Ninth IEEE International Symposium on High Performance Distributed Computing, Pittsburgh, Pennsylvania, August 1-4, 2000

Speculative Defragmentation – A Technique to Improve the Communication Software Efficiency for Gigabit Ethernet

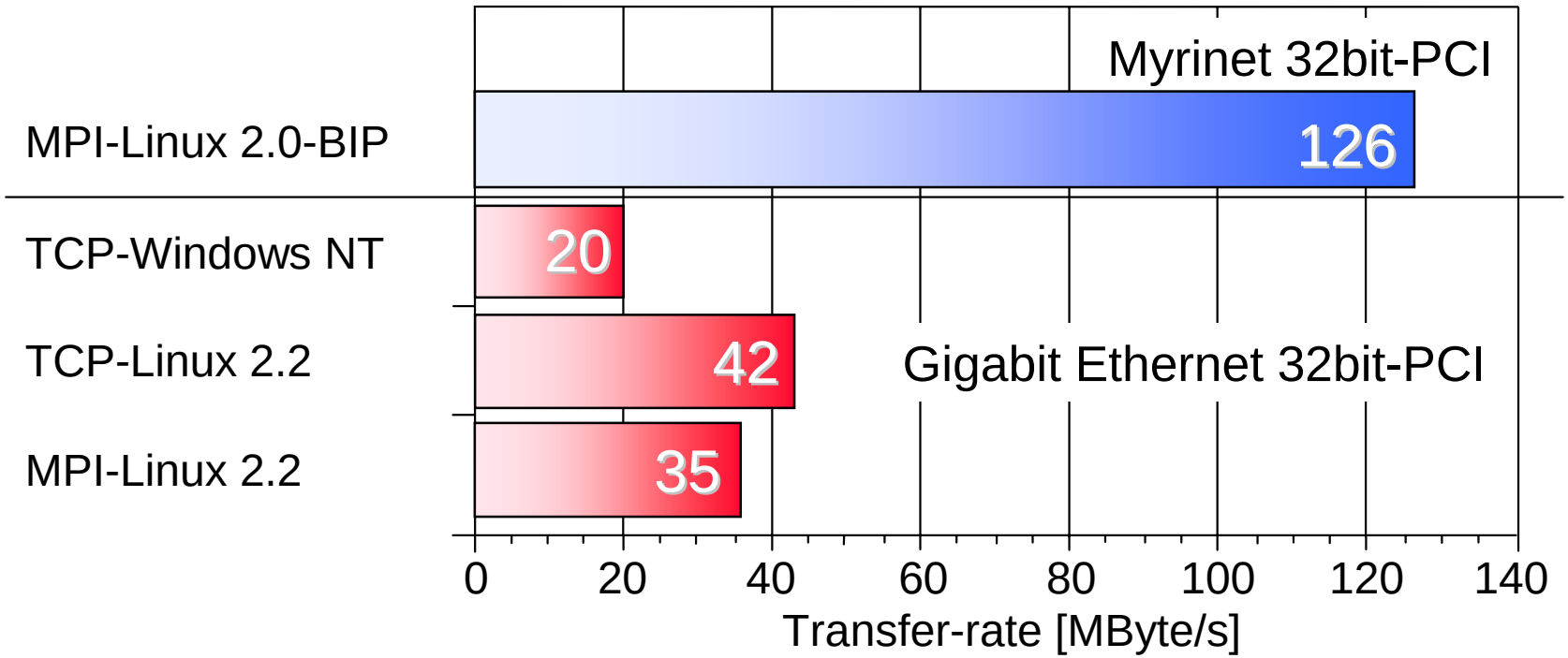
Ch. Kurmann, F. Rauch, M. Müller, T. Stricker

Laboratory for Computer Systems
ETHZ - Swiss Institute of Technology
CH-8092 Zurich

Eidgenössische Technische Hochschule Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Comm. Speeds of Commodity PCs



⇒ For Gigabit Ethernet and TCP/IP the OS-software cannot keep up with the hardware speed

Overview

- Why Gigabit Ethernet
  - Packet Defragmentation
    - TCP/IP Overheads
      - Speculative Packet Defragmentation
        - Performance Analysis
          - Conclusion

