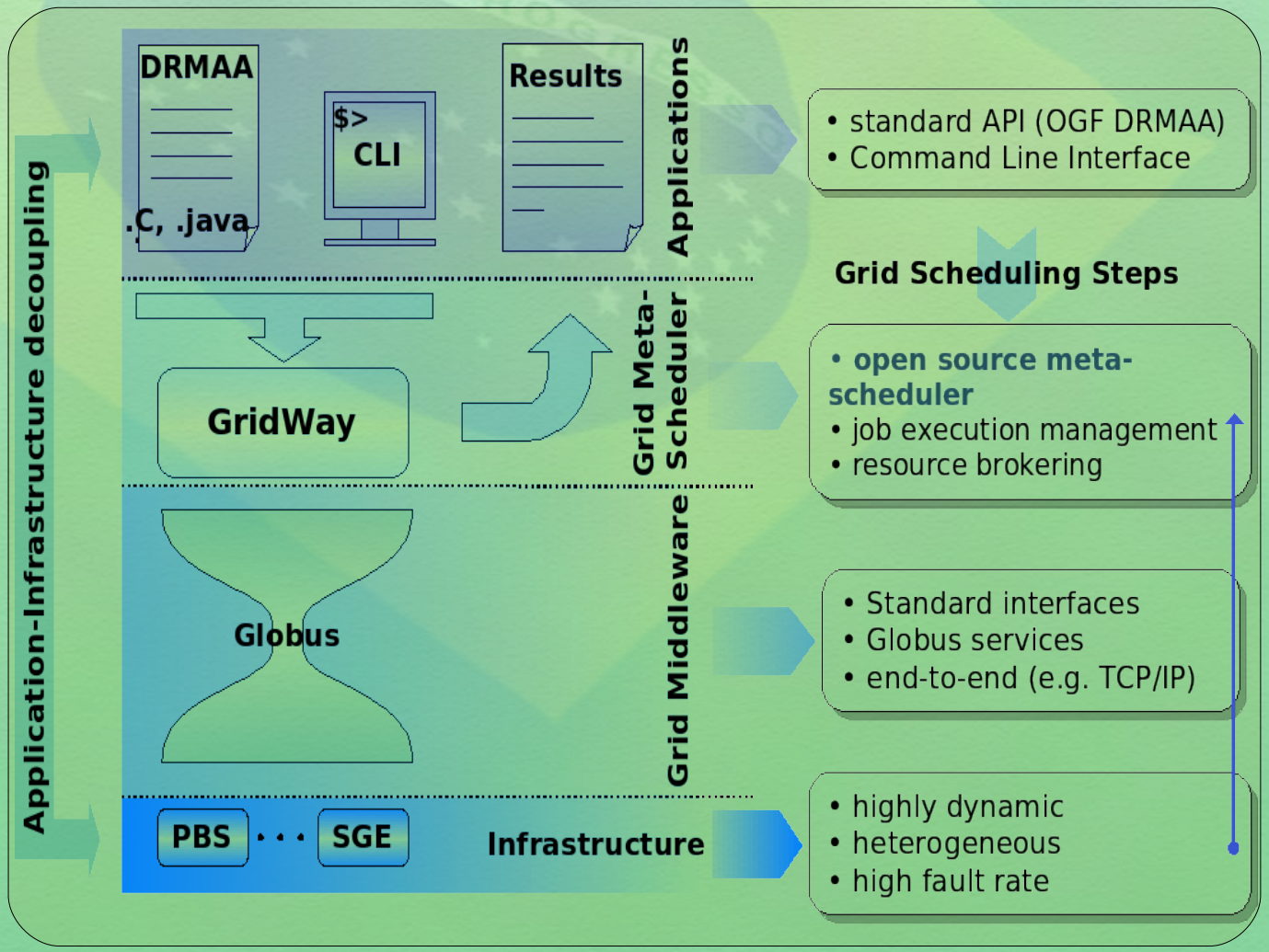
Available Workflow Management Systems

CCGrid 07

- **ICENI**
  - Several scheduling algorithms provided
  - New algorithms can be plugged
  - Historical data used in scheduling
  - Lazy and advanced reservation using WS-Agreement
- **GridAnt**
  - Extends Ant deploy tool
  - Information retrieval through Globus MDS
  - User defines fault tolerance mechanisms
- **GridBus**
  - Resource information from Grid Market Directory
  - Grid accounting and billing through Grid Bank
  - Reschedules to alternative resources
  - Users define service constraints

GridWay's Workflow Management

CCGrid 07



GridWay's Workload Management

CCGrid 07

- Handles DAG based workflows
- Advanced flow structures (loops, branches)
  - Distributed Resource Management Application (DRMAA) API
- Workflow specified without referring to specific resources
