# Low-Latency Communication over ATM Networks using Active Messages

Hot Interconnects II

Aug 11th, 1994

Thorsten von Eicken
Department of Computer Science
Cornell University

Email: tve@cs.cornell.edu

Papers & software: http://www.cs.cornell.edu/Info/People/tve/tve.html

# ATM Clusters of Workstations: The Potential

**Raw performance**

- cpu: 100 SPECint, 100 SPECfp
- network: 15MB/s switched net
- software: standard OS & env.

Idea:

- leverage off-the-shelf hardware & standard software environment
- performance of multiple workstations

Key enabler:

- program cluster as if one machine

**Potential use: Run one big server application on cluster**

e.g: maintain data repository and provide compute power to process/visualize data

**Potential applications**

- "I/O computer"
  - maintain large data repository
  - provide compute power to manipulate data sets
  - act as server to numerous clients
- Compute server

*Disclaimer:* the goal is NOT to make a cluster be a cheaper/better MPP

# BUT: The Reality is...

**Communication performance**

| | MPP | WS cluster |
|---|---|---|
| bandwidth | 10MB/sec | 1MB/sec |
| round-trip | 10µsec | 5ms |
| message ovhd | 1µs | 1ms |

**Prevents use of parallel programming techniques**

cannot implement global address space with reasonable remote read/write
cannot coordinate execution of processes

**Cannot view cluster as one machine**

separate programs on all nodes
loosely coupled, only “embarrassingly parallel” algorithms

# Is it possible to do better?

**Raw performance**

- cpu: 100 SPECint, 100 SPECfp
  >50MB/s I/O bus
- network: 15MB/s switched net
  48-byte cell = 3.2µs latency

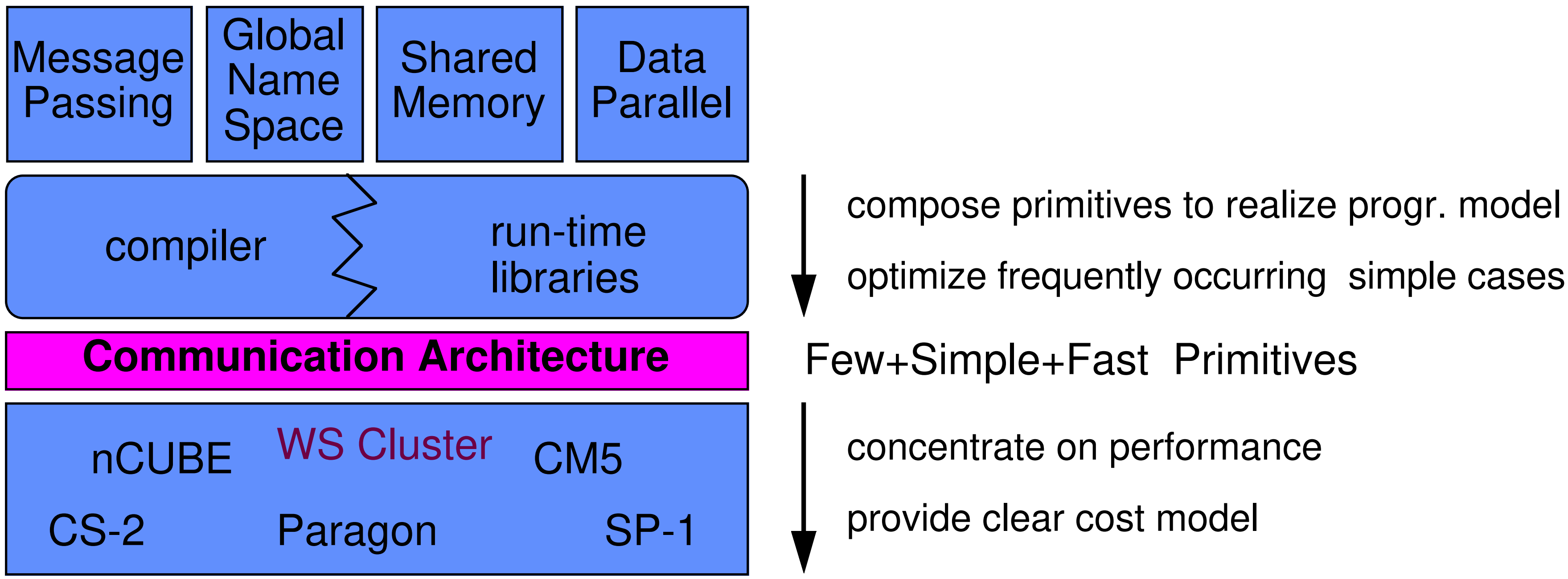
**Expectation:** close to MPP performance
but a small difference will remain

