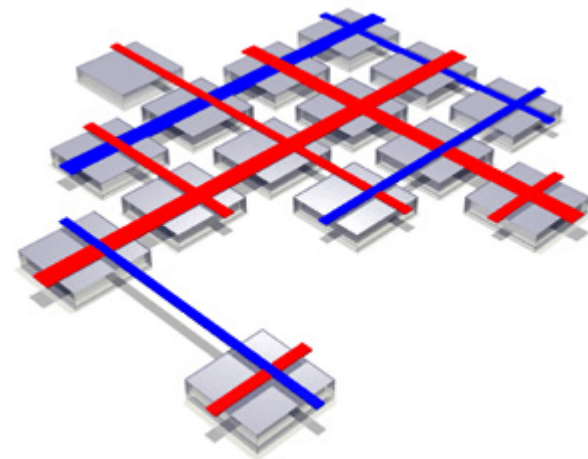
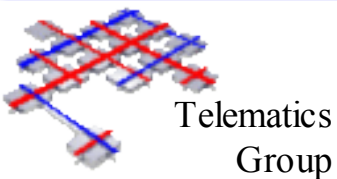


# Performance Optimization in Wireless Networks

Ingo Juchem  
ijuchem@cs.uni-goettingen.de

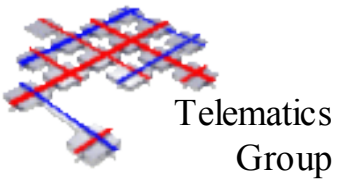
Telematics Group,  
Institute for Informatics,  
University of Goettingen





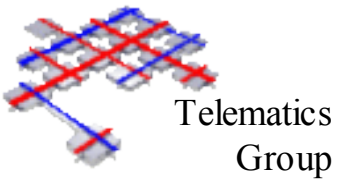
# Content

- Introduction
- TCP Cross-Layer optimizations in MANETs
- GPRS TCP Wireless WAN optimizations
- Summary



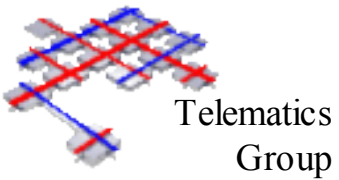
## Introduction (I)

- What's the situation for TCP ?
  - Mobile networks – mobile nodes
  - TCP not designed for mobile nodes (assumes congestion → slow start mechanism etc.)
- TCP performance degrades
  - **Route failures** due to mobility
  - **Packet loss** due to **link** and **route failures**



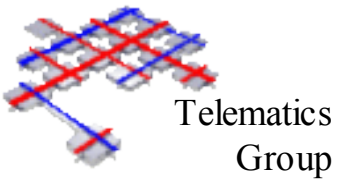
## Introduction (II)

- Approaches to solve the issue(s):
  - Provide feedback to TCP about link failure  
→ ELFN (Explicit Link Failure Notification)
- BUT:
  - Route failures still present
    - Reason: stale (outdated) routes in cache
- SO:
  - New cache update algorithms needed (DSR)



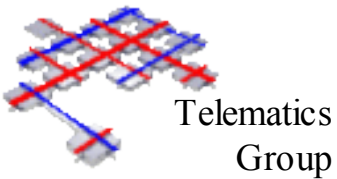
## Introduction (III)

- **Also:** Packets are dropped/lost after link failure
  
- Two main issues:
  1. Find solution for route failures and packet loss
  2. Make TCP more efficient by approaches in network layer and cross-layer



## Introduction (IV)

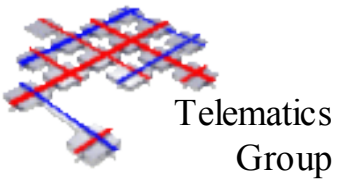
- Idea(s):
  - Make routing protocols aware of lost data and ACK packets
  - Reduce TCP timeouts caused by mobility
    - Two NEW mechanisms
      1. EPLN (Early packet loss notification)
      2. BEAD (Best-effort ACK delivery)



## Routing protocols optimization

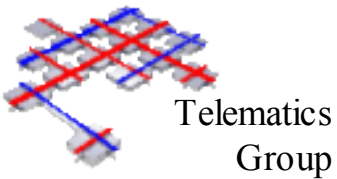
What's next / How to do it ?

