

## Grid Computing: a step-by-step introduction

Youki Kadobayashi, Ph.D.  
Associate Professor  
Graduate School of Information Science,  
Nara Institute of Science and Technology (NAIST),  
Japan

November 12, 2002 / SOI Asia

## What is Grid computing?

- Vision:
  - Utilize computing power much like we utilize electric power today
  - Plug into "computational grid" and obtain the computing power much like we plug into "power grid" for the electric power...

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## What is Grid computing?

- Implementation:
  - Utilize Internet standards as building blocks
  - Re-architect distributed computing in interoperable, scalable manner

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Step 1: Exploit remote computing power

## Before Grid: local computation

- $y = f(x)$
- $x$  and  $y$  are stored locally
- function  $f$  is executed locally
- "procedure call" as we know it
  
- Performance depends on local computer's processor speed, memory latency, bus bandwidth, etc.

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Extending function call to remote machines: RPC

- $y = f(x)$
- $x$  and  $y$  are stored locally
- function  $f$  is "stub" function
  - Sending argument to remote computer
  - "marshall"  $x$  to "wire format"
- remote computer receives wire representation of  $x$ , decodes it, and computes  $f(x)$ 
  - Returns result to requesting computer
  - "marshall"  $y$  to "wire format"

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## What if remote machine is distant?

- RPC overhead
  - o =  $RTT * i(x, y) + e(x) + e(y)$
  - $i$  varies depending on transport protocol, transmitted data size
  - $e(x)$ : marshalling and un-marshalling overhead
- Minimize overhead by maximizing unit of computation
  - from fine-grained RPC to coarse-grained RPC

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## What if remote machine has distinct administrative policy?

- access control policy
  - Machine A accepts computation requests from institution X and Y
- logging
  - Institution X wishes to keep track of who has used its computing resources



- GridRPC

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## GridRPC and RPC: a comparison

- delay tolerance
- access control
- logging
  
- node selection
- registry

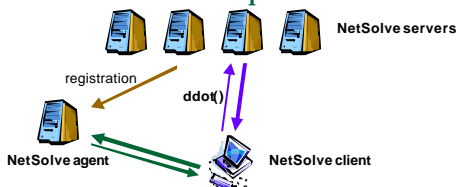
Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## GridRPC in action

```
■ #include <stdio.h>
■ #include <stdlib.h>
■ #include "grpc.h"
■ main()
■ {
■     int i, status;
■     int n = 5;
■     int incx = 1, incy = 1;
■     double ns_result = 0.0;
■     double dx[] = {10.0, 20.0, 30.0, 40.0, 50.0};
■     double dy[] = {60.0, 70.0, 80.0, 90.0, 100.0};
■     grpc_function_handle_t handle;
■
■     if (grpc_initialize(NULL) != GRPC_OK) {
■         grpc_perror("grpc_initialize");
■         exit(EXIT_FAILURE);
■     }
■     /* ddot computes dot product of two vectors-- c.f. BLAS */
■     grpc_function_handle_init(&handle, host, port, "ddot");
■     status = grpc_call(&handle, &n, dx, &incx, dy, &incy, &ns_result);
■     printf("Result from GridRPC = %lf", ns_result);
■     grpc_finalize();
■     exit(EXIT_SUCCESS);
■ }
```

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## GridRPC in action: NetSolve as an example



- Server registers available problems (= functions) to agent
- Agent receives requests from clients and guides them to one of the available servers...