Problem Statement

How can we sustain network bandwidths of
75-100 MByte/s with a commodity PC cluster node:

- memory copy 90 MByte/s
- 32-bit PCI I/O-bus 132 MByte/s
- commodity Gigabit Ethernet adapter 100 MByte/s

- standard TCP/IP protocol
- fully transparent standard socket-API

Papers 10 Years Ago

The same problem — 30-100 times slower

- memory copy < 3 MByte/s
- VME I/O-bus < 3 MByte/s
- commodity 10BaseT Ethernet adapter 1 MByte/s

- special purpose blast transfer protocol [Zwaenepoel85]
- optimistic bulk transfers [Carter89]
- transparent blasts by header padding [Peterson90]

Not standard protocol & not fully transparent

⇒ Solutions did not find their way into current systems!

Why Gigabit Ethernet

- Compatible to Ethernet and Fast Ethernet (UTP Cat5)
- Uncomplicated technology which results in high reliability and low cost
- Switched Ethernet provides link level flow control on full duplex channels
- In larger networks only unacknowledged, connectionless datagram delivery service $\Rightarrow$ TCP needed
- Standard frame size is still limited to 46-1500 Byte of data

Alternatives / Extensions

- Dedicated network hardware with customized lightweight protocols: Myrinet, SCI, Giganet, ServerNet

  ⇒ primarily designed for internal communication in server farms

- Jumbo Frames (9 KByte) for Gigabit Ethernet to reach a Maximal Transfer Unit (MTU) of a memory page:

  ⇒
  - change of standard
  - higher latencies in store and forward switches
  - do not solve the header/payload separation

Packet [De]Fragmentation

- IP standard technique
- Data to be sent is fragmented into small chunks < network MTU (Maximal Transfer Unit)
- Network protocols enclose the frames with header/trailer
- Receiver separates the headers from the payload and defragments the data again
- Implications for Ethernet:
  - MTU < Memory Page
  - DMA-logic not optimal