- Route failures:
  - Extensive use of cached routes:
    - DSR
    - Distributed Cache Update Algorithm
- Link failures and packet loss:
  - ELFN
  - EPLN
  - BEAD



## Cached routing - DSR

- **PROBLEM:**  
More mobility → more cached routes → more stale routes !
- Need new approach to remove stale routes
- Idea: Dynamic Source Routing (DSR)
  - Two mechanisms
    - Route Discovery (REQUEST & REPLY)
    - Route maintenance  
→ used to update stale routes in cache



## DSR – Route maintenance

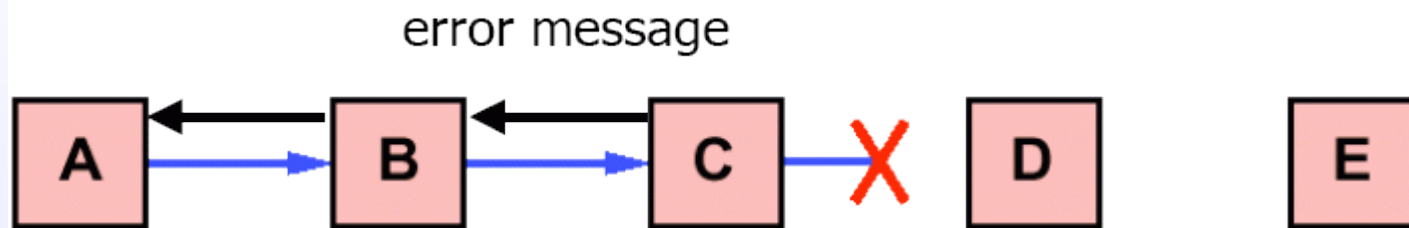
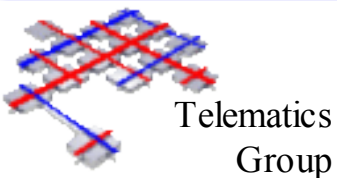


Fig.1 Route maintenace in DSR [1]

- C waits for ACK from D, doesn't receive it
- Sends RouteError to A
- A deletes route from cache
  - Another route in cache → send through there
  - Else → re-initiate Route Discovery
- But quite time cosuming



## Distributed Cache Update Algorithm (I)

- Proactive – inform other nodes immediately
- For each link, node knows neighbors having the same link in their cache

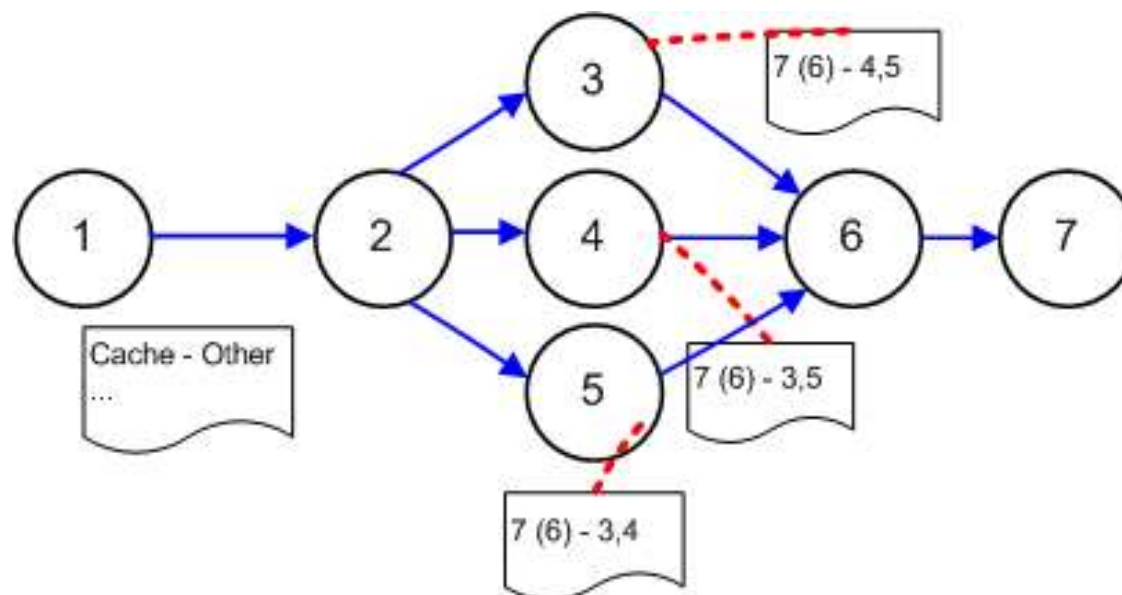
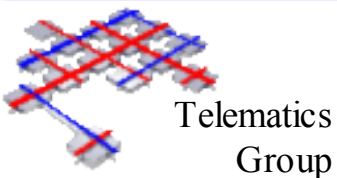