**Trick:** cut out the layers of software
(without compromising security)

**Off-the-shelf handicaps**

- workstation ➥ network interface is on I/O bus
  ➥ no protected user-level access to NI
- operating system ➥ no coordination among nodes
- ATM network ➥ unreliable communication and no flow-control

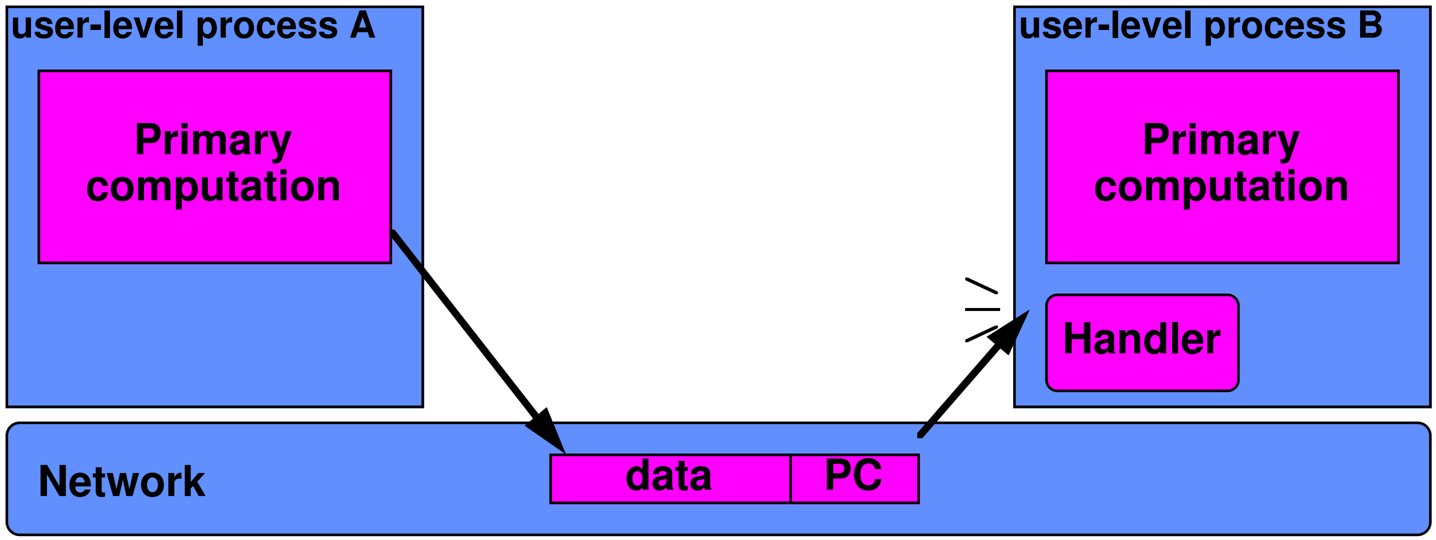
# Active Messages: an MPP Communication Architecture




Communication architecture functionality

- check validity of messages sent
- demultiplex arriving messages
- provide access to message data
- notify destination processes on message arrival