$\Rightarrow$ Therefore memory copy for packet [de]fragmentation

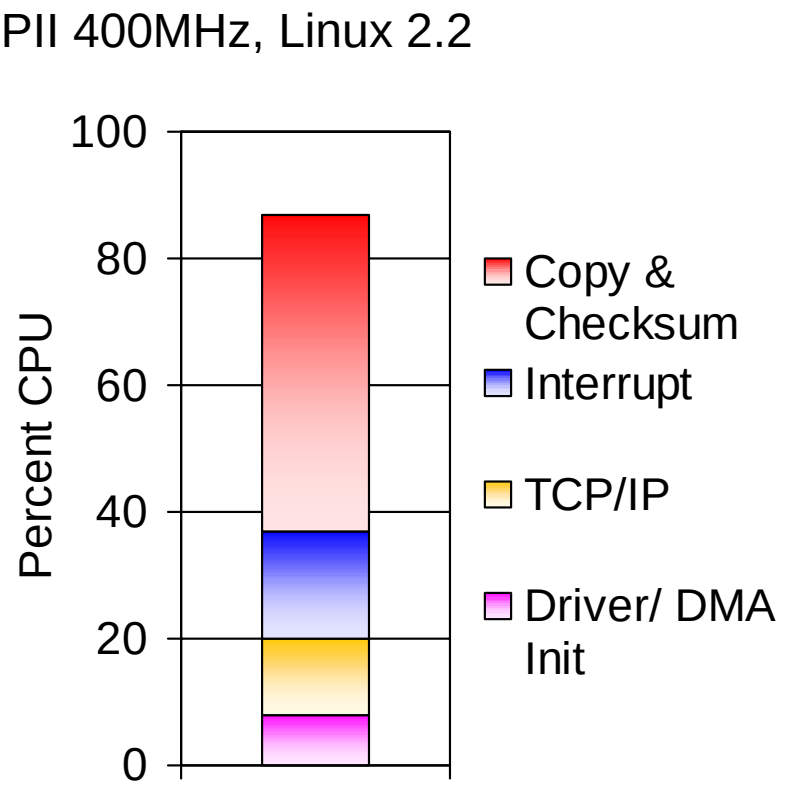
TCP/IP Host Overheads

- Single largest overhead: copying and checksums

  ⇒ Zero-copy techniques

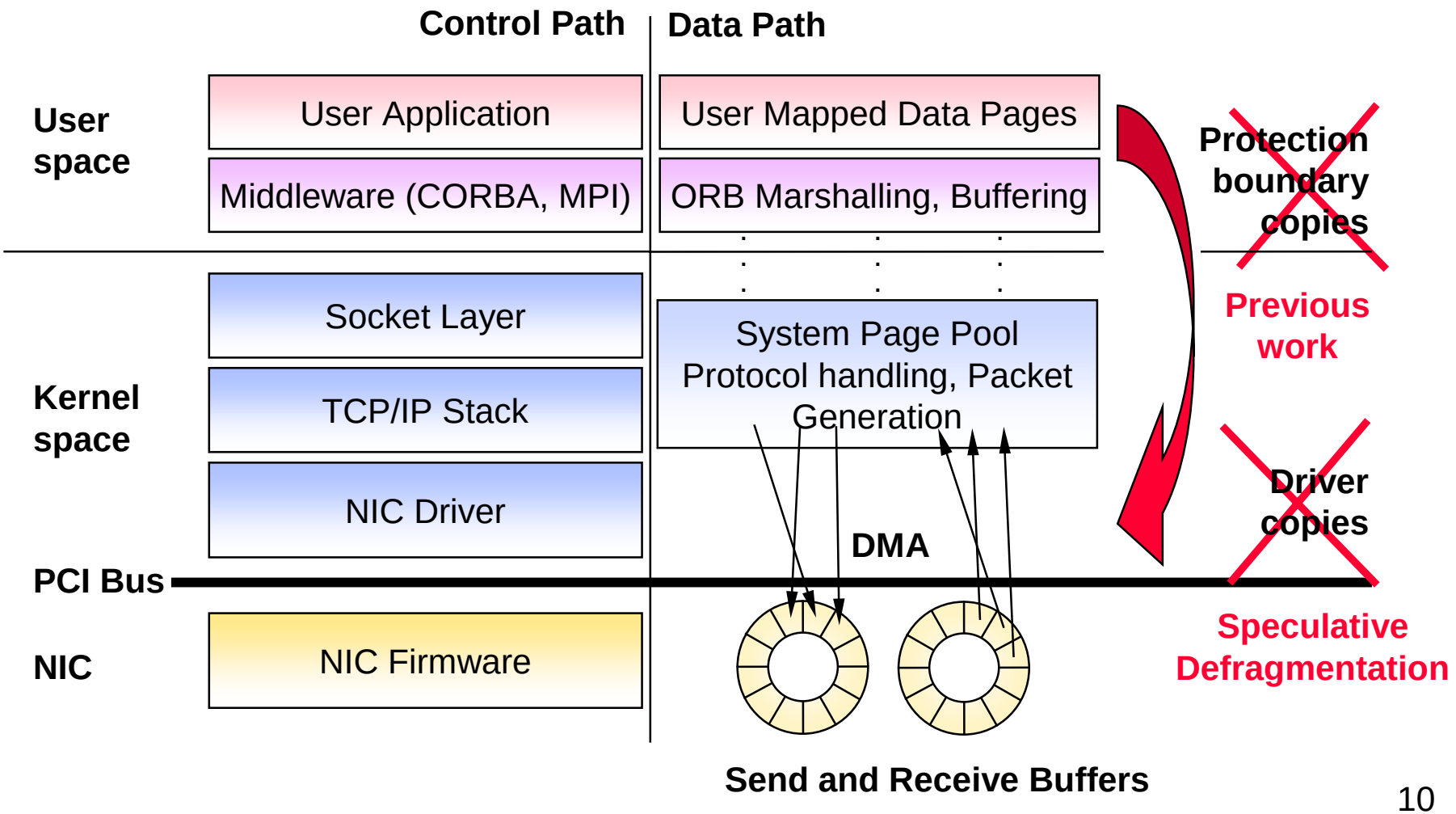
- Per-packet processing and interrupt overhead also high

  ⇒ Interrupt coalescing



Host Overhead for TCP/IP over Gigabit Ethernet

OS Environment



Required Technologies

- Well known solutions to eliminate the User/Kernel copy:
  - User-Level Network Interface (U-Net) or Virtual Interface Architecture (VIA)
  - User/Kernel Shared Memory (FBufs, IOLite), Copy Emulation or Page Remapping with Copy on Write
- The Driver copy remains for Gigabit Ethernet