Fig.2 Distributed Cache Update Algorithm

- Activated by link failure OR update notification



## Distributed Cache Update Algorithm (II)

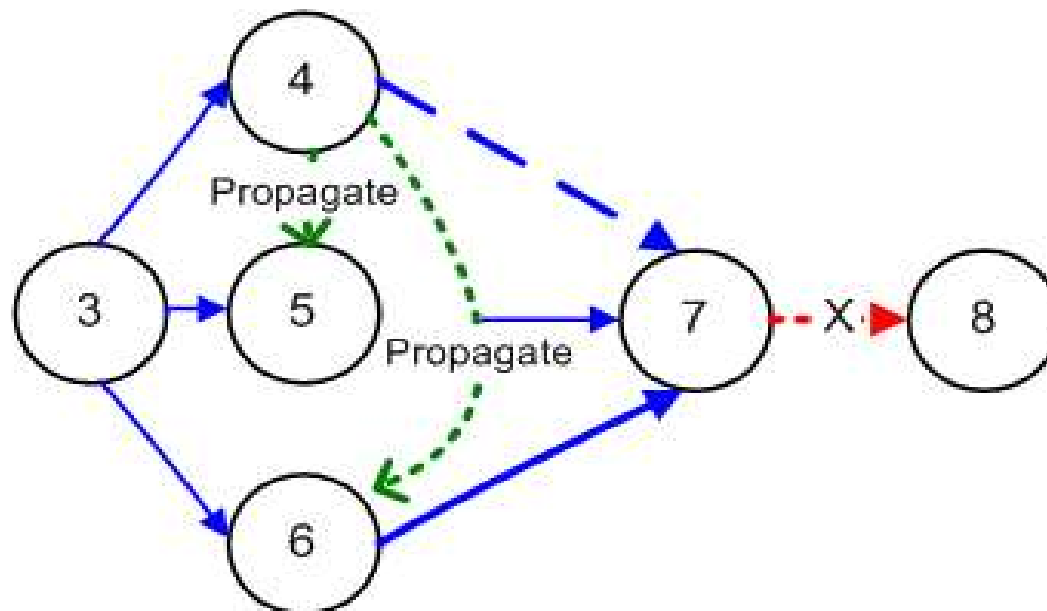
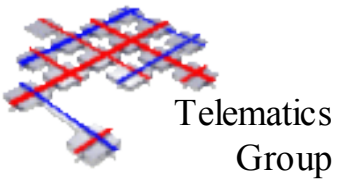


Fig.3 Distributed Cache Update Algorithm II

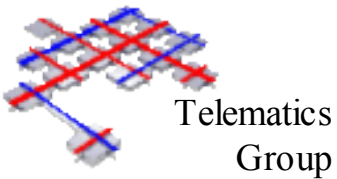
- → Three benefits:
  - Reduced packet loss
  - Reduced packet delivery latency
  - Reduced Route Errors



## Problem solved, next one please

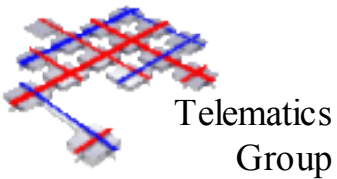
- Route failure problem seems to be solved
- „Sophisticated Cache Update Solutions improve behavior in case of route failures“

But, what about link failures?



## ELFN (Explicit Link Failure Notification)

- ELFN provides link failure feedback to TCP
  - Approach: Freezing of TCP timer
    - Node detects link failure →
    - notify TCP sender →
    - TCP freezes timers and state →
    - Sends probing packets for ACKs through alternate route if present →
    - Continue as normal
- Performance gains for TCP in Mobile Ad Hoc

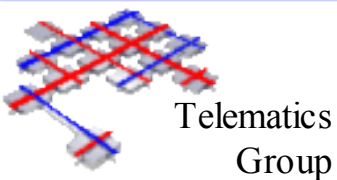


## What about lost packets and ACKs?

- ELFN doesn't indicate if packets or ACKs are LOST or just a link failure occurred
- Routing protocols silently DROP packets and ACKs in queue
- → TCP times out waiting for ACKs
- → TCP retransmits unACK'ed packages