# Active Messages




Sender injects the message as directly into the network as possible

- Active Messages checks validity of destination

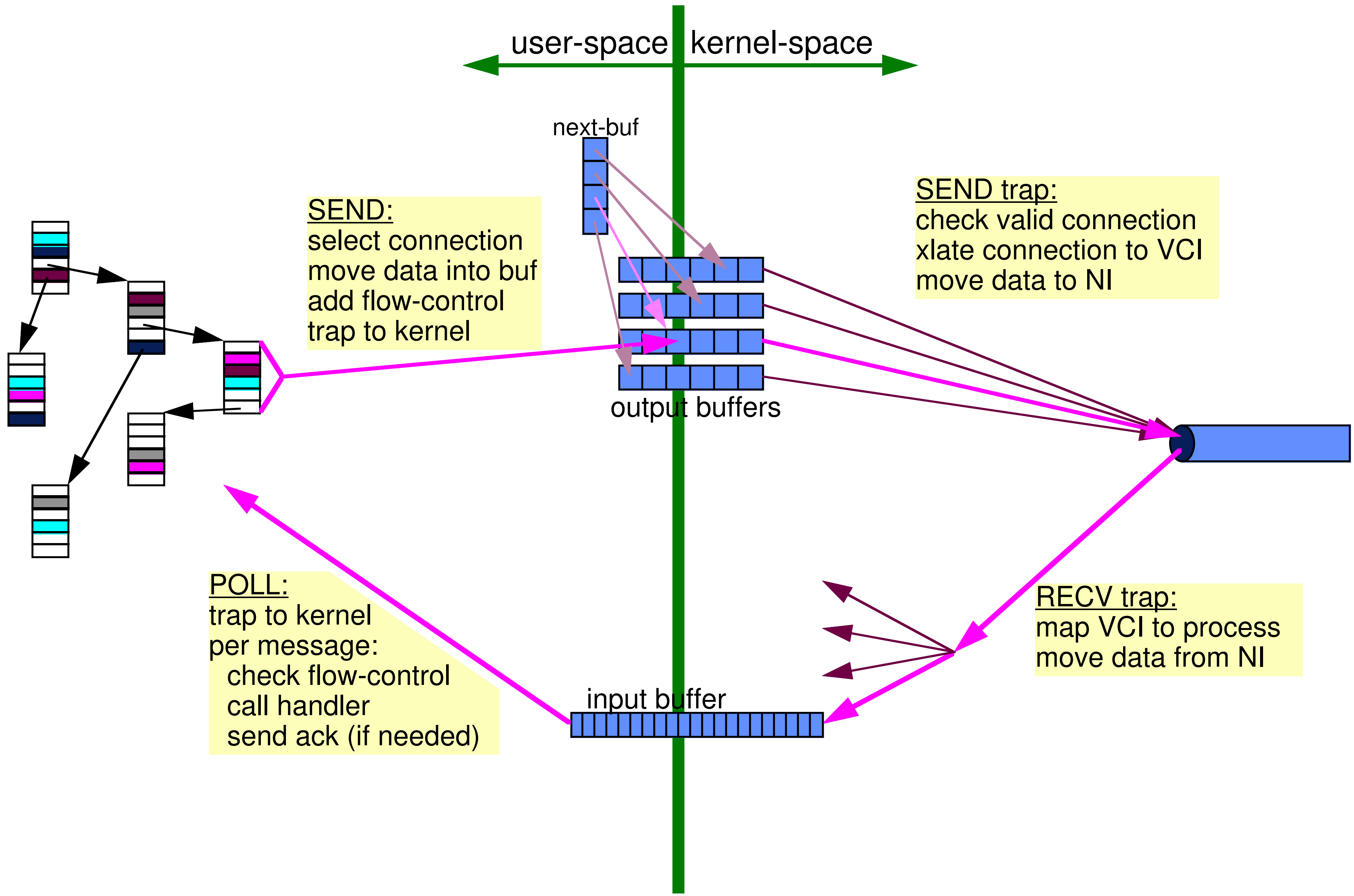
Associates a small user-level handler with each message