⇒ Goal: Elimination of driver copy for the packet defragmentation and header separation

⇒ True zero-copy

Commodity GE-Adapters

- Until now, zero-copy support is only available for “intelligent” network adapters (ATM, SiliconTCP)
- Today’s Gigabit Ethernet adapters are too simple
  - no processor, TLBs on board
  - limited DMA capabilities
  - no protocol filtering implemented

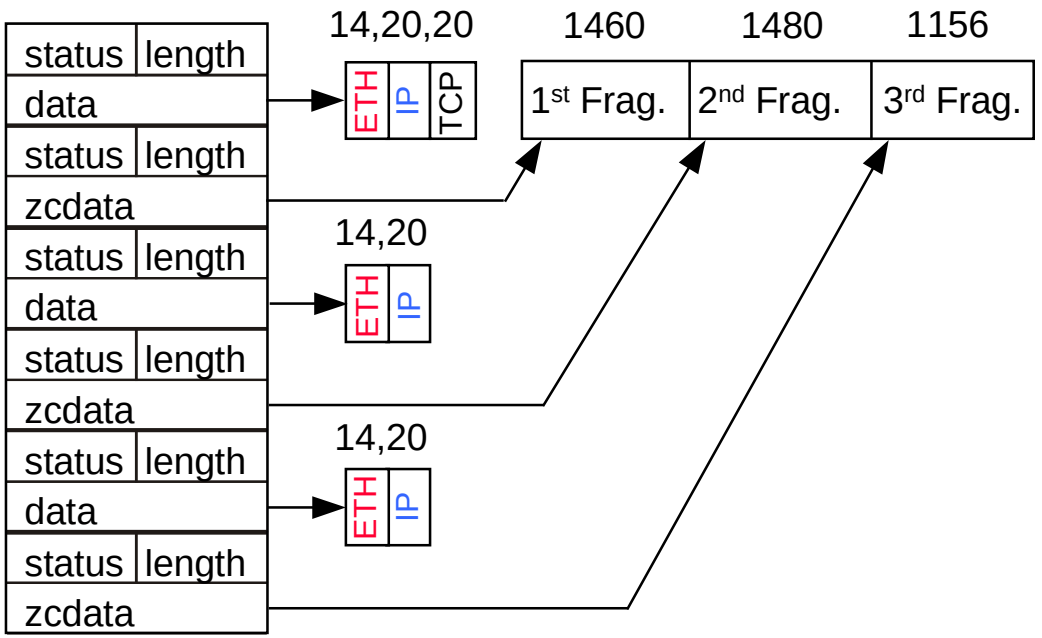
  ⇒ Deterministic zero-copy implementation with commodity GE adapters is not possible!

- Approach: Making just the common case fast

  ⇒ Speculation Techniques for Defragmentation

Speculative Defragmentation I

- Our driver manages to send/receive entire 4 KByte pages
- Decomposition of 4 KByte IP-packets into 3 IP-fragments on driver level (standard IP fragmentation)
- Attachment of headers to the payload data with a separate DMA-descriptor



Speculative Defragmentation II

What are we speculating about?

