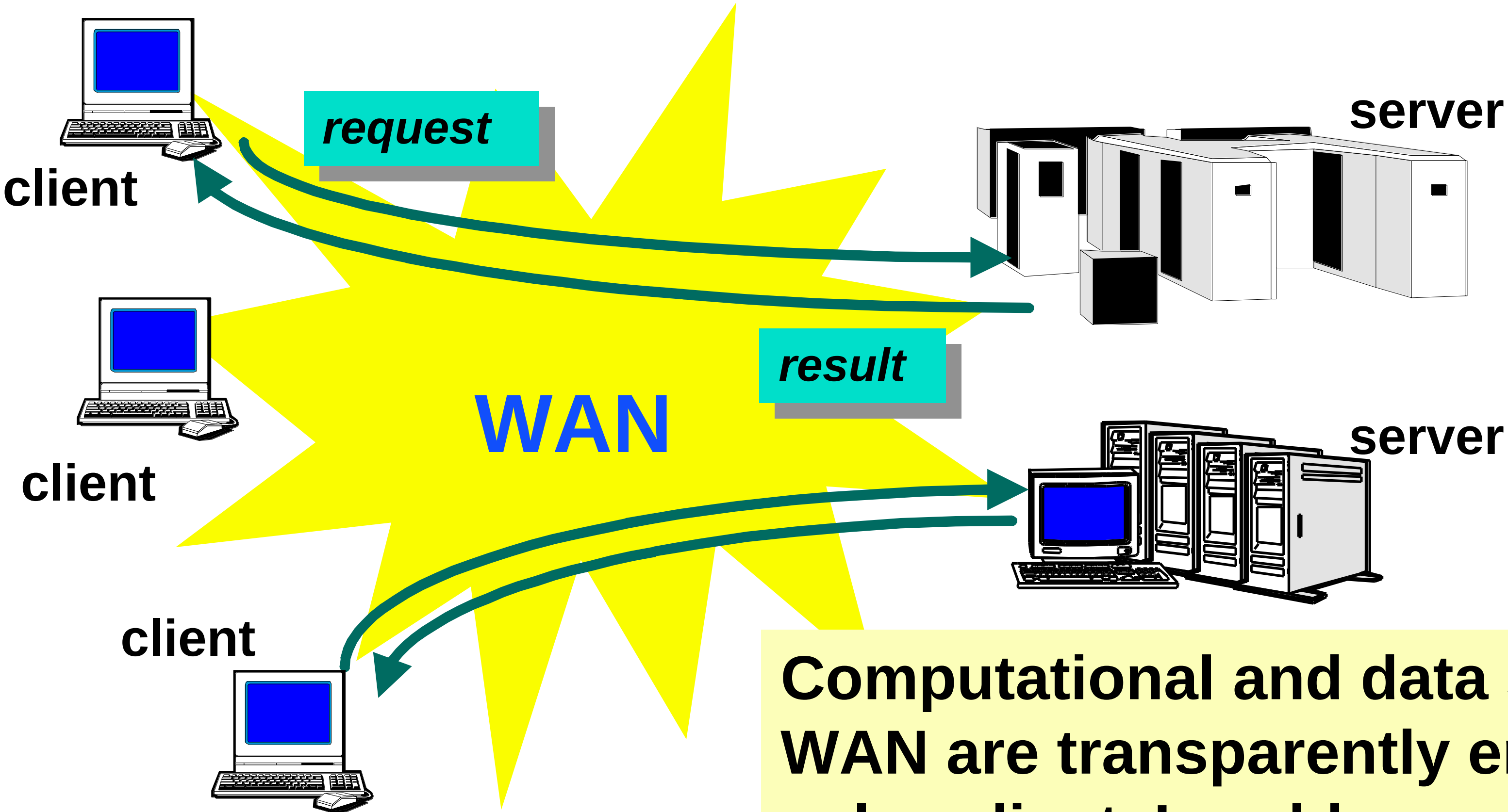
A Performance Evaluation Model for Scheduling in Global Computing Systems

Kento Aida (TIT)
Atsuko Takefusa (Ochanomizu Univ.)
Hidemoto Nakada (ETL)
Satoshi Matsuoka (TIT)
Satoshi Sekiguchi (ETL)
Umpei Nagashima (National Institute of Materials and Chemical Research)

http://ninf.etl.go.jp/

Global Computing System



Computational and data servers in WAN are transparently employed to solve clients' problems.

Proposed Global Computing Systems:

- Globus, Legion, NetSolve, Ninf, RCS, etc.

Scheduling for Global Computing System

An effective resource allocation / scheduling is required to achieve high-performance global computing!

Scheduling among computational servers

- under Dynamic, Hetero. Env.
  - computing server performance / load
  - network topology / bandwidth / congestion
  - multiple users at multiple sites

Software Systems for Effective Scheduling

- AppLeS, NetSolve agent, Nimrod, Ninf Metaserver, Prophet, etc.

Framework to evaluate scheduling

Benchmarking on Real Systems

- practical measurement
- difficult to perform large-scale experiments
- a small number of replications
- partial solution

No Effective Frameworks to evaluate the performance of scheduling in global computing systems!

Performance Evaluation Model

Objective

- modeling various global computing systems
- large-scale simulation
- reproducibility

Contents

- overview of the model
- verification of the model
- evaluation of scheduling algorithm on the model

General Arch. of Global Computing System

Clients

Computing Servers

Scheduling System

- Schedulers (e.g. AppLes, Prophet)
  - perform scheduling according to system / user policy
- Directory Service (e.g. Globus-MDS)
  - central database of resource information
- Monitors/Predictors (e.g. NWS)
  - monitor and predict server and network status