- Uses static and dynamic information about resources
- Scheduling decisions taken at run-time (task-level)
  - Requirements + ranking
  - Made by central instance
- Considers historical information about task execution
- Phases: Prolog (input files staging), Wrapper (execution) and Epilog (output files staging)
- Fault tolerance at task-level
  - Retries execution/transfer on same resource
  - Submission to alternate resource
  - Checkpoint/restart (migration)

The Bioinformatics Application

CCGrid 07

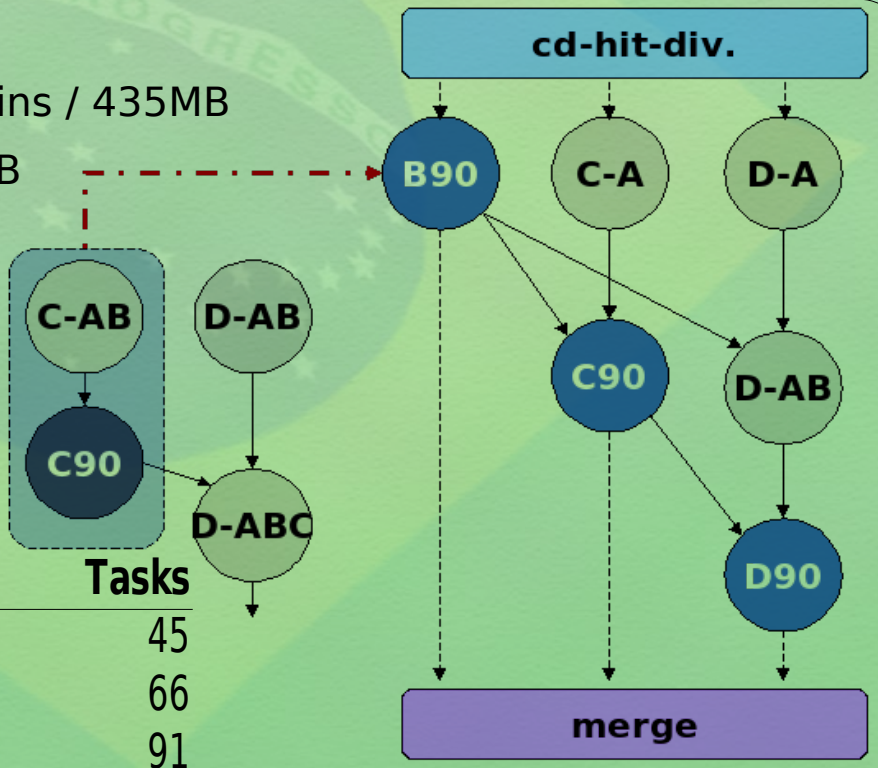
cnio

- CD-HIT: Protein clustering
  - Compares protein DB entries
  - Eliminates redundancies
- Example: Used in UniProt for generating UniRef data sets
- Our case: Widely used in the Spanish National Oncology Research Center (CNIO)
- Infeasible to be executed on single machine
  - Memory requirements
  - Total execution time

