Inter-Task Communication on Volatile Nodes

Jaspal Subhlok

University of Houston

Intergalactic Workshop

Big Picture -- VOLPEX: Parallel Execution on Volatile Nodes

Communicating Parallel Programs

ON

Ordinary Desktop Volatile Nodes

Key problem : High failure rates AND coordinated execution

Collaborators

Edgar Gabriel , Rong Zheng (UH Faculty)

Nagarjan Kanna, Troy LeBlanc, Girish N. (UH Students)

David Anderson

CS@UH

Major Challenges in VOLPEX

Failure Management
- Replication and/or Checkpointing

Programming/Communication Model
- Asynchronous PUT/GET API (Like LINDA)
- Message Passing

Execution management
- Selection of “good” nodes for execution

Integration with BOINC

Test case, examples, applications
- Real world value? Need help!

Failure Management

Replication:

- Concurrent replicas of each processes.
- Application at the speed of the fastest replicas
- Application fails only if all replicas fail

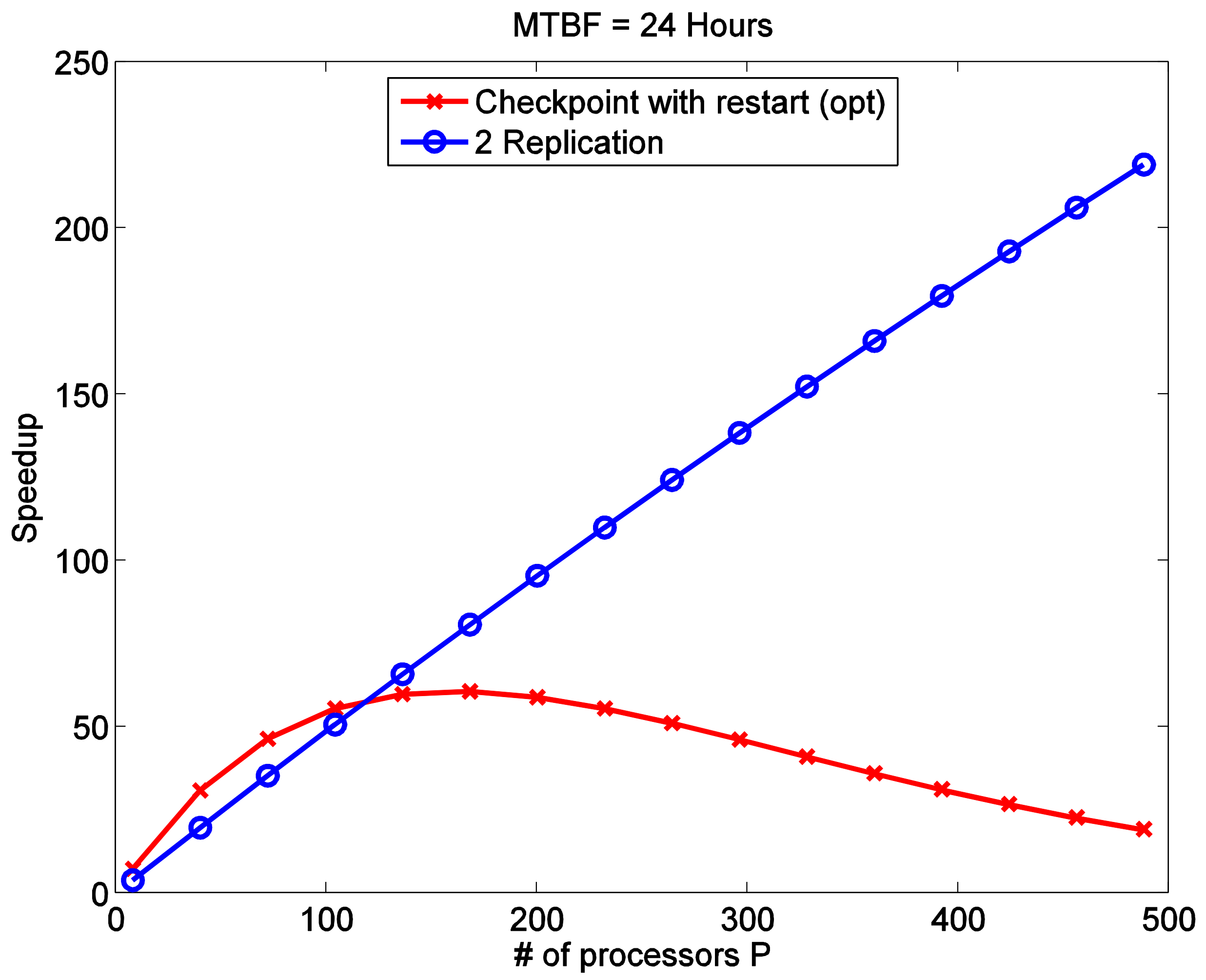
Checkpointing:

- Independent checkpoints
- Recovery from process checkpoint and communication logs
- All processes wait during recovery

Hybrid: Checkpoint-restart to maintain degree of replication

CS@UH

Checkpointing versus Replication



CS@UH

PUT/GET : “Dataspace” API
(M.S. Thesis of Nagarajan Kanna)

Asynchronous, Independent, One way, PUT/GET transactions with an abstract dataspace (~Linda)

PUT (tag, data) place **data** in dataspace indexed with **tag**

READ (tag, data) return **data** matching the **tag**.

GET (tag, data) return and remove **data** matching **tag**.

A Powerful API

- **Message passing can be implemented on this API**
- **And more, global variables, producer-consumer, etc.**

Implementation of Dataspace API

LINDA API has been implemented many many times!

Consistency in face of fault management is a major challenge.

- **Replication and checkpoint-restart imply that a logical PUT/GET may be executed many times physically.**

Consistency demands:

Additional PUTs must be ignored

All READ/GETs corresponding to the same logical call must return the same data

SOLUTION APPROACH : Data returned for PUT/GETs is logged. Replica calls processed from data logs

Dataspace API : Status and plans

Implementation of basic API is nearly complete! What still needs to be done:

- Testing, Validation
- Integration with BOINC
- Application development (replica exchange)

Implementation based on a dedicated datapace server

- Multiple distributed dataspace servers possible
- API may not be ideal for direct client to client communication

Volpex MPI

(Ph.D. work of Troy LeBlanc, with Prof Gabriel)