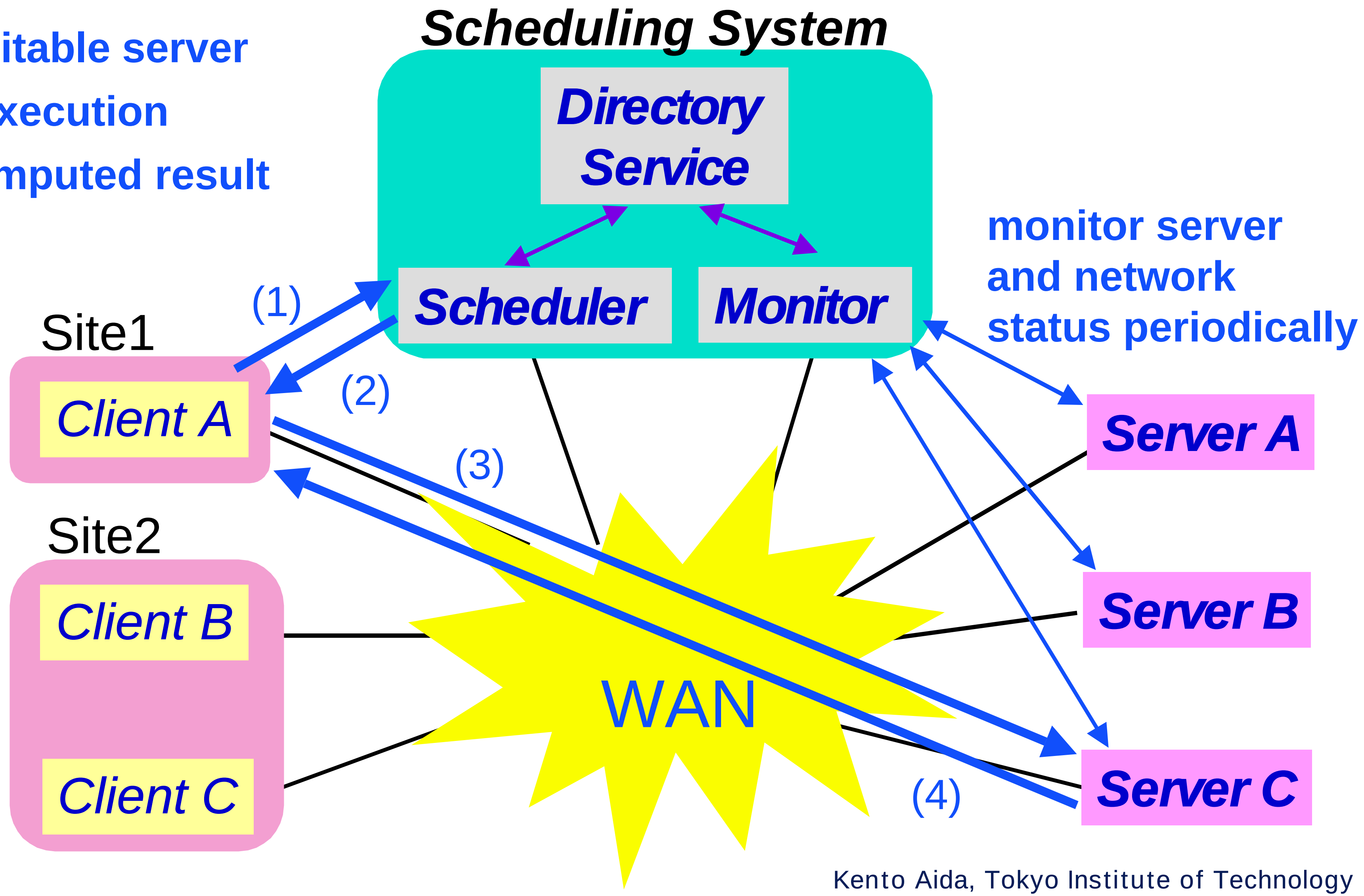
Canonical Model of Task Execution

(1) query for suitable server

(2) assign suitable server

(3) request execution

(4) return computed result



Requirements for the Model

Model

- topology
  - clients, servers, networks
- server
  - performance, load (congestion), variance over time
- network
  - bandwidth, throughput (congestion), variance over time

Perform

- large-scale simulation
- reproducible simulation

Proposed Performance Evaluation Model

Queueing Network

Global Computing System

- Qs computational servers
- Qns network from the client to the server
- Qnr network from the server to the client

Congestion on Servers and Networks

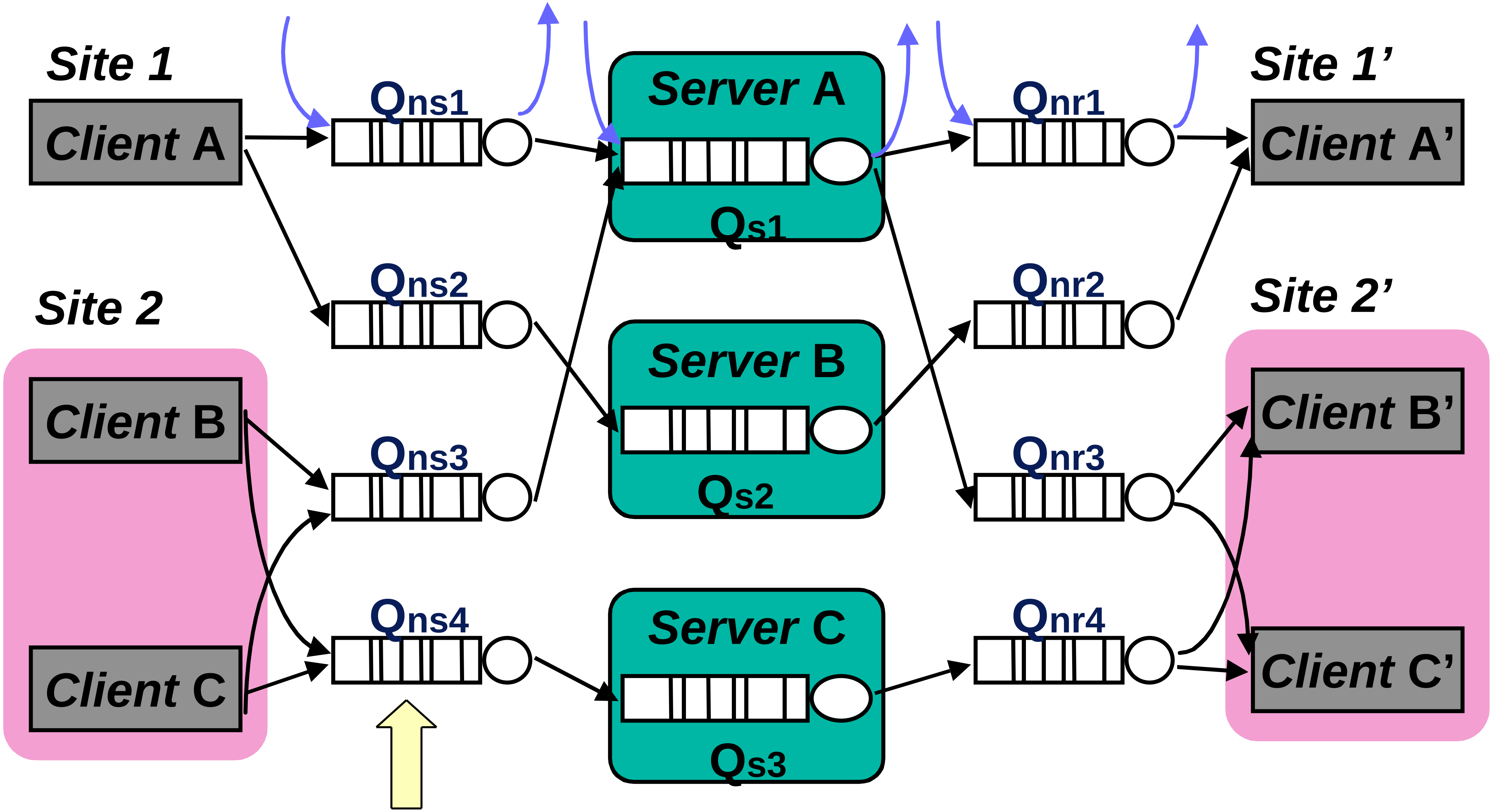
- 'other' tasks

  tasks which are invoked from other processes and enter Qs

- 'other' data

  data which are transmitted from other processes and enter Qns or Qnr

Example of Proposed Model



The net work is shared by Client B and C

Client

Task Invoked by a Client

- data transmitted to the server (D_{send})
- computation of the task
- data transmitted from the server, or computed result (D_{recv})

Procedure to Invoke Tasks

- query the scheduler for a suitable server

  The scheduler assigns a server.