Handler executes immediately upon arrival

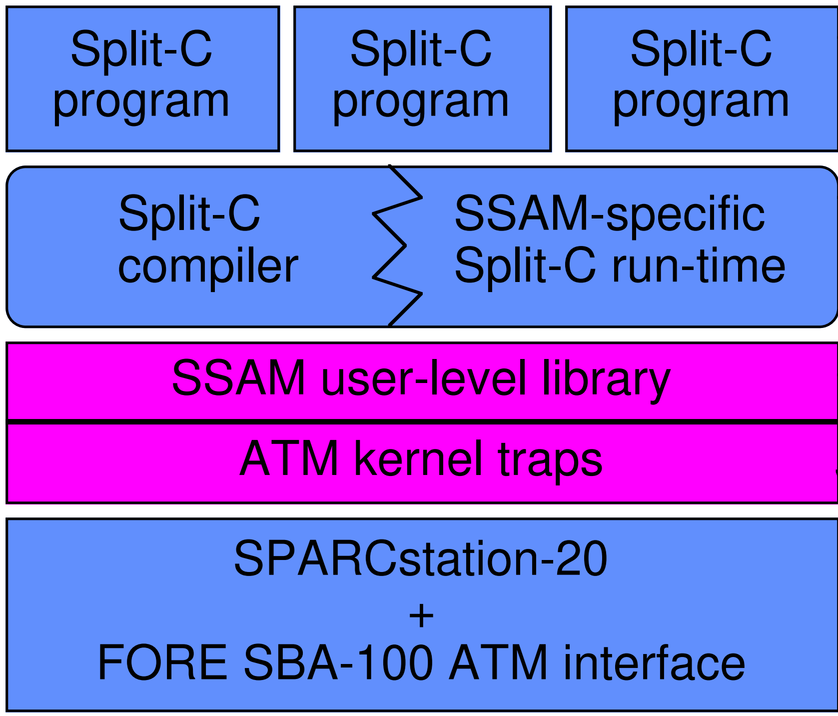
- pulls the message out of the network, and
- integrates it into the ongoing computation
- or performs simple remote service and replies

No buffering (beyond transport), no parsing, no allocation, primitive scheduling

# SSAM: SparcStation Active Messages Prototype



# SSAM software layers




Software layers

Split-C programs

- use global pointers and pointer dereferencing to access remote memory

language run-time system

- implements communication model, e.g. send&receive, global memory access, ...

user-level library

- formats messages, implements Active Messages
- performs flow-control and error-checking

