Modelling and Improving Group Communication in Server Operating Systems

Michael Kwok, Tim Brecht

Martin Karsten, Jialin Song

University of Waterloo, Canada

Introduction

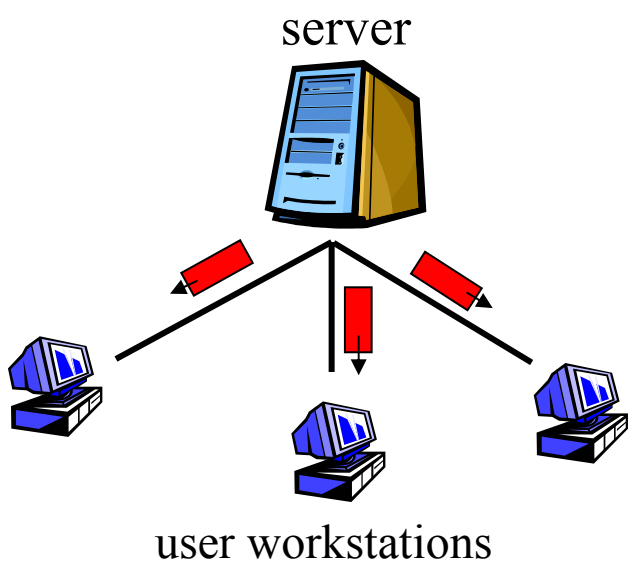
- Many app. send data to groups of recipients
- Need efficient group communication mechanisms
- Need an analytic framework
  - Micro-benchmarking → macro-benchmarking

Example Application – DVE

- DVE (Distributed Virtual Environment)
  - A shared virtual environment
  - May involve a large number of simultaneous users
  - E.g., multi-player online games, CSCWs

Example Application – DVE

- Updates need to be distributed as soon as possible
- DVE system
  - Client-server
  - UDP
- Concerned with the efficiency of the group communication mechanism used



Group Communication

- Common approach:

  user-level unicast (*user-groupcast*)

```
for (i=0; i<GRPSIZE; i++) {
    fds[i]=socket(PF_INET, SOCK_DGRAM, 0);
}
for (i=0; i<GRPSIZE; i++) {
    totalbytes+=send(fds[i], buf, bytes);
}
```

Group Communication

- Proposed approach:

  kernel-level unicast (*kernel-groupcast*)

```
grp=socket(PF_INET, SOCK_DGRAM, 0);
setsockop(grp, SOL_SOCKET, SO_SETGRP,
    addrs, GRPSIZE*sizeof(struct
    sockaddr_in));
totalbytes=send(grp, buf, bytes);
```

Problem

- How does kernel-groupcast improve the server capacity?

- Develop an analytic framework
  - Translate improved groupcast efficiency into server capacity improvement

$$\frac{\max N \text{ with new groupcast}}{\max N \text{ with original groupcast}}$$

Immediate Send Model

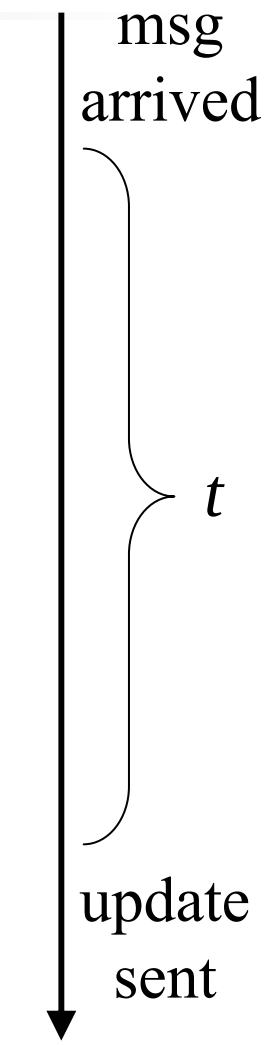
- Pseudo-code

```
while (1) {
  receive_client_msg();
  process_client_msg();
  send_update_to_group();
}
```

H (receive_client_msg(); process_client_msg();)

S (send_update_to_group();)

- $\lambda N(H+S) \leq T$
  - T : max. avg. t
  - λ : avg. msg. from a user per T
  - N : no. of users