- decompose D_{send} into logical packets and transmit these packets to Q_{ns} connected to the assigned server

  The server completes the execution of the task.

- receive D_{recv} from Q_{nr}

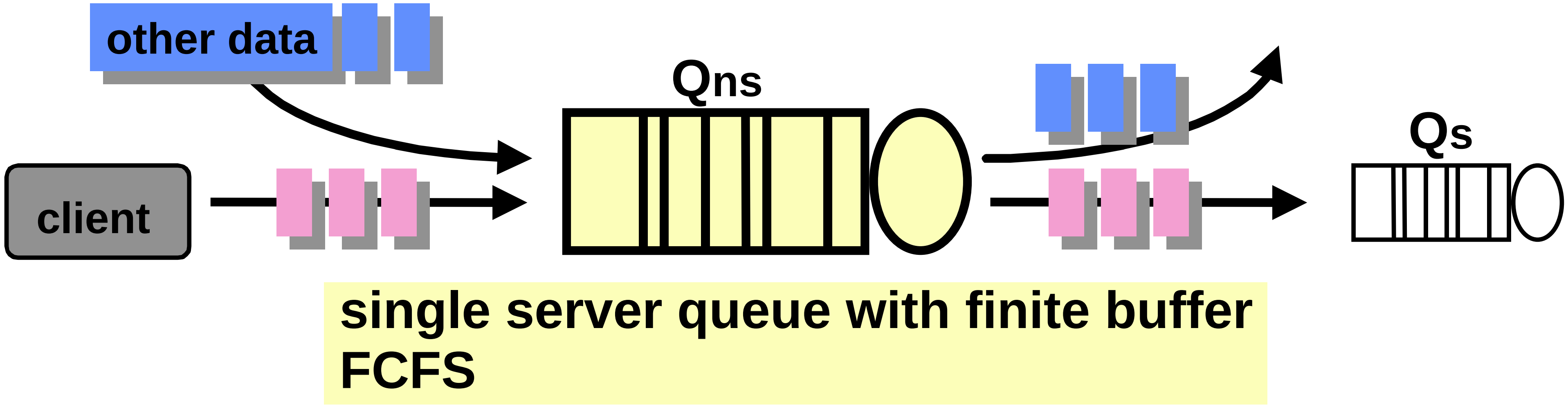
Parameters for the Client

Packet Transmission Rate

$$\lambda_{packet} = T_{net} / W_{packet}$$

T_{net}	bandwidth of the network between the client and the server
W_{packet}	logical packet size

Queue modeling Client-to-Server Network (Q_{ns})



Process

- A packet transmitted from the client enters Q_{ns}.
- A packet is retransmitted when buffer is full.
- A packet in Q_{ns} is processed for [W_{packet} / T_{net}] time.
- A packet of the client's task leaves for Q_s.

Arrival rate of other data indicates congestion of the network.

Parameters for Qns

Arrival Rate of Other Data

- determines network throughput
- Arrival is currently assumed to be Poisson.

$$\lambda_{ns\_others} = (T_{net} / T_{act} - 1) \times \lambda_{packet}$$

T_{act} avg. actual throughput of the network to be simulated

Buffer Size of Queue

- determines network latency

$$N = T_{latency} \times T_{net} / W_{packet}$$

$T_{latency}$ avg. actual latency of the network to be simulated

Example

Simulated Condition

- bandwidth — $T_{net} = 1.0$ [MB/s]
- avg. actual throughput — $T_{act} = 0.1$ [MB/s]
- latency — $T_{latency} = 0.1$ [sec.]
- logical packet size — $W_{packet} = 0.01$ [MB]

Arrival Rate of Other Data and Latency

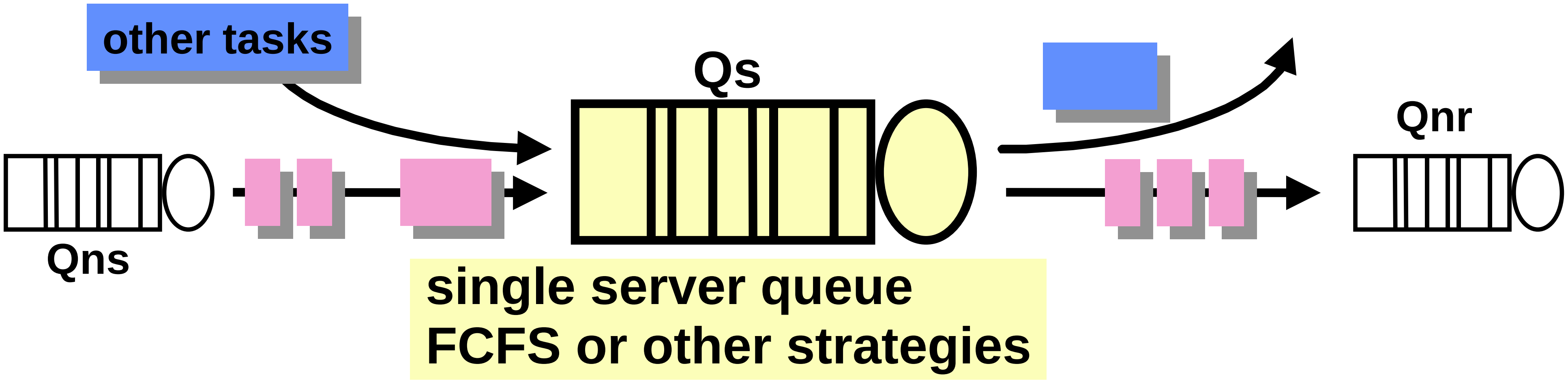
$\lambda_{packet} = T_{net} / W_{packet} = 1.0 / 0.01 = 100$

$\lambda_{ns\_others} = (T_{net} / T_{act} - 1) \times \lambda_{packet}$

$= (1.0 / 0.1 - 1) \times 100 = 900$

$N = T_{latency} \times T_{net} / W_{packet} = 0.1 \times 1.0 / 0.01 = 10$

Queue Modeling Server Behavior (Qs)



Process

- The computation of the client's task enters Qs after all associated data arrive at Qs.
- A queued task waits for its turn and is processed for [Wc / Tser] time. (Tser : server performance, Wc : avg. comput. size)
- Data of computed result are decomposed into logical packets and these packets are transmitted to Qnr.