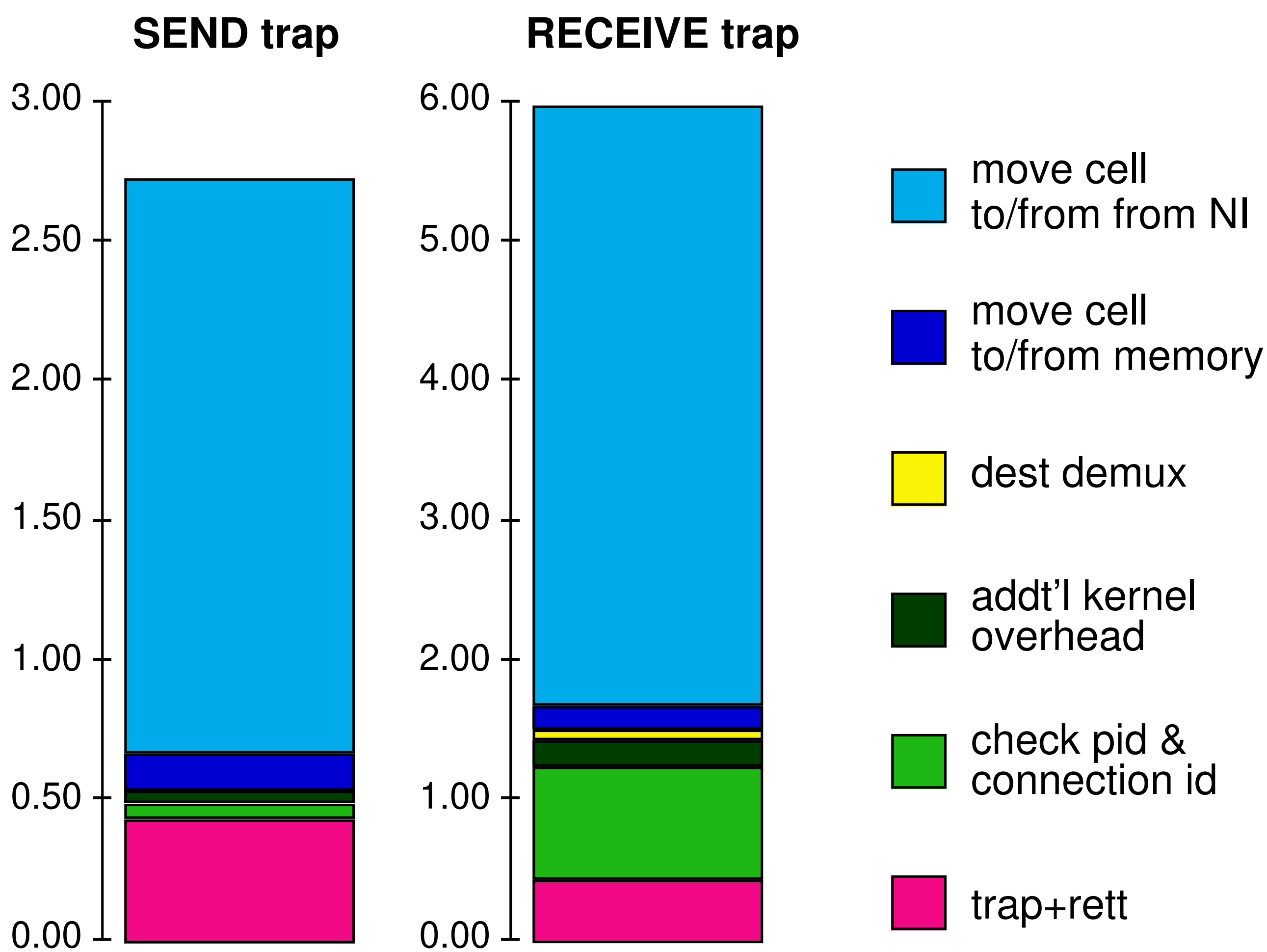
kernel traps

- move data between user-space and network interface
- provide protection and demultiplexing