Approach: EPLN and BEAD





## EPLN Example

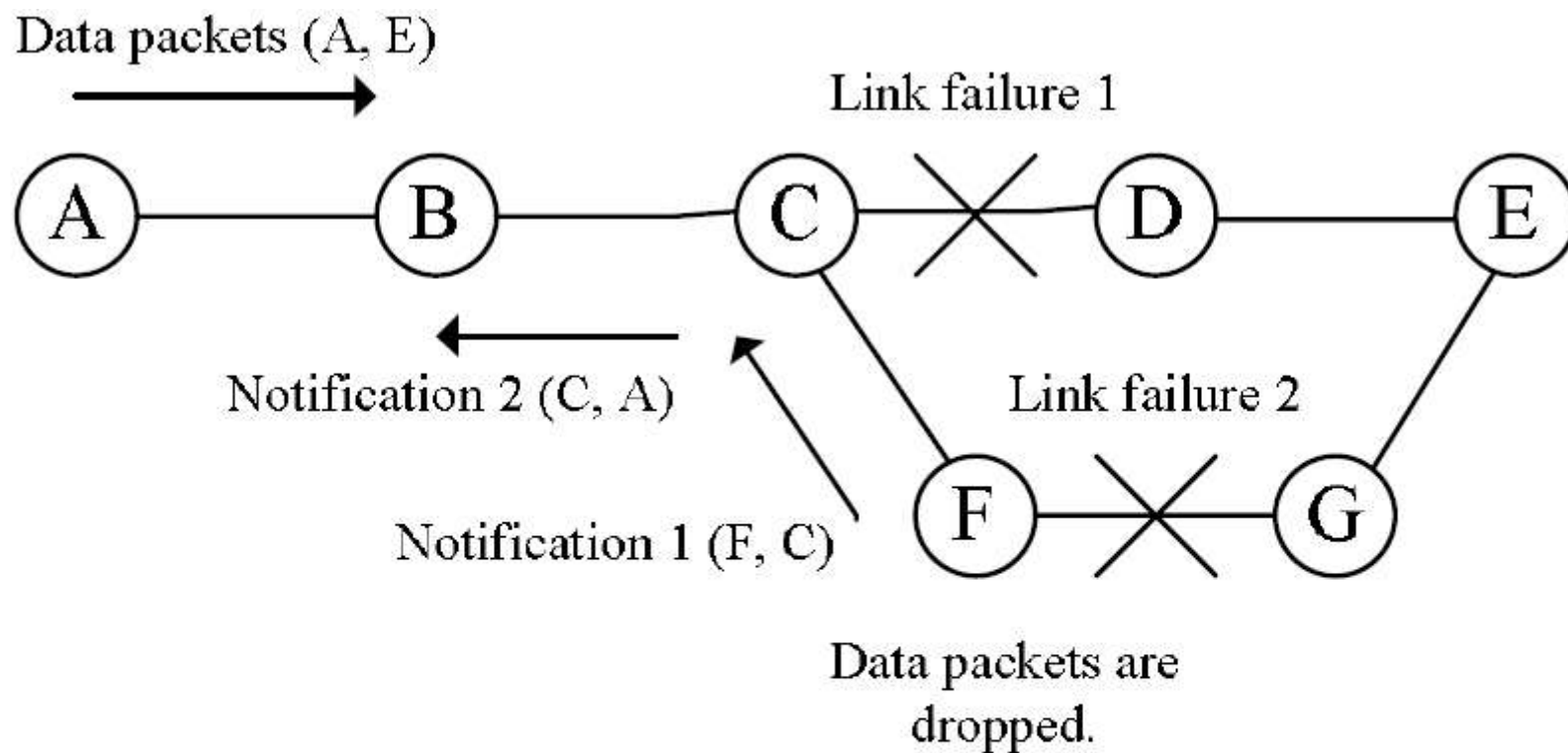
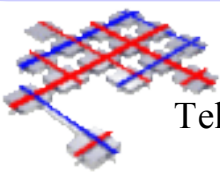


Fig.5 Example of EPLN [2]