Implementation

CCGrid 07

- Input DB:
  - 504,876 proteins / 435MB
- Executables: 1.1MB



DB Div.	Mean Size	Tasks
10	44MB	45
12	36.5MB	66
14	31.5MB	91
16	27.5MB	120
18	24.5MB	153
20	22MB	190
22	20MB	231

Implementation

CCGrid 07

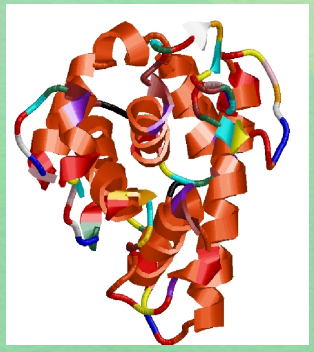
egee
Enabling Grids for E-sciencE

- Launch site: Universidad Complutense de Madrid
- Launch period: July 2006 (different times/days/weeks)

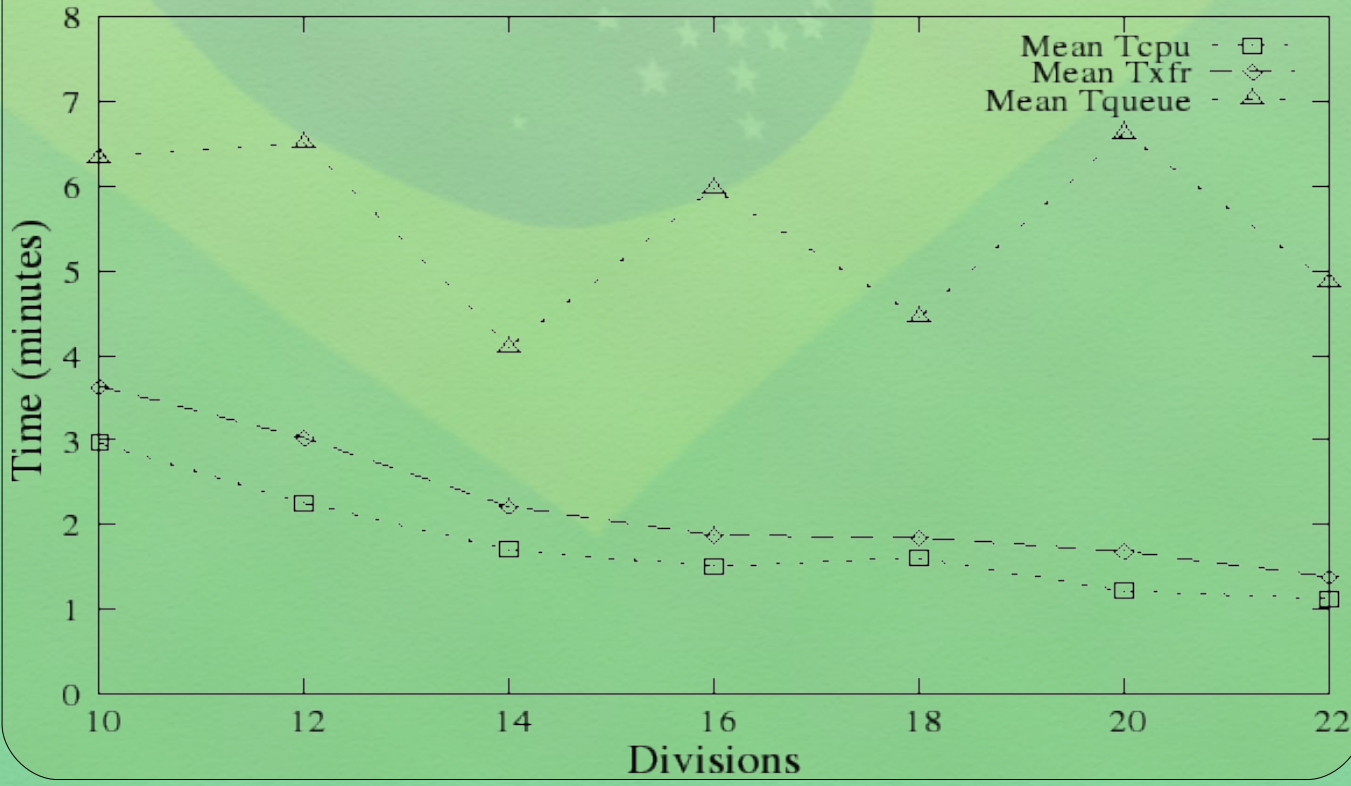
Site	Processor	Nodes	Speed
BIFI	Intel P.IV	56	3.2GHz
CESGA	Intel P.III	16	500MHz
CGG	Intel P.III	58	1.2GHz
CIEMAT	Intel Xeon	226	3.2GHz
GRIF	Intel P.IV	14	2.8GHz
JINR	Intel P.D	30	2.8GHz
L.-HEP	Intel P.IV	374	3GHz
PNPI	Intel P.IV	60	3GHz
RAL	Intel P.IV	62	2.8GHz
RALPP	Intel P.III	1064	1GHz
ScotGRID	Intel Xeon	6	2.8GHz
SINP	Intel Xeon	94	2.8GHz