# SSAM Trap Cost

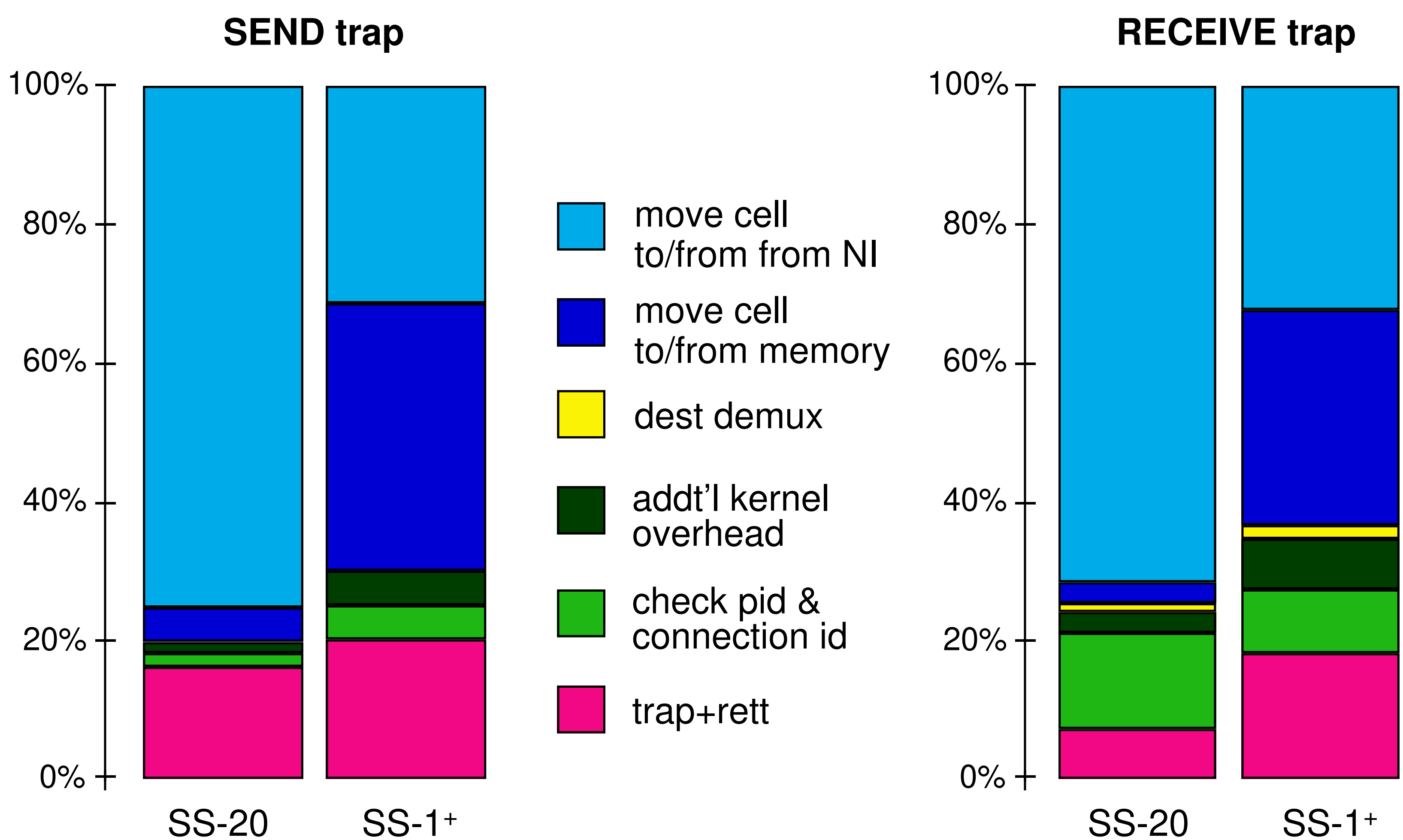
On SPARCstation-20 (60Mhz)