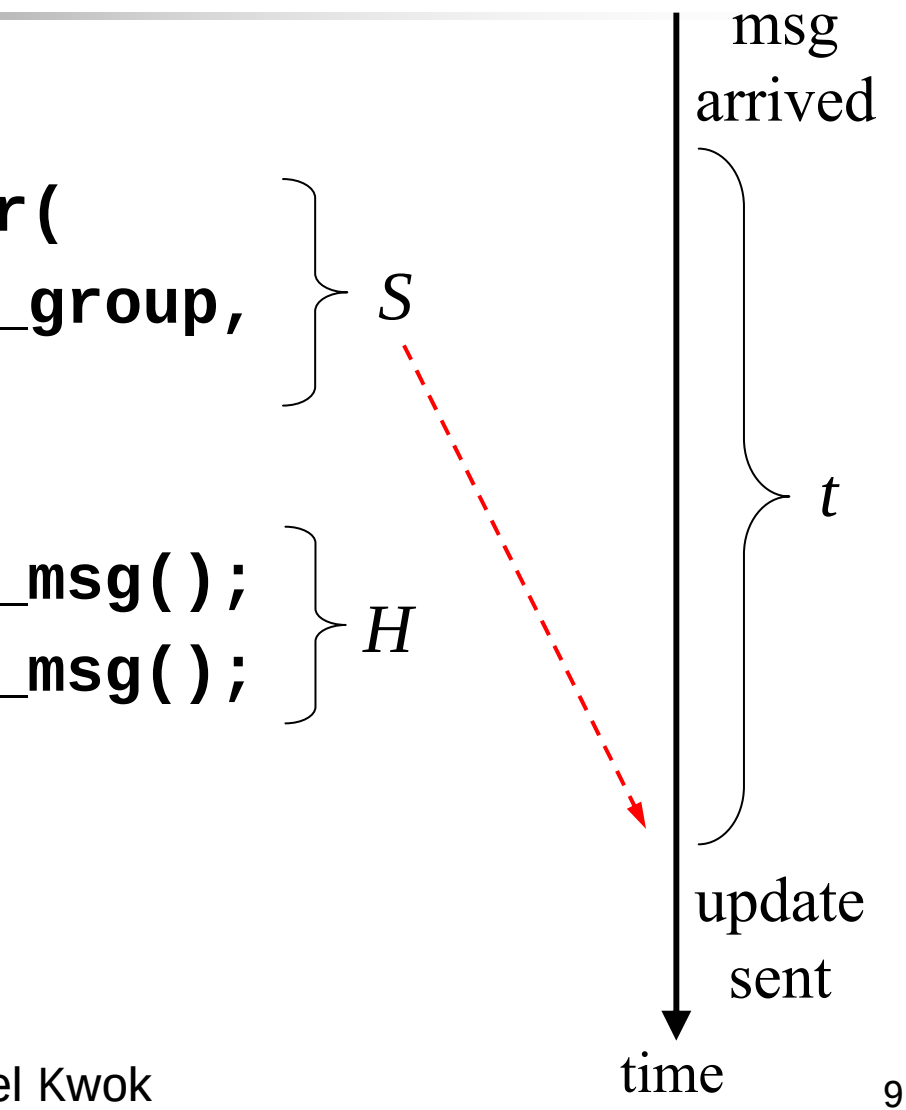


Periodic Send Model

- Pseudo-code

```
set_alarm_handler(
  send_update_to_group,
  period);
while (1) {
  receive_client_msg();
  process_client_msg();
}
```

- $\lambda NH + S \leq T$



Analytic Framework

- Of interest is the capacity improvement ratio:

$$k = \frac{N_{new}^{max}}{N_{orig}^{max}}$$

Parameter Characterization

- Send operation, *S*
  - Characterize *S* by measurement experiments
  - Based on our results, *S* can be best described by

$$xN + yNb$$

where *x* and *y* are constants

Parameter Characterization

- Other operation, H
  - Consider two example cases
    1. Constant: $H_1 := c_1$
    2. Linear: $H_2 := c_2 N$