Experimental Results

CCGrid 07

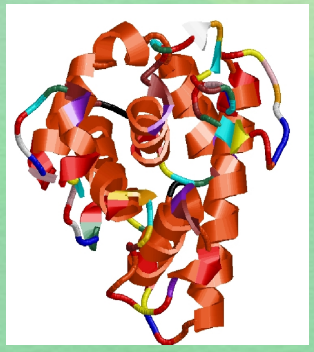


- T_{cpu}: average CPU time
- T_{xfr}: average transfer time
- T_{que}: queuing time



Experimental Results

CCGrid 07

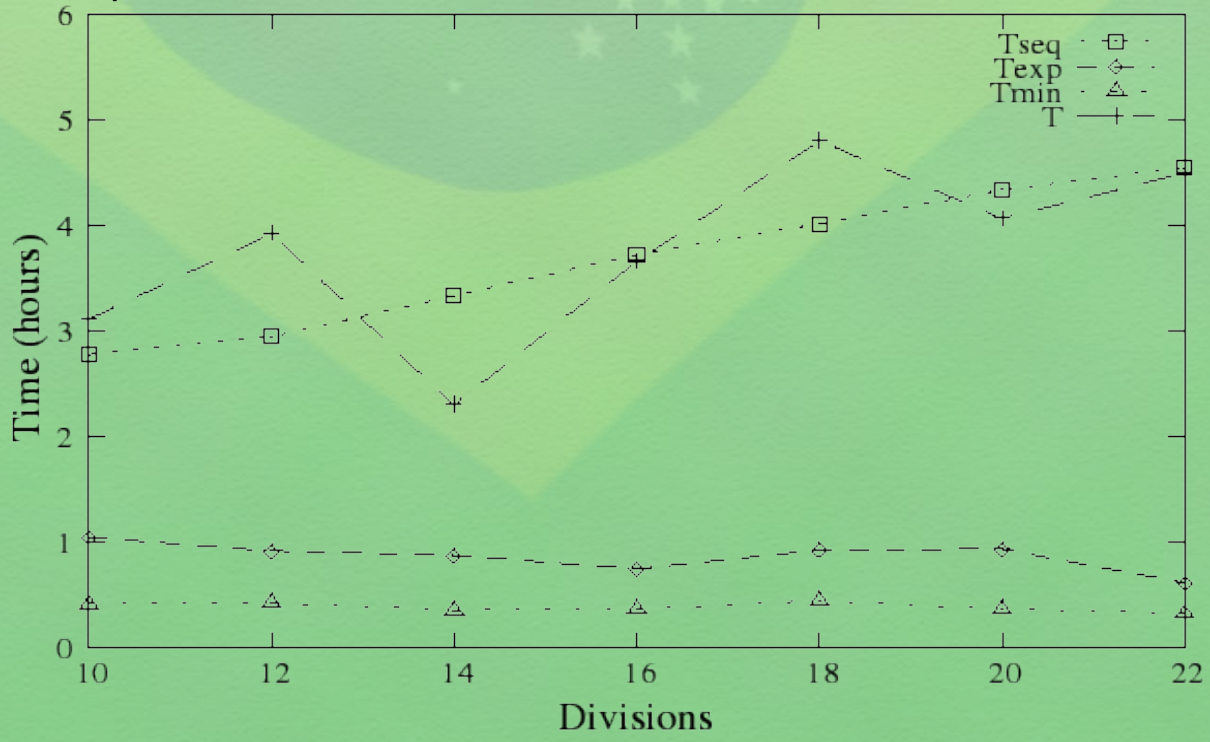


- $T_{exp} = N \cdot (T^A_{cpu} + T^A_{xfr})$: Expected walltime