Arrival rate of other tasks indicates congestion on the server.

Parameters for Qs

Arrival Rate of Other tasks

- determines server utilization
- Arrival is currently assumed to be Poisson.

$$\lambda_{s\_others} = T_{ser} / W_{s\_others} \times U$$

T_{ser} performance of the server
W_{s_others} avg. computation size of other tasks
U avg. actual utilization on the server to be simulated

Packet Transmission Rate

$$\lambda_{packet} = T_{net} / W_{packet}$$

Example

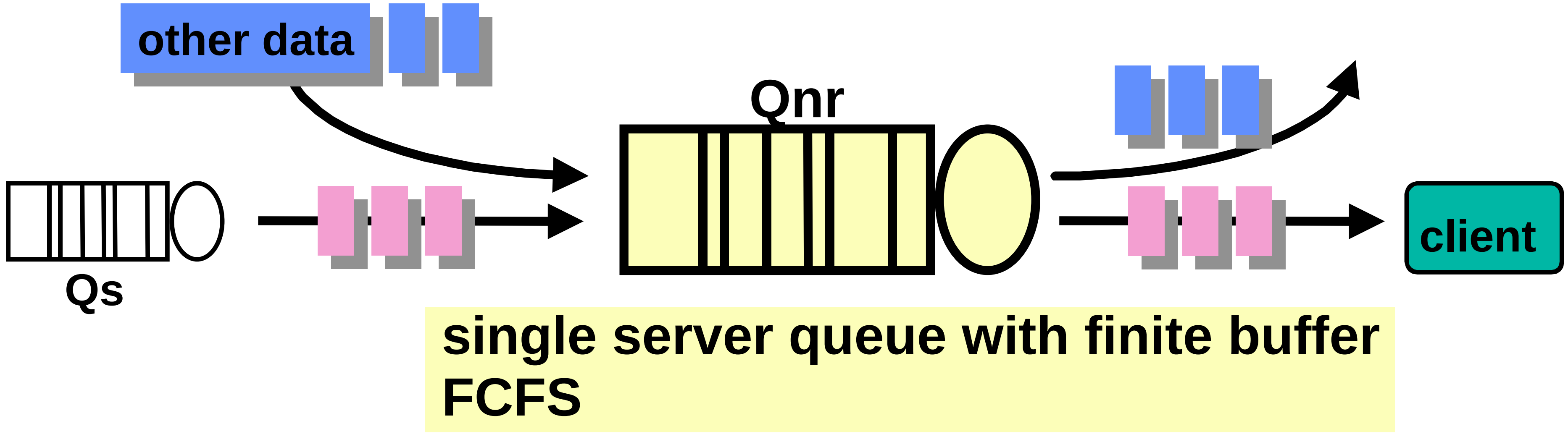
Simulated Condition

- server performance $T_{ser} = 100$ [MFlops]
- avg. actual utilization $U = 0.1$
- avg. computation size $W_{s\_others} = 0.1$ [MFlops]

Arrival Rate of Other Tasks

$$\lambda_{s\_others} = T_{ser} / W_{s\_others} \times U$$
$$= 100 / 0.1 \times 0.1$$
$$= 100$$

Queue Modeling Server-to-Client Network (Qnr)