    where c_1 and c_2 are constants

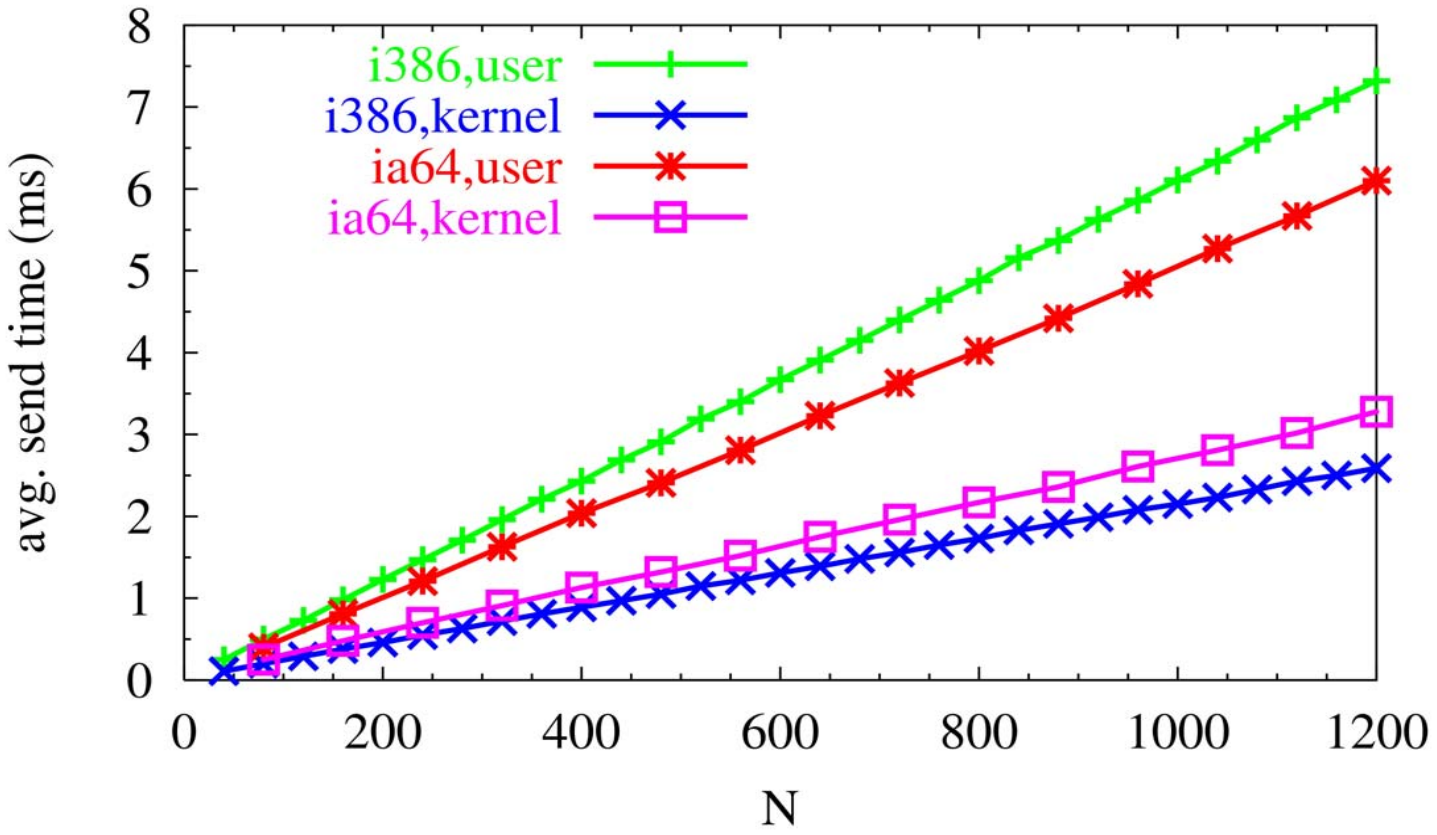
Numerical Examples

- User-groupcast vs. kernel-groupcast
- Conduct micro-benchmarks on a server running on
  - i368: a 2.8 GHz Intel Xeon
  - ia64: a 900 Mhz Intel Itanium2