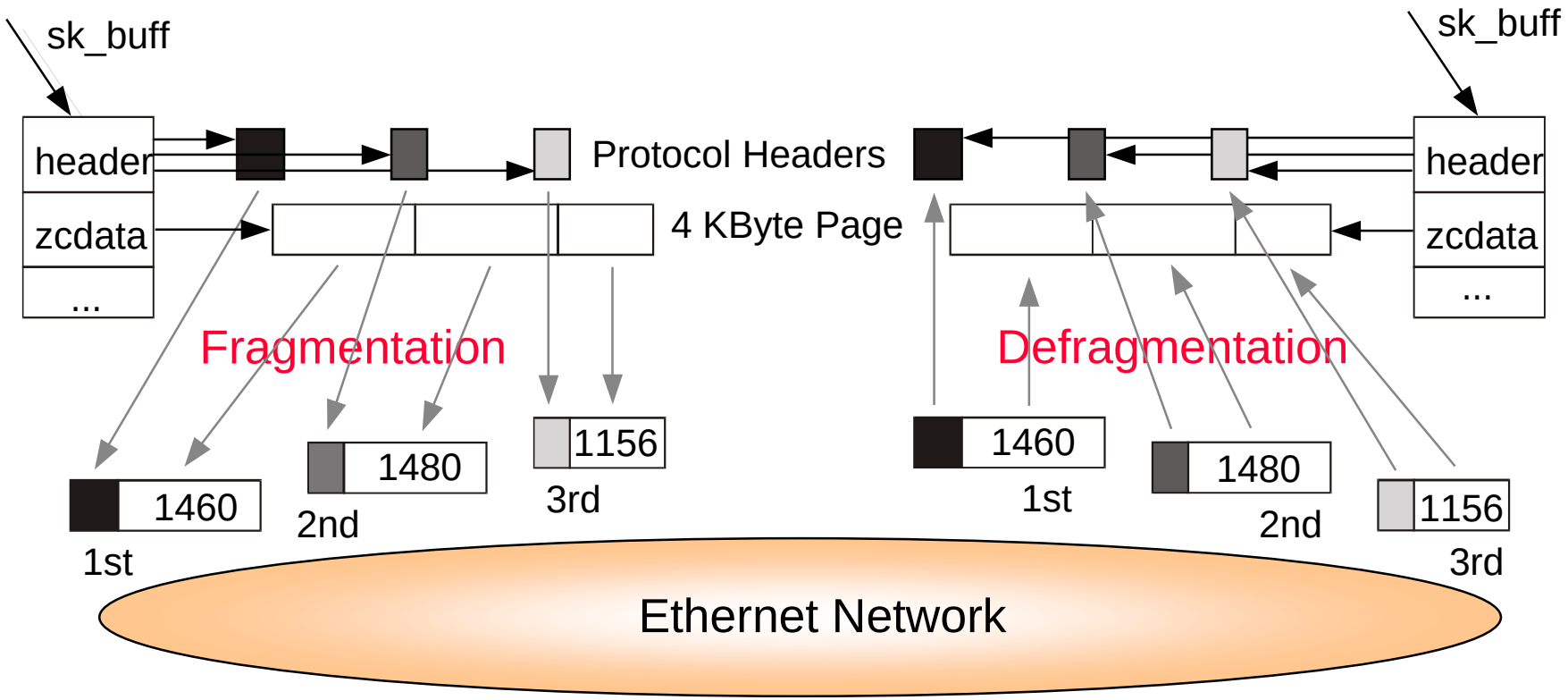
- Speculation that all fragments of a whole page will be received in order
- Speculation about the precise packet format (header-lengths, data-fields)

- The receiver has to fix the DMA descriptors without knowledge about the next packets to arrive
- In clusters with one or two switches, the probability is high, that the three fragments arrive in order