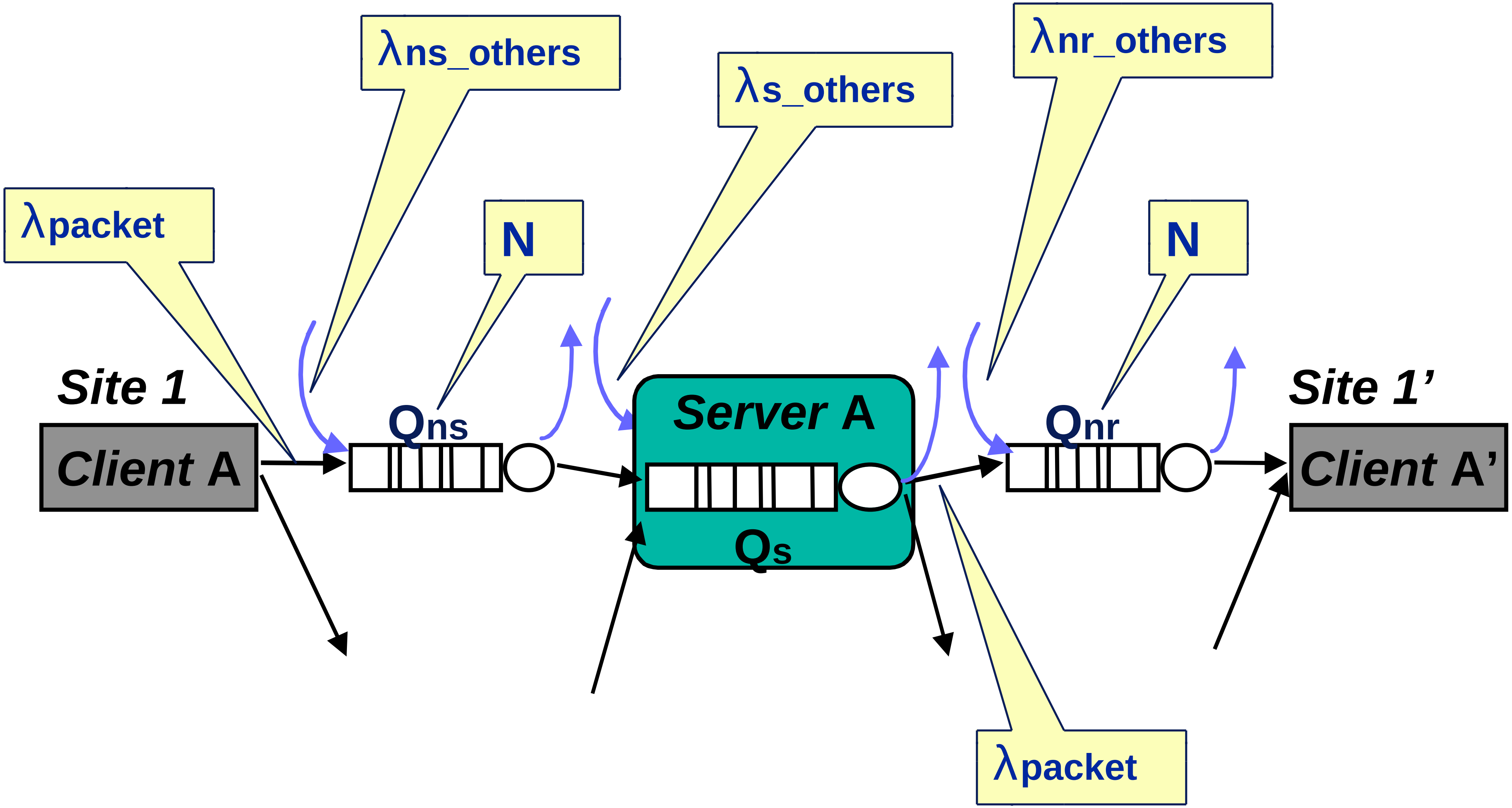


Process

- A packet transmitted from **Qns** enters **Qnr**.
- A packet is retransmitted when buffer is full.
- A packet in **Qnr** is processed for [W_{packet} / T_{net}] time.
- A packet transmitted from **Qns** leaves for the client.

Arrival rate of other data indicates congestion of the network.