- Lower bound estimation of walltime
- Sequential execution time
- T: Experimental time

$$T_{exp} = N \cdot (T^A_{cpu} + T^A_{xfr})$$

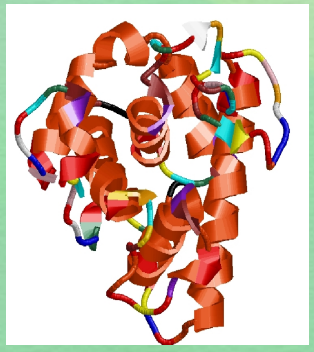
$$T_{min} = N \cdot T^A_{cpu}$$

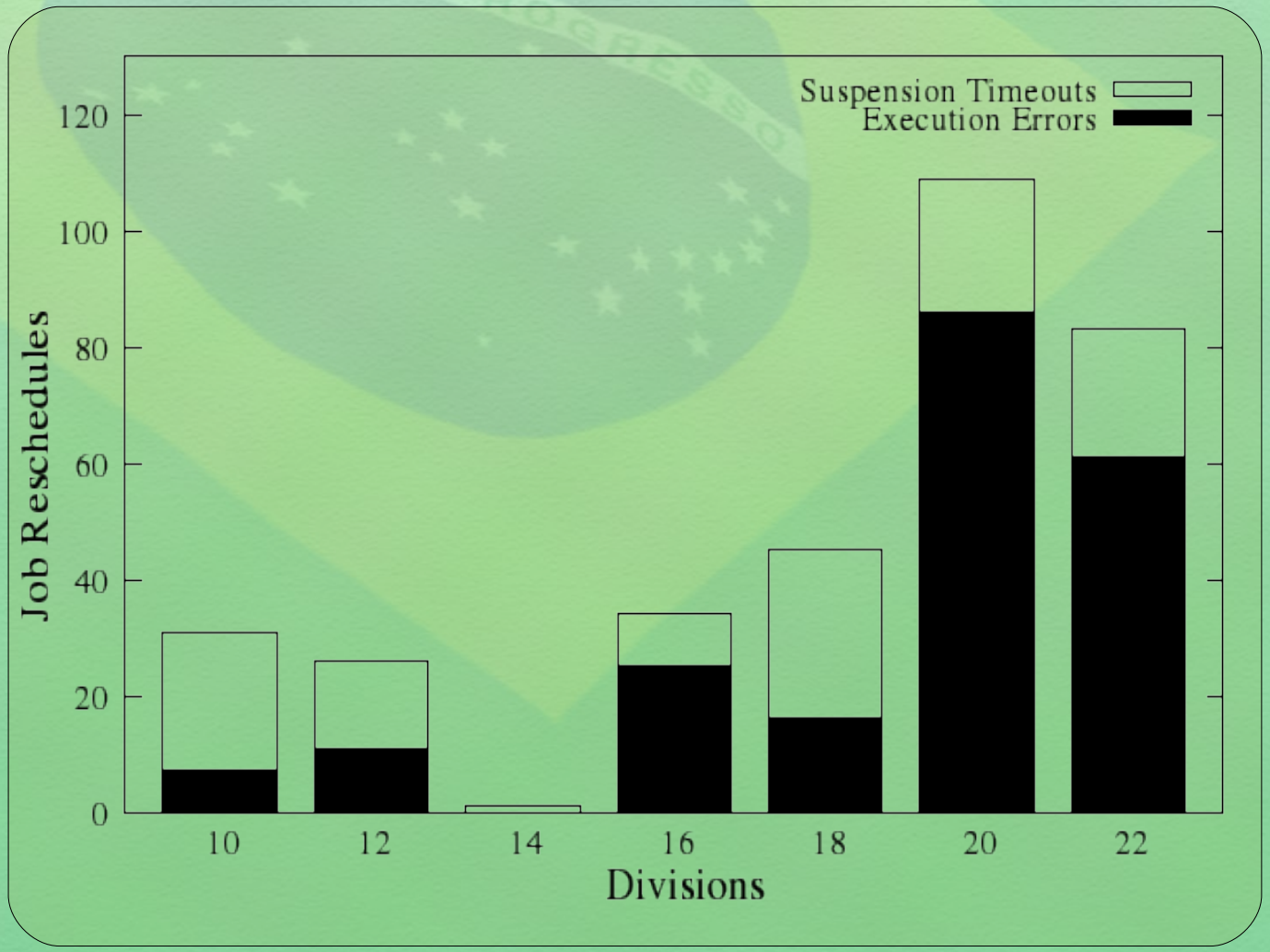
$$T_{seq} = N \cdot T^A_{cpu} + T^B_{xfr} \cdot \sum_{n=2}^{N} (n-1)$$