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

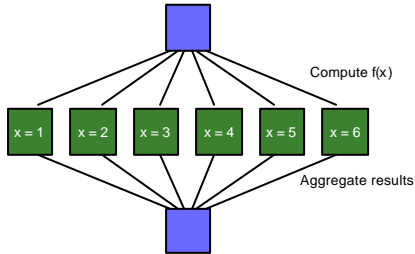
## Performance of GridRPC

- Caution: delay affects overall performance
- Not network-optimized
- May use inefficient wire format (XML)
- Suitable for coarse-grained problems

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## GridRPC application areas: parameter-sweep problems

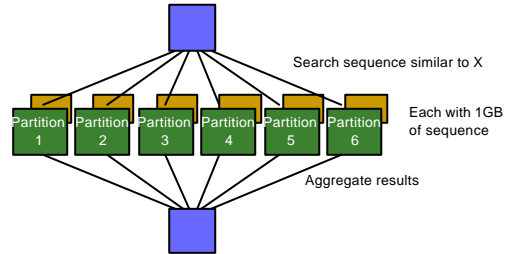
- embarrassingly parallel



Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## GridRPC application areas: parallel processing of data-intensive task

- Partition the search space



Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Question #1

- Describe an application that you think GridRPC can be applied.

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Question #2

- GridRPC alone cannot satisfy both user's requirements and administrator's requirements.

Imagine what kind of potential requirements users and/or administrators might have.

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Step 2: Integrate remote computing resources

## From RPC library to middleware

- Variety of requirements - not just function call
- Rich set of components that lays between OS and user program
  - so called "middleware"
- "Grid toolkit" - middleware for Grid
- Grid toolkit provides variety of services to integrate resources
  - keep track of computing resources
  - secure communication between nodes
  - authenticates user against remote computing resources

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Management of computing resources

- What if you don't know remote node's IP address or hostname?
- How to keep track of computation node's status?



directory service

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Information stored in directory: an example

- static host information
  - OS version, CPU type, number of CPUs, available memory
- dynamic host information
  - load average, available disk space
- network information
  - network bandwidth and latency

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Information stored in directory: an example

gsis.globus.org

- mds-vo-name=vo-index-ou-grid
- Mds-Vo-name=CST
- Mds-Host-Name=ip818r.isi.edu
- Mds-Host-Name=TestLab
- GridClusterUniqueID=Example\_Cluster.isi.edu
  - Mds-Host-Name=gl01.isi.edu
    - Mds-Device-Group-name=filesystems
    - Mds-Device-Group-name=memory
    - Mds-Device-Group-name=networks
    - Mds-Device-Group-name=processors
    - Mds-Device-Name=cpu 0
    - Mds-Software-deployment=MCS
    - Mds-Software-deployment-jdkmanagers-foo
    - Mds-Software-deployment-operating system
  - Mds-Host-Name=ds-user.isi.edu
  - Mds-Host-Name=monte.isi.edu
  - Mds-Host-Name=pygor.isi.edu
  - Mds-Host-Name=TestLab
  - Mds-Vo-name=Cust
  - Mds-Vo-name=anf-dg

Distinguished Name = Mds-device-name=cpu 0, Mds-Host-Name=ip818r.isi.edu

object class = MdsDevice

object class = MdsCpu

object class = MdsCpuCache

Mds-Device-name = cpu 0

Mds-Cpu-vendor = GenuineIntel

Mds-Cpu-model = Intel(R) XEON(TM) CPU 2

Mds-Cpu-version = 15.0.0

Mds-Cpu-features = fpu vme de ppe tac mar pae tm

Mds-Cpu-speedMHz = 0

Mds-Cpu-cache-2KB = 512

Mds-validfrom = 20021107084916Z

Mds-validto = 20021107085016Z

Mds-accepto = 2002110034236Z

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Resource selection with directory service

- Some grid toolkits provide methods to select computation node with variety of constraints, e.g.:
  - number of CPUs
  - amount of available memory

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Summary (of Day 1)

- Grid computing = inter-domain RPC + resource abstraction + resource management?
- Many other ingredients will be added later

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.

## Preview of Day 2

- Step 2 (resource integration) continued
  - Security
  - Data management
  - Some research-oriented topics
- Grid standardization
- Grid operation

Copyright(C)2002 Youki Kadobayashi / NAIST. All rights reserved.