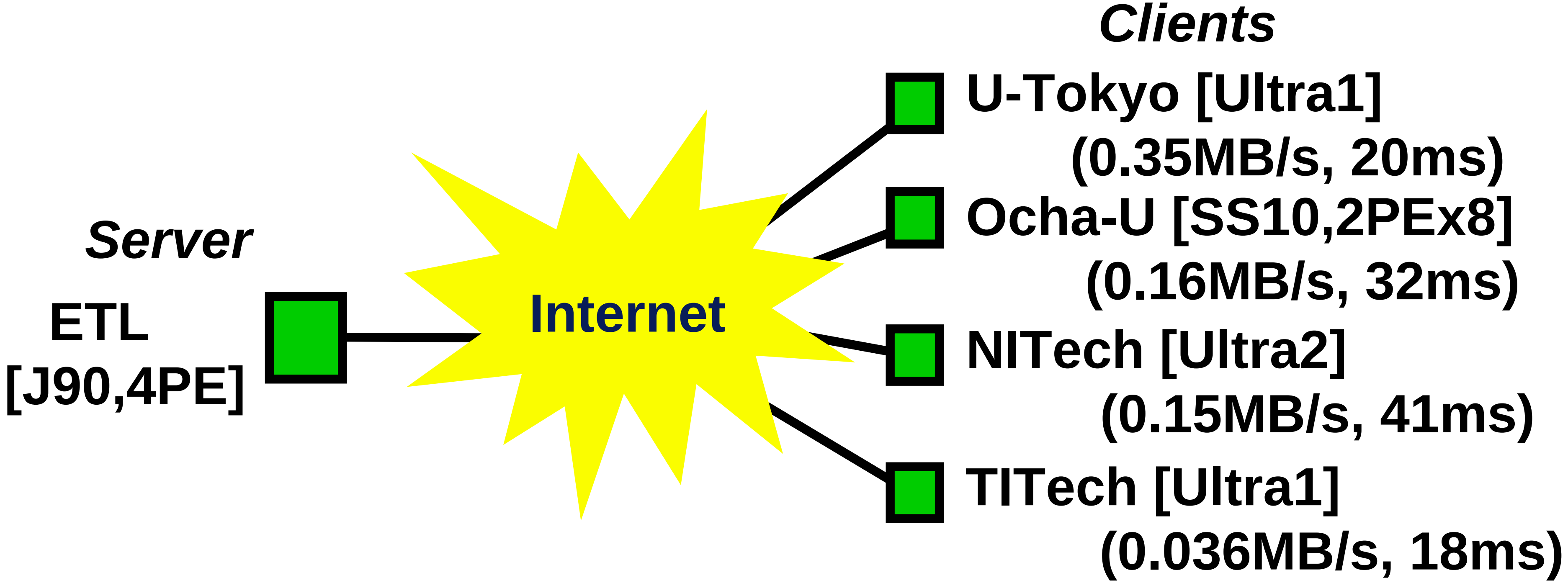
Summary of Parameters



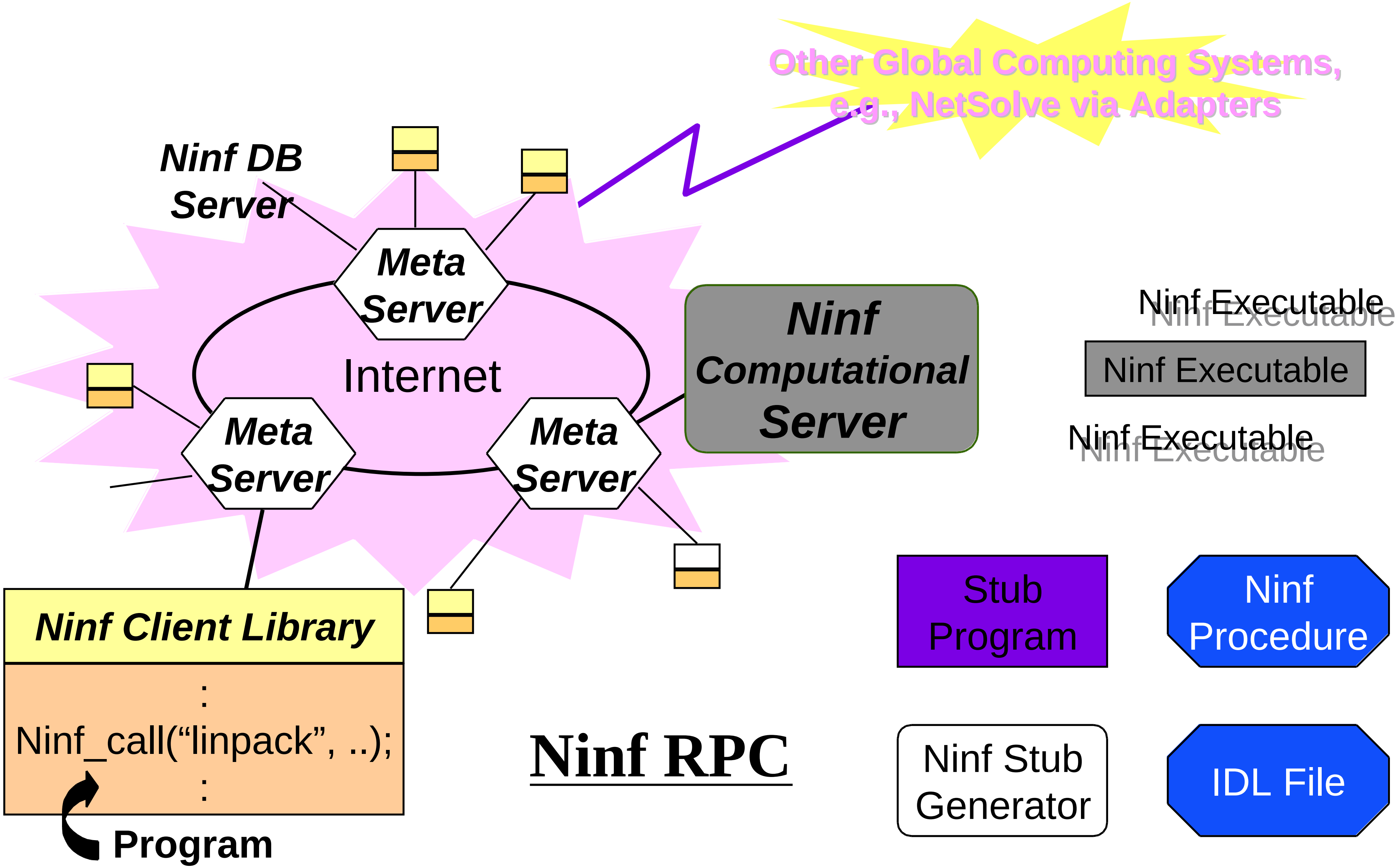
Verification of the Proposed Model

Comparison

- results in simulation on the proposed model
- results in experiments on the actual global computing system, Ninf system



Ninf System


Other Global Computing Systems,
e.g., NetSolve via Adapters
Ninf DB Server
Meta Server
Meta Server
Meta Server
Internet
Ninf Computational Server
Ninf Executable
Ninf Executable
Ninf Executable
Ninf Client Library
:
Ninf_call("linpack", ..);
:
Program
Ninf RPC
Stub Program
Ninf Procedure
Ninf Stub Generator
IDL File

Simulation Parameters (1)

Client

- invoking tasks repeatedly

  Linpack (problem size = 600, 1000, 1400)

  (comput. = $O(2/3n^3 + 2n^2)$, comm.= $8n^2 + 20n + O(1)$)

- invocation rate of Ninf_call at the client

  invoke tasks in non-overlapping manner

  $\lambda_{request} = 1 / (\textit{worst response time} + \textit{interval})$

- packet size = 10, 50, 100 [KB]

  small packet size ➡ accurate network simulation

  large packet size ➡ short simulation time

Simulation Parameters (2)

Network

- bandwidth = 1.5[MB/s]
- other data

  avg. packet size = 10, 50, 100[KB] (Exp. Dist.)

  Poisson Arrival

Server

- CPU performance = 500[MFlops]
- avg. actual utilization = 0.04

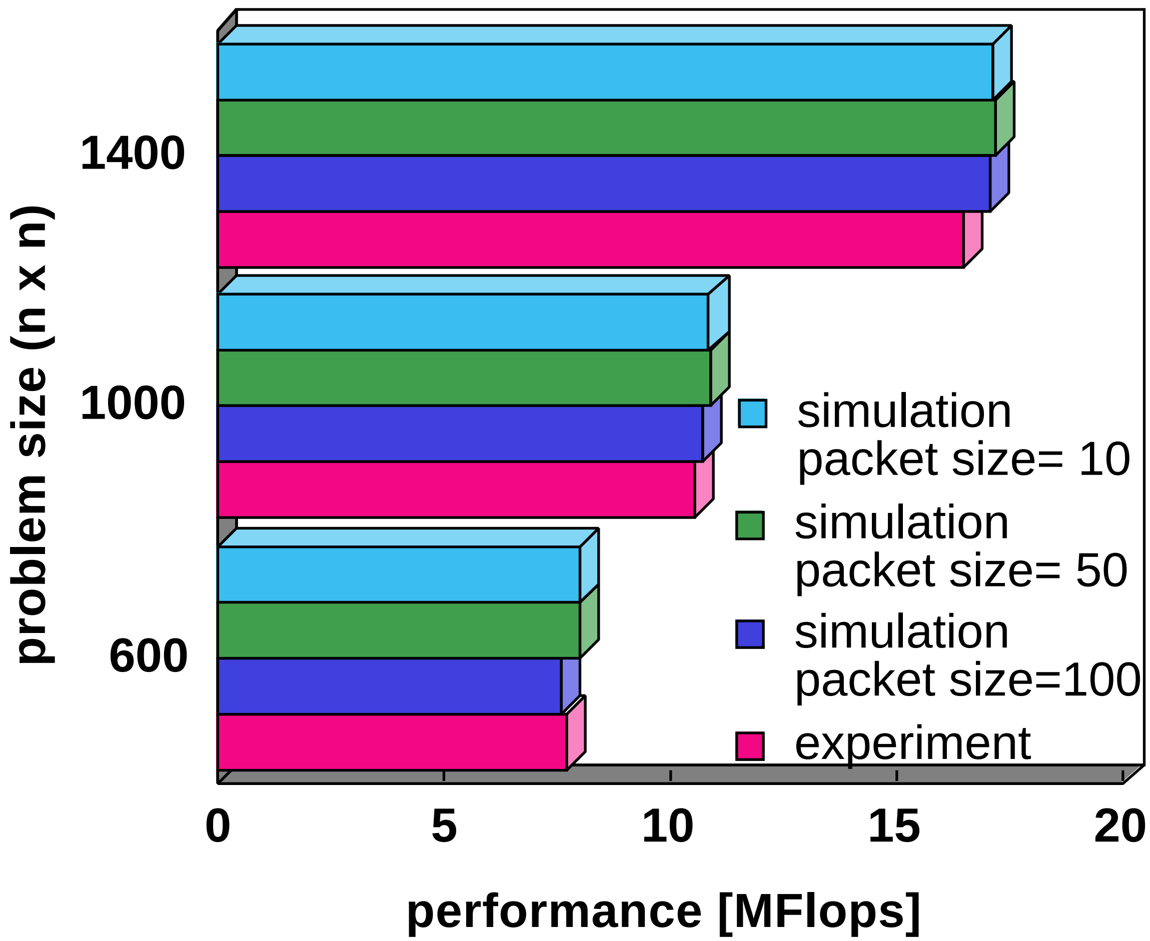
  other tasks

  avg. computation size = 10[MFlops] (Exp. Dist.)