- Software cleanup when mis-speculation

Speculative Defragmentation III

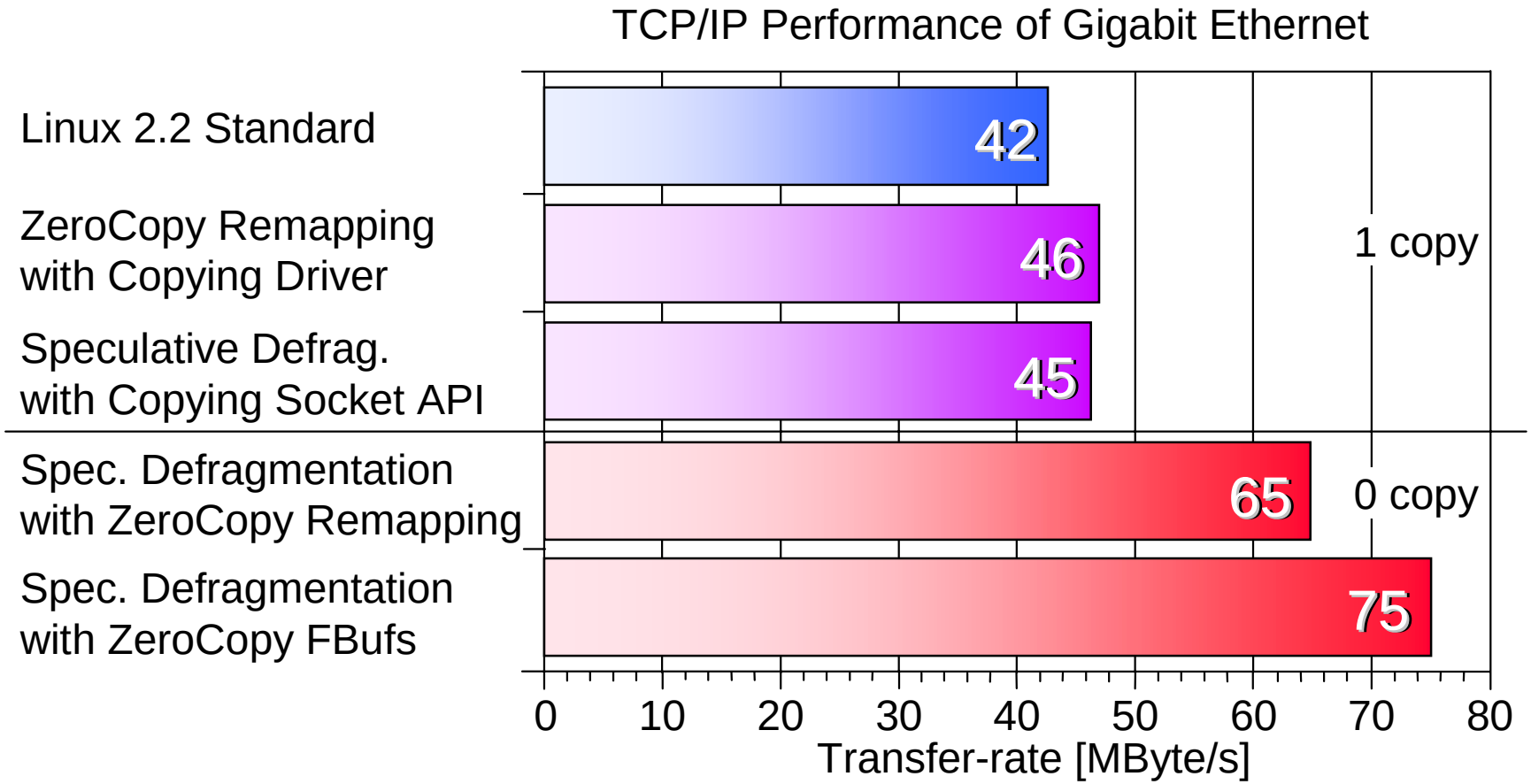
Fragmentation/Defragmentation of a 4 KByte memory page by the DMA of the network interface



Performance Evaluation

- Gains by Successful Speculation
- Penalty for Speculation Misses
- Speculation Success Rates in Applications
- Consequences:
  - Network Control Architecture
  - Suggested Hardware Improvements