  to characterize *S*

Numerical Examples

- Experimental results



University of Waterloo

Numerical Examples

- To characterize H, define

$$f = \frac{\text{total send time}}{\text{total processing time}}$$

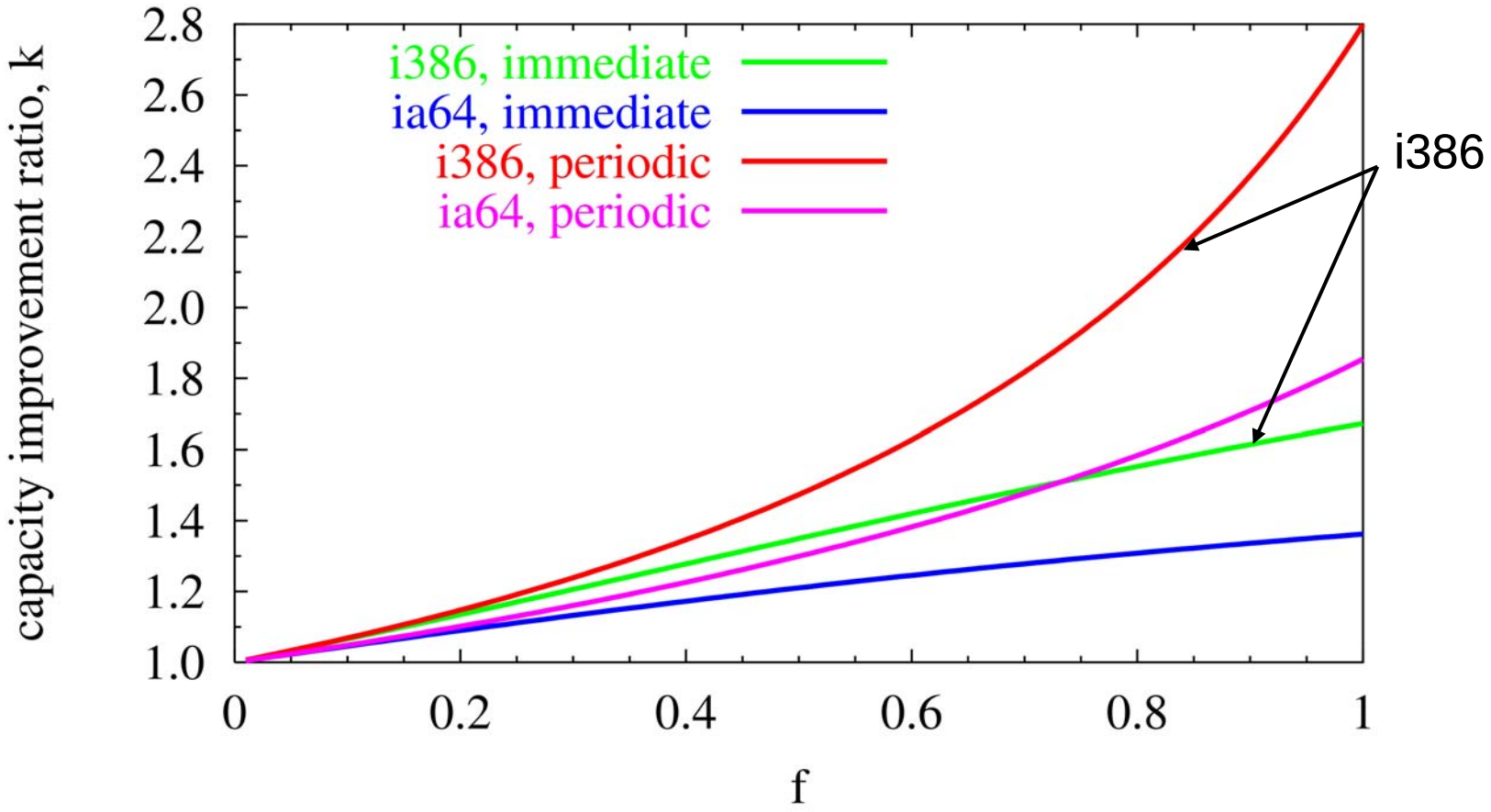
- Vary f between 0 and 1

Numerical Examples

- Other input parameters
  - $T = 33.3$ ms (30 fps refresh)
  - $\lambda = 1$ (avg. 1 msg per period)
  - $b = 80$ bytes (avg. msg size [Farber 2002])