*Note: null system call: 6.9µs on SS-20*

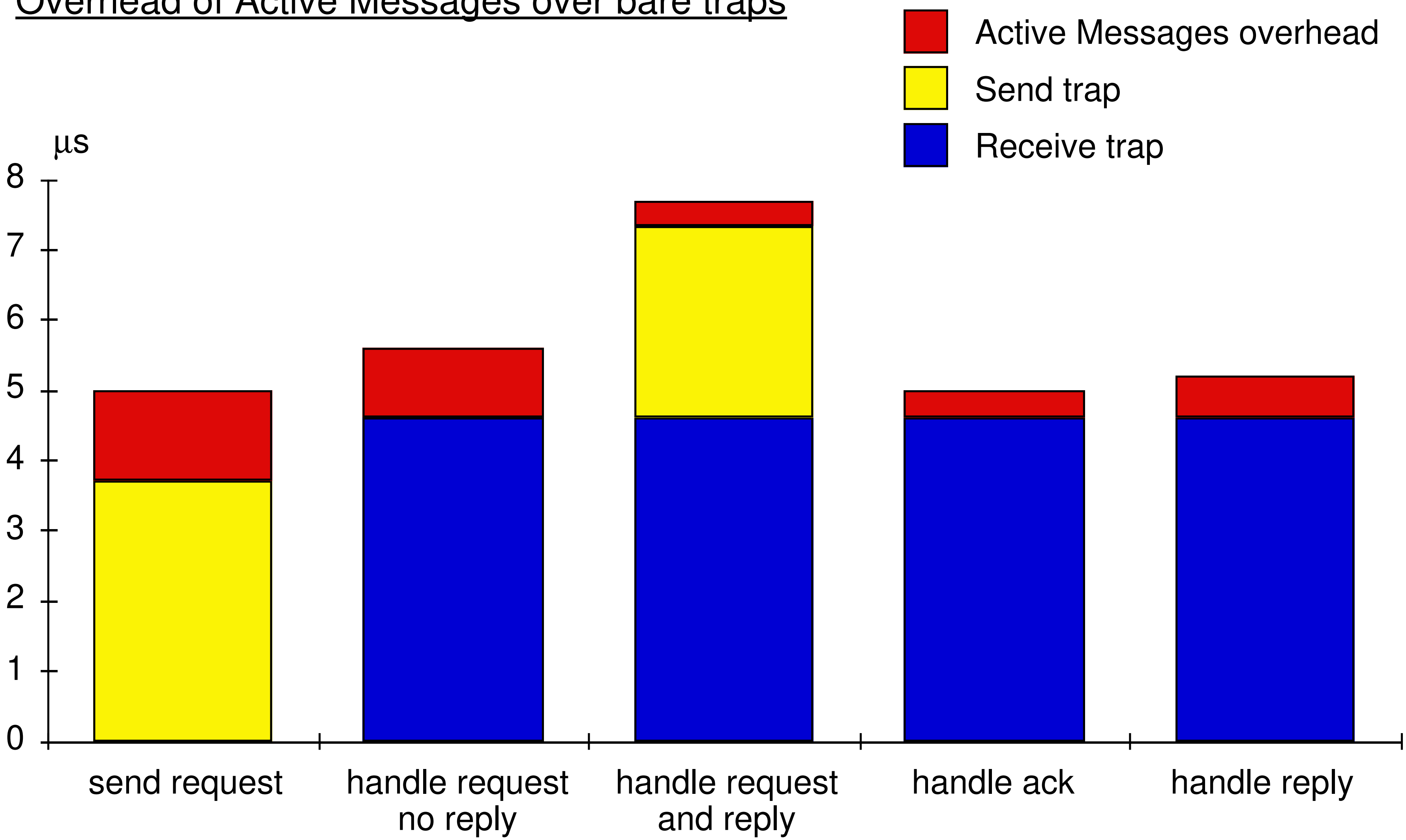
# SSAM Trap Cost (cont.)

Trap cost comparison: SS-20 (60Mhz, superscalar) vs.
SS-1+ (25Mhz, single issue)