Gains with Speculation



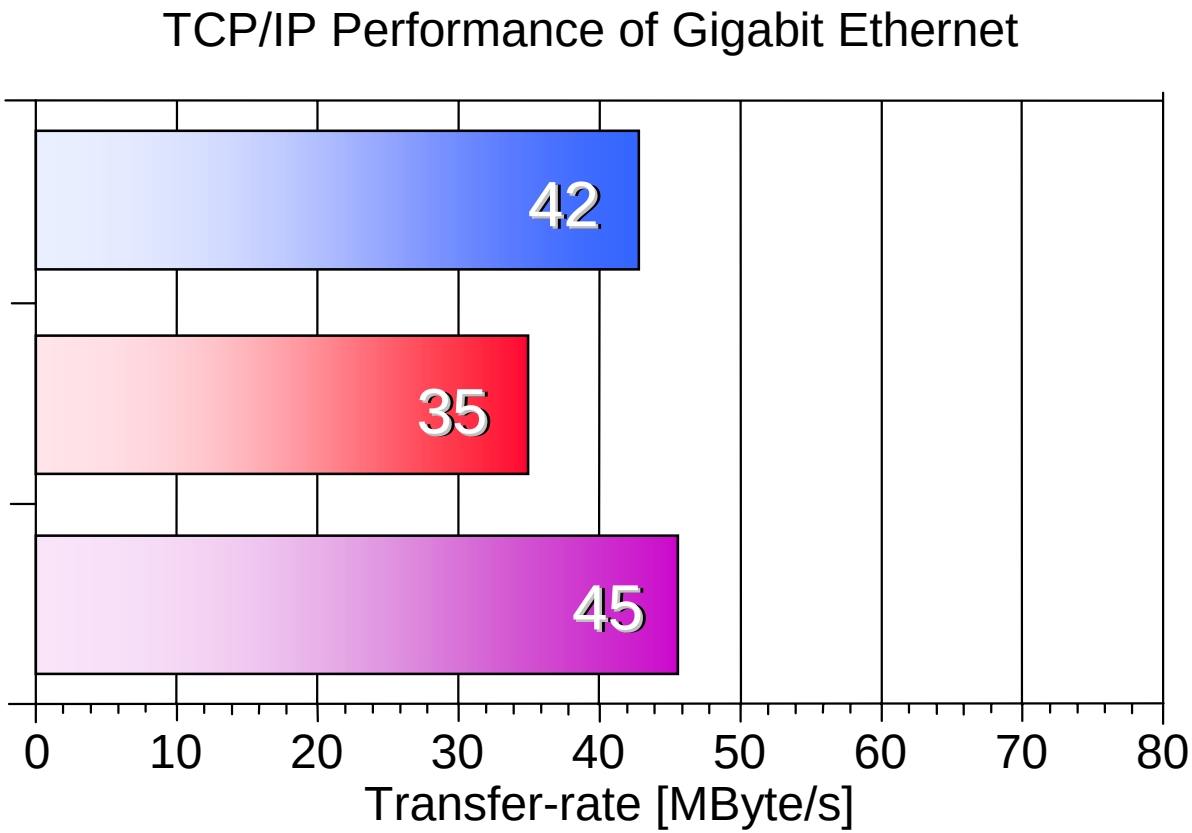
$\Rightarrow$ 80 % increase in performance (bandwidth)

Penalty with Speculation Misses

Linux 2.2 Operation
Standard Sender
Standard Receiver

Fallback
Standard Sender
Zero-Copy Receiver

Compatibility
Zero-Copy Sender
Standard Receiver



⇒ The common case is fast, the fallback not much slower

Evaluation of Success Rates

- Application traces show success of speculative transfers

Traces		Oracle TPC-D			TreadMarks SOR		
		Master	Host1	Host2	Master	Host1	Host2
Ethernet frames	total	129835	67524	62311	68182	51095	50731
	large	90725	45877	44848	44010	30707	30419
	zcopy	79515	37833	41682	44004	30693	30405
	ok	38235	37833	41682	44004	30675	30399
Success Rate		48 %	100 %	100 %	100 %	> 99 %	> 99 %

- TreadMarks has an inherent scheduling that prevents interference
- TPC-D needs a control architecture or hardware changes

Network Control Architecture

- Problem: Multiple synchronous, fast receives may garble the zero-copy frames

  ⇓

- Solution: Admission Control on Ethernet driver level with negotiation for one single sender to blast
- Implicit channel allocation by OS works
- Fully transparent
- No explicit scheduling of transfers through a special interface ⇒ the API remains the same

Suggested Hardware Improvements

- Additional control-path between the checksumming- and the DMA-logic for detection of protocol & header fields

  ⇒ Reliable header/payload separation

- Stream detection with a simple matching register and a separate DMA descriptor chain for fast transfers:

  ⇒ Detection of at least one high performance stream

  ⇒ Separation of this stream with its DMA descriptors

  ⇒ Improvement of the speculation rate

  Lower driver complexity

Conclusion

- Existing Ethernet protocols and standard network interface chipsets prevent an accurate, fully deterministic defragmentation in hardware.
- Speculation techniques open a new horizon for optimized network drivers and permit an “almost”-zero-copy implementation for TCP/IP over Gigabit Ethernet.
- The performance in our implementation was raised from 42 to 75 MByte/s (80%) using the standard Linux TCP-stack and commodity network interface hardware.
- Speculation works in network interfaces as well as in “Instruction Level Parallelism” and should be considered to find simple and effective hardware improvements for network interfaces.