Numerical Results

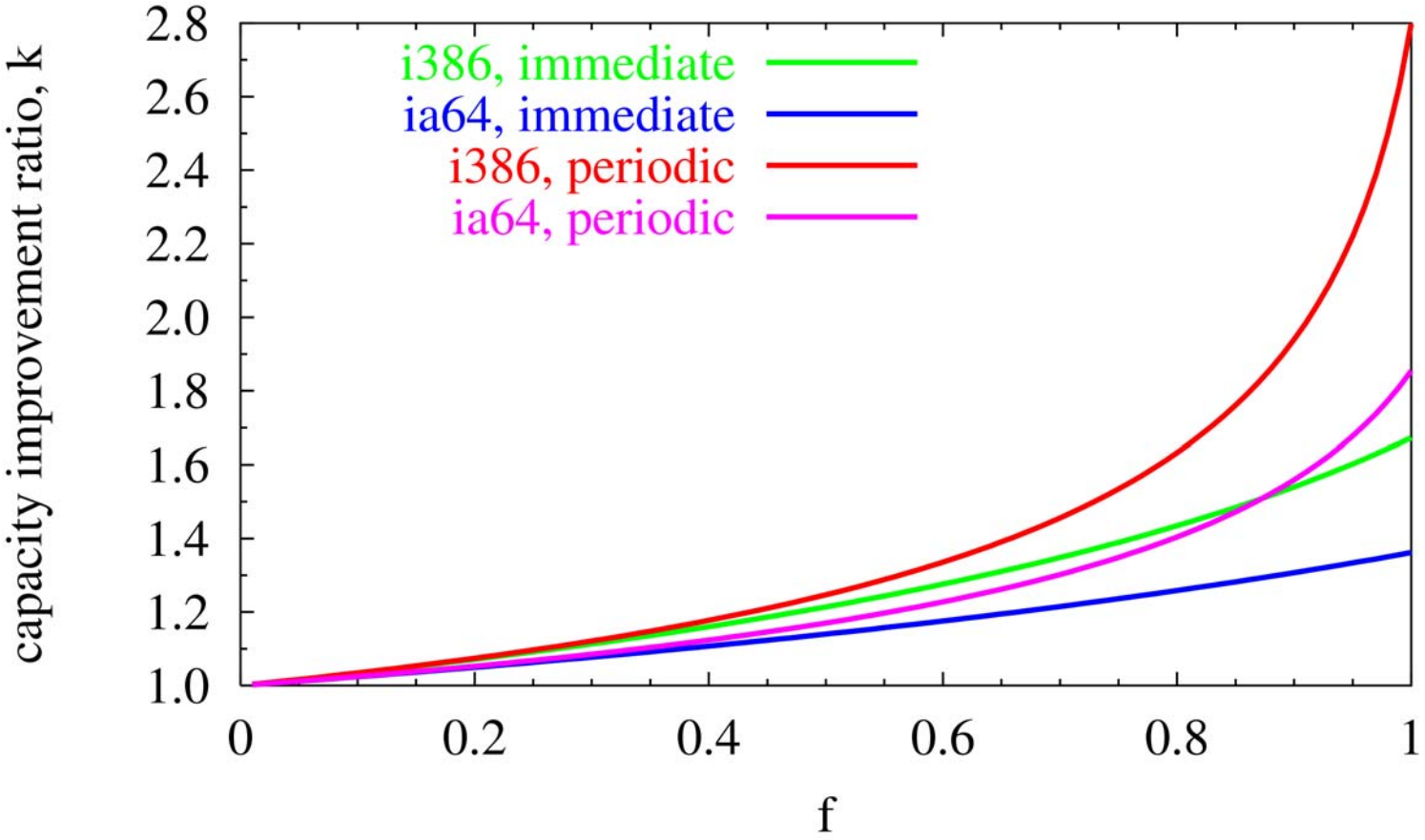
- H_1 – constant



University of Waterloo

Numerical Results

- H_2 – linear



University of Waterloo

Bounds on Capacity Improvement

- For the immediate model

$$k \leq \sqrt{r}$$

where r is the speedup factor defined as $r = \frac{S_{orig}}{S_{new}}$

- For the periodic model

$$k \leq r$$

Summary

- Kernel-groupcast
  - Improve the efficiency of group communication
- Analytic framework
  - Consider two server models
  - Predict performance impact of an improved group communication mechanism
  - Derive the upper bounds on capacity improvement

Future Work

- Study the behavior of other group communication applications
- Extend our analytic framework to consider stochastic workload and processing requirement of different operations
- Apply kernel-groupcast to other transport protocols
- Design improved system call interface

Thank you

University of Waterloo