Experimental Results

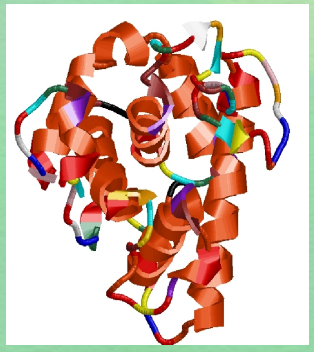
CCGrid 07



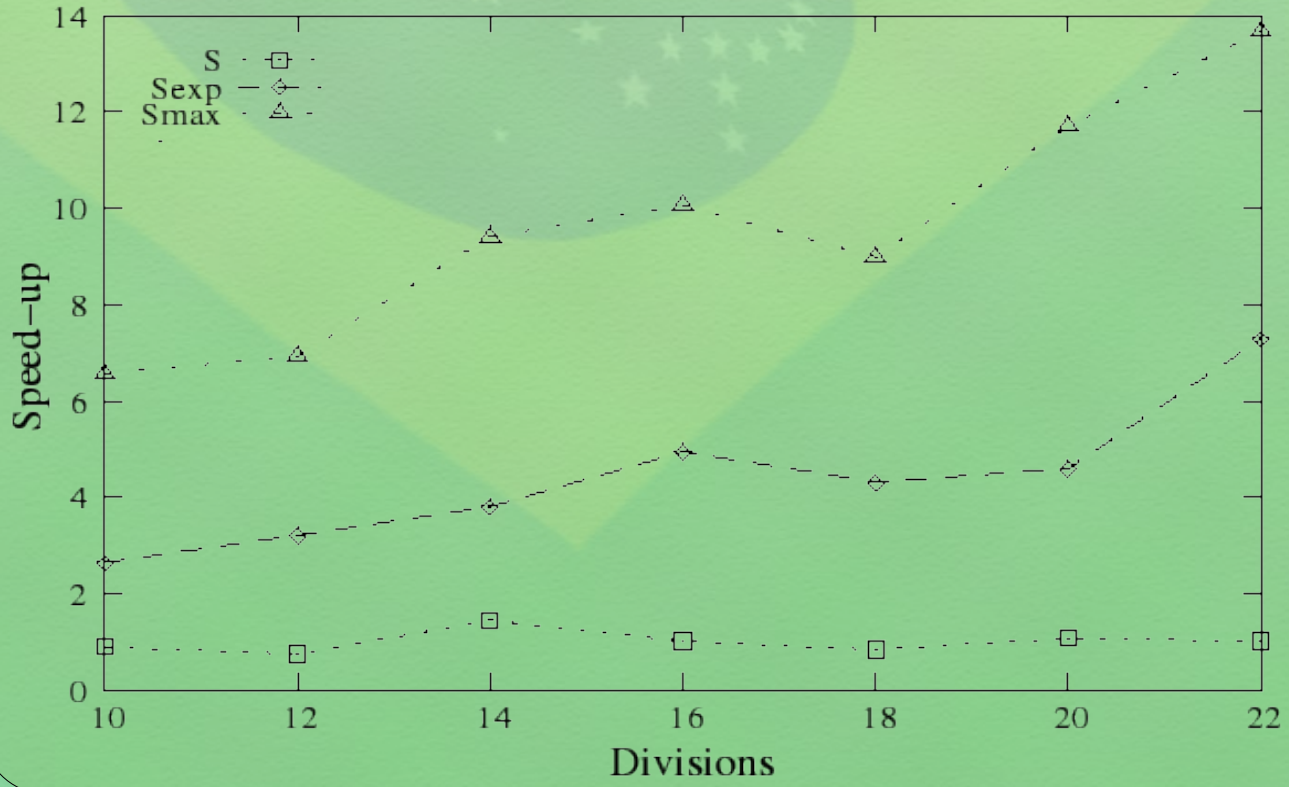


Experimental Results

CCGrid 07



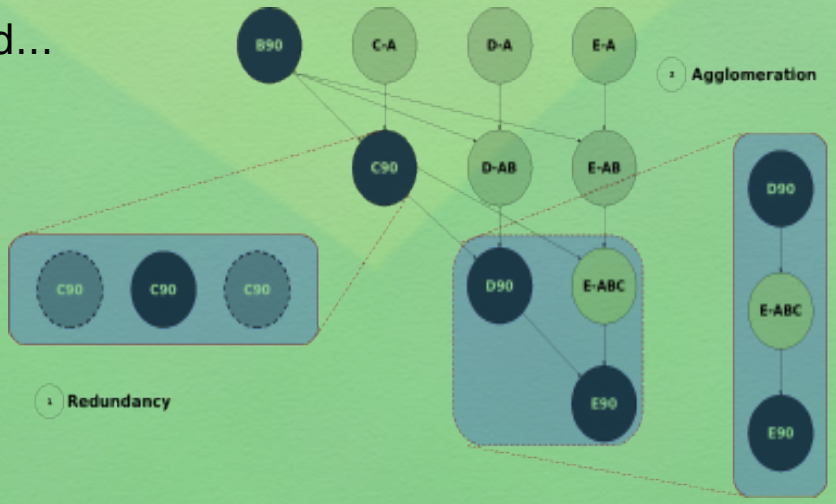
- S: Speed-up
- S_{exp}: without queue times or job failures
- S_{max}: upper bound limit using T_{min}



Conclusions and Future Work

CCGrid 07

- "Gridification" of a Bioinformatics application was analyzed
  - Large DB cannot be processed in a single machine
- Efficiency expected dramatically limited
  - Nature of the Grid: Dynamism + Heterogeneity + High fault rate
- GridWay is robust and reliable
- The Future:
  - Production DB (~1.7GB – 4,186,284 Proteins)
  - And…



Thank you! - Muito Obrigado!

CCGrid 07