# SSAM Active Messages Cost

Overhead of Active Messages over bare traps

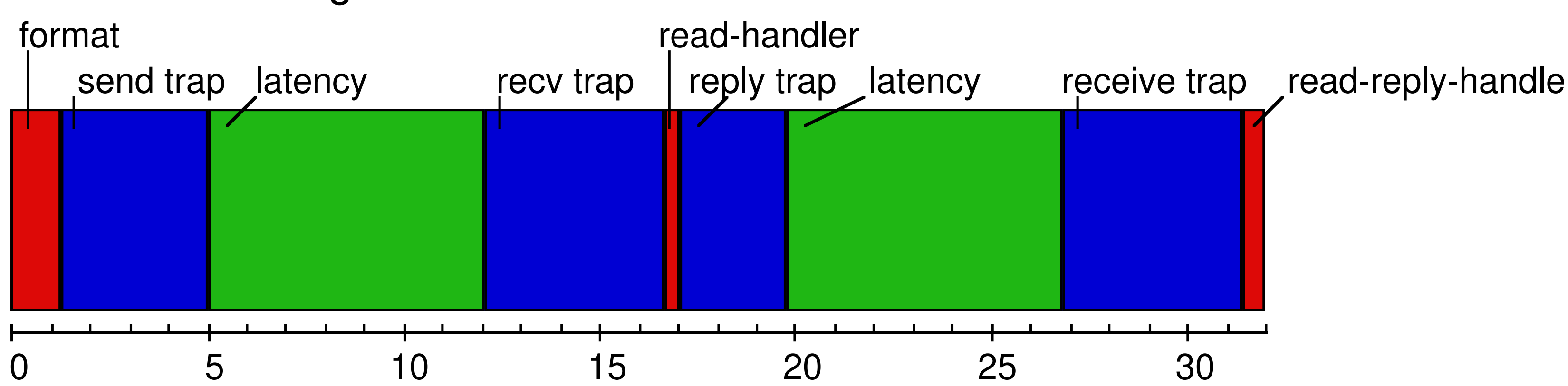


# Split-C performance

## Split-C remote read

- global pointers
  - ```
    struct node { ... } N;
    double *global P = toglobal(&N);
    ```
- remote read
  - ```
    struct node my_N;
    my_N = *P;
    ```
- implementation
  - send request Active Message to remote read-handler
  - read-handler sends reply Active Message with data to read-reply-handler
  - read-reply-handler stores data local and signals completion

## Remote read timing



# Comparison to Related Work

## MPP performance

- SPARCstation-20 cluster vs CM-5
  - 2/3 the bandwidth
  - 4x the round-trip latency (incl. one switch)
- bottlenecks
  - network interface on the I/O bus
  - slow network switches (10µs/hop)
  - no protection, no flow-control, no error detection in hardware

| machine | peak bandwidth | round-trip latency† |
|---|---|---|
| SP-1 + MPL/p | 8.3MB/s | 56µs |
| Paragon + NX | 73MB/s | 44µs |
| CM-5 + CMAML | 10MB/s | 12µs |
| **SS-20 + SSAM** | **5.6MB/s** | **42µs** |

† incl 1 switch

## Univ. Washington Remote Memory Access

- *C.A. Thekkath, et.al, UW TR93-04-03*
- set-up: SS-20 vs. DS-5000, same NI, same back-to-back fiber, same cell payload, similar implementation (traps)
- basically same data transfer performance
- SSAM provides more flexibility, includes notification, guarantees delivery, but requires process scheduling coordination
- Remote Memory Access does not require remote process to run, but requires memory mapping coordination (actually, special memory segments are used)

| operation | SSAM | remote mem acc |
|---|---|---|
| read latency | 32µs | 45µs |
| write latency | 22µs | 30µs |
| addt'l control transfer ovhd | none | 260µs |
| block write | 5.6MB/s | 4.4µs |

† no switch included

# Summary & Conclusions

*Papers&software:* `http://www.cs.cornell.edu/Info/People/tve/tve.html`

## ATM workstation clusters show promise

- off-the-shelf hardware works
- raw performance numbers look competitive with MPPs
- standard networking software looses most of the potential
- **but:** MPP-style communication architecture delivers expected performance

## SPARCstation Active Messages design

- must start from raw hardware and build-up
- must use direct traps to kernel to achieve protection at minimal cost
- must carefully orchestrate buffering to minimize data movement
- **result:** competitive with MPPs,
  difference to fastest MPPs can be explained by hardware limitations

## Bottlenecks

- network interface on the I/O bus
- slow network switches (10μs/hop)
- no protection, no flow-control, no error detection in hardware

## Open questions

- operating system coordination among nodes
- flow-control for parallel computing communication patterns (highly correlated)