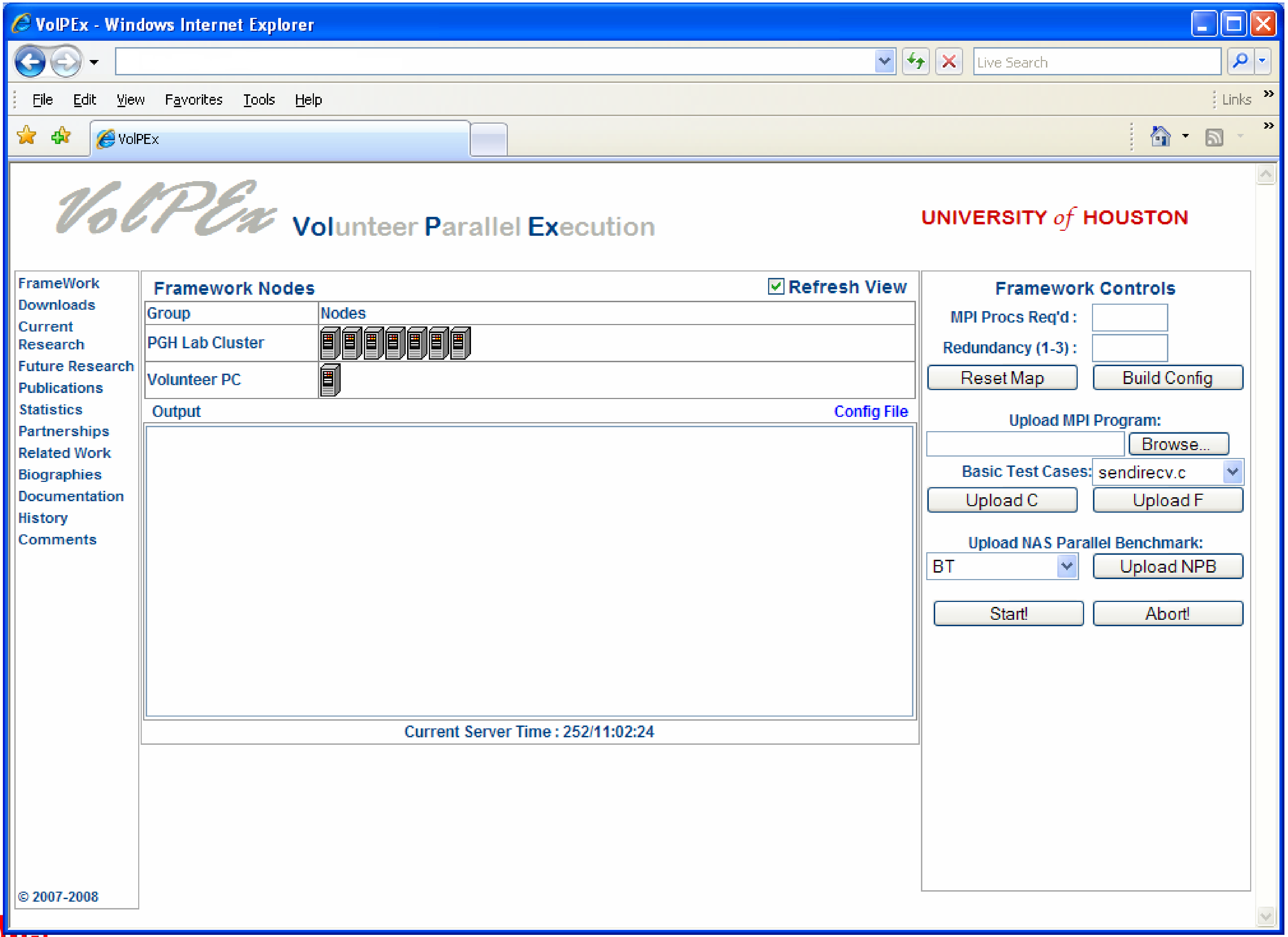
A subset MPI implementation developed for volatile environments

- **Multiple process replicas created**
- **Checkpoint-restart to create replicas – not done**
- **Direct Client to Client communication**

Approach is receivers GET (or PULL) data

On a RECV, the process contacts all possible replicas of potential SENDers repeatedly until the data transfer is complete.

A "Global Map" maps logical processes (MPI ranks) to all physical processes (IP addrs) executing it.



VolPEx - Windows Internet Explorer

VolPEx Volunteer Parallel Execution

UNIVERSITY of HOUSTON

FrameWork
Downloads
Current Research
Future Research
Publications
Statistics
Partnerships
Related Work
Biographies
Documentation
History
Comments

© 2007-2008

Framework Nodes ✓ Refresh View

Group	Nodes
PGH Lab Cluster	▮▮▮▮▮▮▮
Volunteer PC	▮

Output — Config File

Current Server Time : 252/11:02:24

Framework Controls

MPI Procs Req'd :
Redundancy (1-3) :
Reset Map | Build Config

Upload MPI Program: Browse...
Basic Test Cases: sendirecv.c
Upload C | Upload F

Upload NAS Parallel Benchmark: BT | Upload NPB

Start! | Abort!

CS@UH

Conclusions, sort of

This work is trying to extend the class of algorithms/applications that can employ volunteer computing.

We need more collaboration of application folks

- Scenarios where this work will help
- Provide interesting benchmarks
- Guinea pigs for API when it is ready

- Thanks to NSF

- jaspal@uh.edu www.cs.uh.edu/~jaspal