  Poisson Arrival

Performances of a Client's Tasks

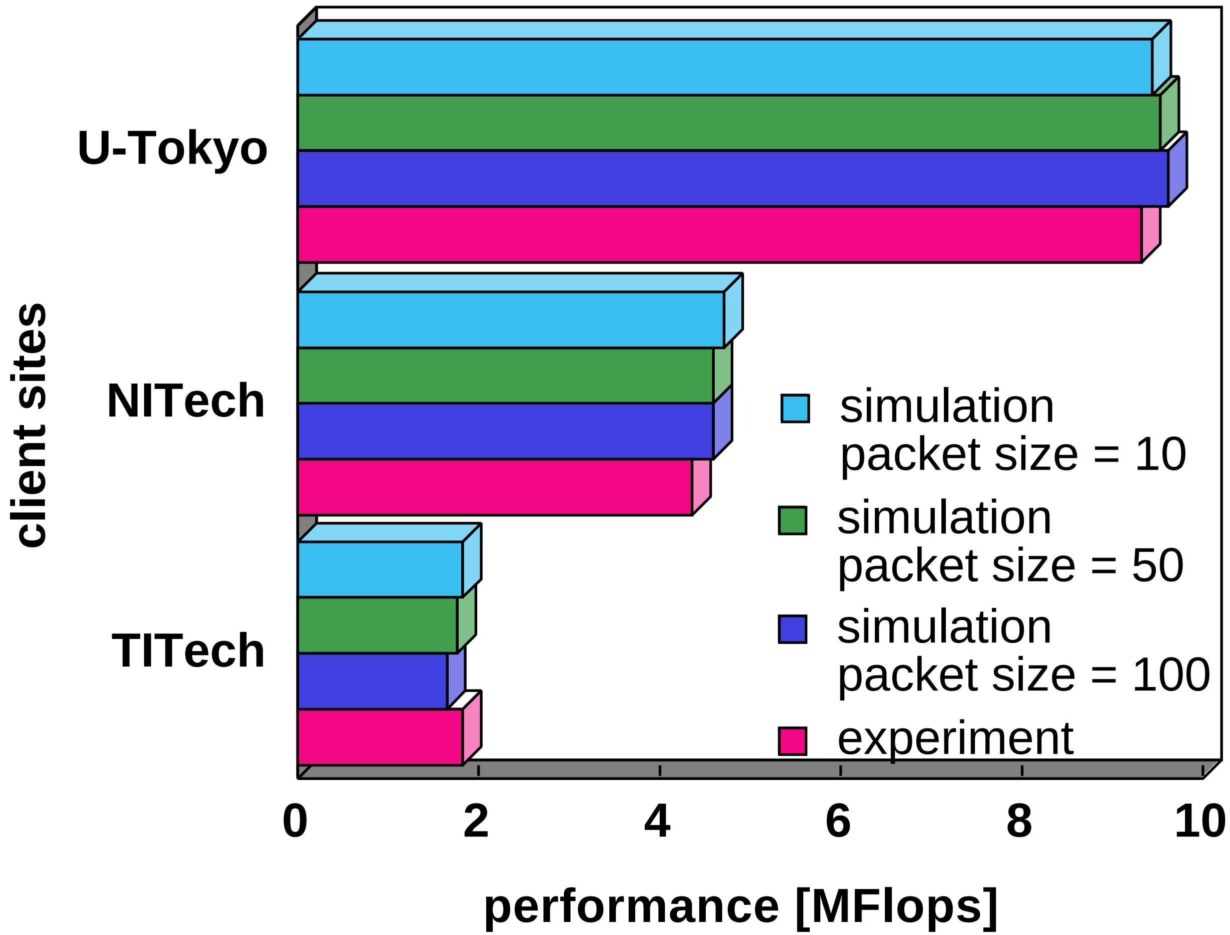
client : WS in Ochanomizu Univ., server : J90 in ETL



- **The performances of a client's tasks in the simulation closely matches experimental results.**
- **Effect of different packet sizes is almost negligible.**
- **Simulation cost could be reduced.**

Performances of Clients' Tasks

clients : WS in U-Tokyo, NITech and TITech,
server : J90 in ETL



- **The performances of tasks invoked by multiclients in the simulation closely matches experimental results.**
- **Effect of different packet sizes is almost negligible.**
- **Simulation cost could be reduced.**

Evaluation of Scheduling Algorithm

Evaluation

- **Evaluation of basic scheduling algorithm on imaginary environment in the simulation on the proposed model**

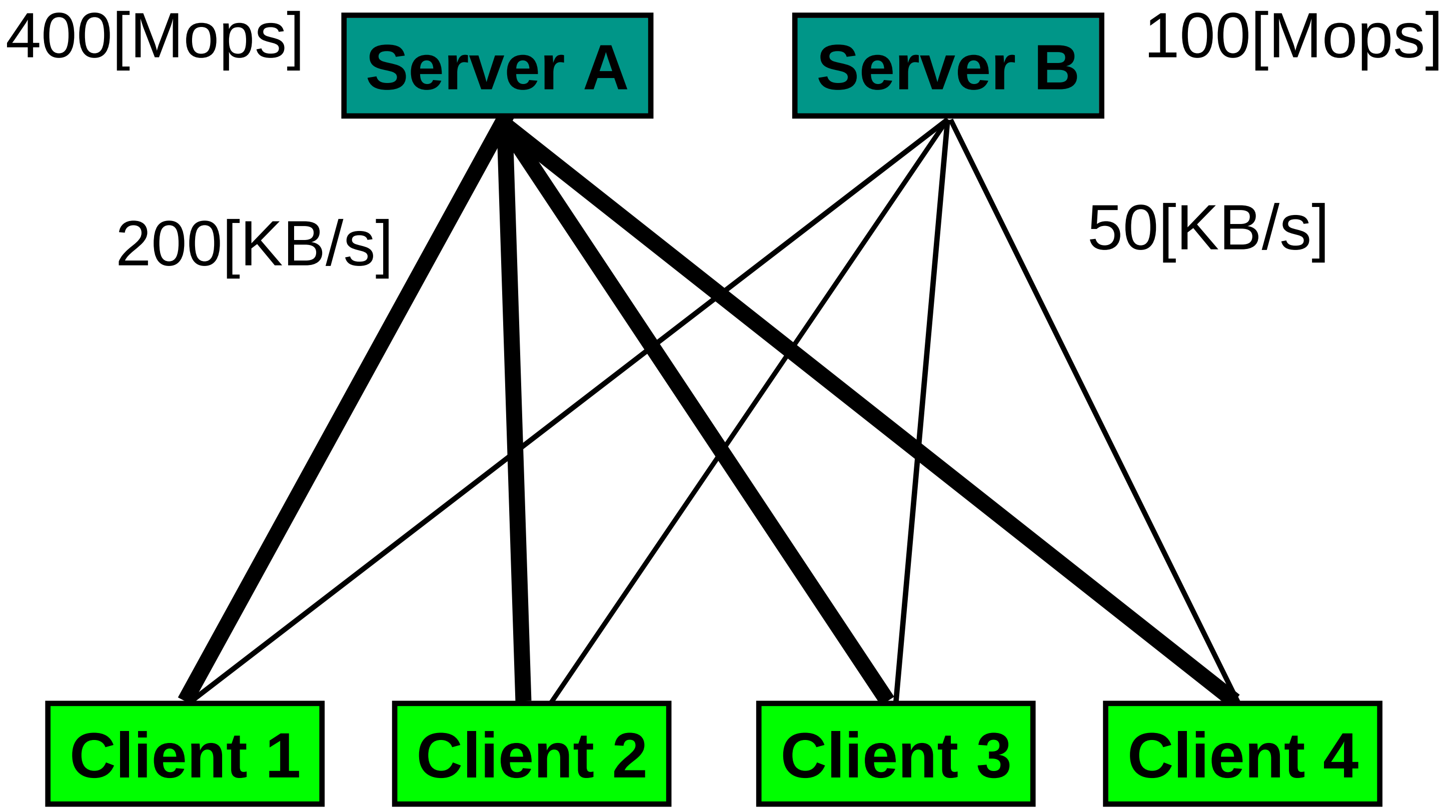
Scheduling Algorithm

- **RR** **round robin**
- **LOAD** **server load**

  min. (L + 1) / P (L : avg. load, P : server performance)
- **LOTH** **server load + network congestion**

  min. Compt. / (P / (L + 1)) + Comm. / T_{net}

Imaginary Environment



Simulation Parameters (1)

Client

- **invoking tasks repeatedly**

  Linpack (problem size = 600)

  (comput. = $O(2/3n^3 + 2n^2)$, comm.= $8n^2 + 20n + O(1)$)

  EP (problem size = 2^{21})

  (comput. = number of random number, comm. = $O(1)$)

- **invocation rate of Ninf_call at the client**

  $\lambda_{request}$ = 1 / (worst response time + interval)

  interval: Linpack 5[sec.], EP 20[sec.]

  Poisson Arrival

- **packet size = 100 [KB]**

Simulation Parameters (2)

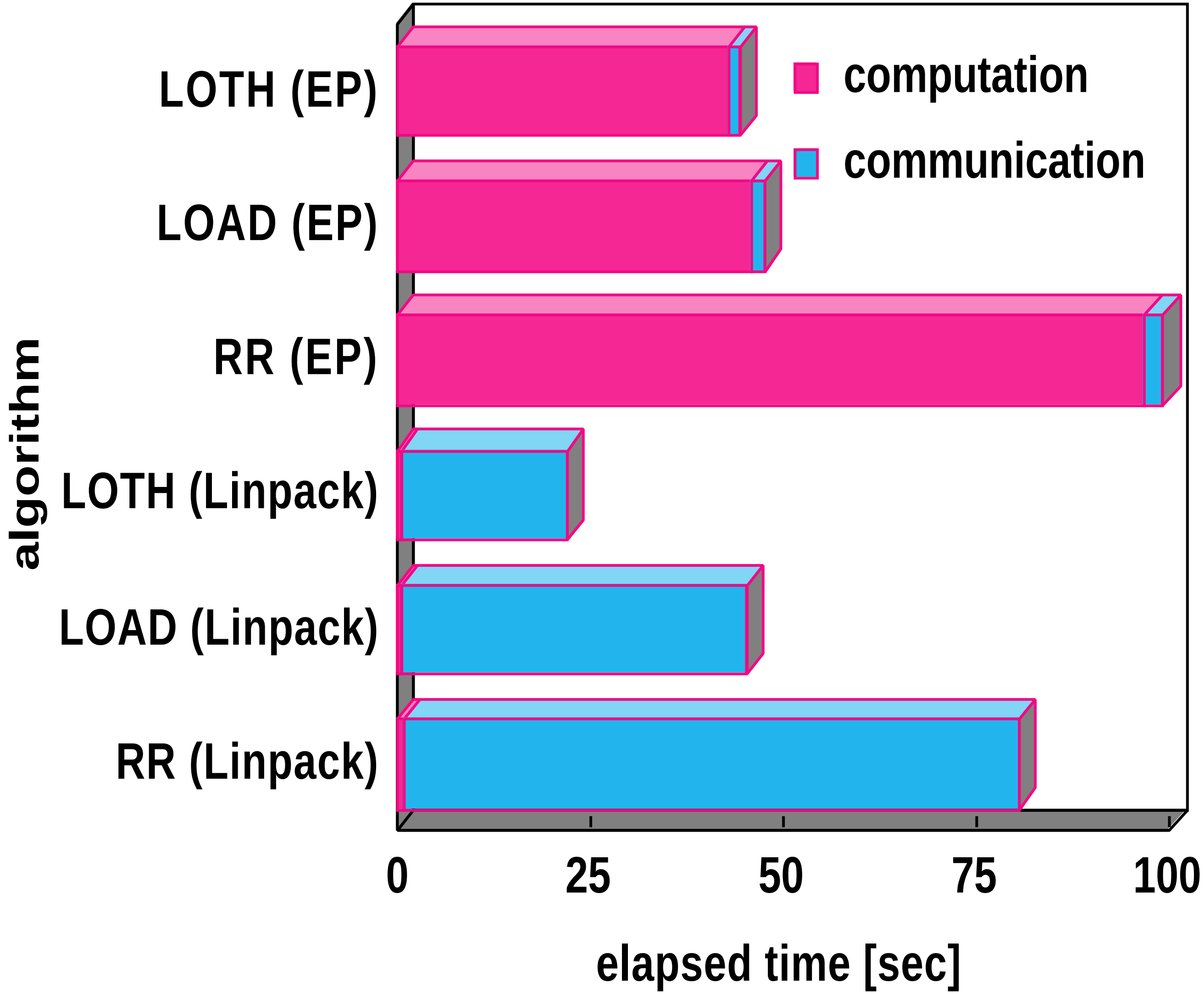
Network

- bandwidth = 1.5[MB/s]
- other data
  - avg. packet size = 100[KB] (Exp. Dist.)
  - Poisson Arrival

Server

- avg. actual utilization = 0.1
- other tasks
  - avg. computation size = 10[Mops] (Exp. Dist.)
  - Poisson Arrival

Scheduling Performances



RR
- performs worst

LOAD
- performs well for EP
- causes network congestion and degrades performance for Linpack

LOTH
- performs best

Conclusions

Proposal

- performance evaluation model for scheduling in global computing systems

Verification of the Model

- The proposed model could effectively simulate the performances of clients' tasks in simple setup of the actual global computing system, Ninf system.

Evaluation on the Model

- Dynamic information of both servers and networks should be employed for scheduling.

Future Work

Modeling

- parallel task execution
  - invocation of parallel tasks at the client
  - Inter-server communication / synchronization
  - co-allocation of parallel tasks
- application scheduling
  - AppLeS, etc.
- arrival of other data / task

Developing Scheduling Algorithm

- prediction of server load and network congestion
  - NWS