Telematics  
Group

## BEAD Example

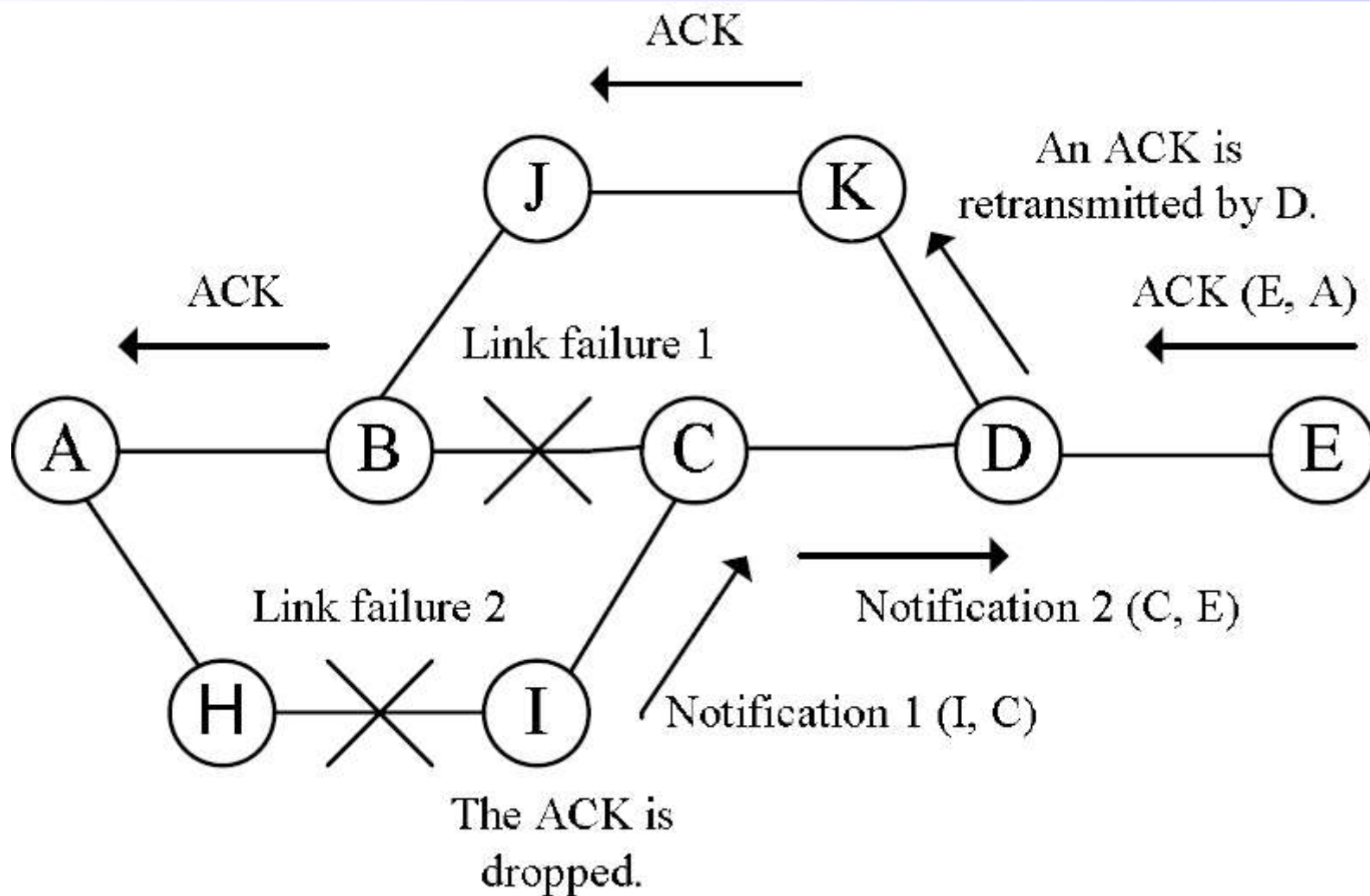
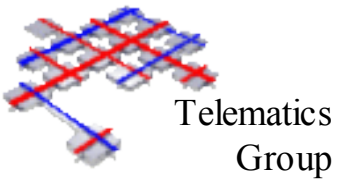
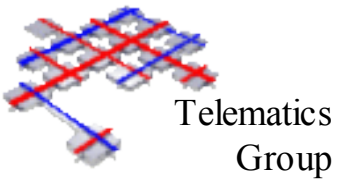


Fig.6 Example of BEAD [2]

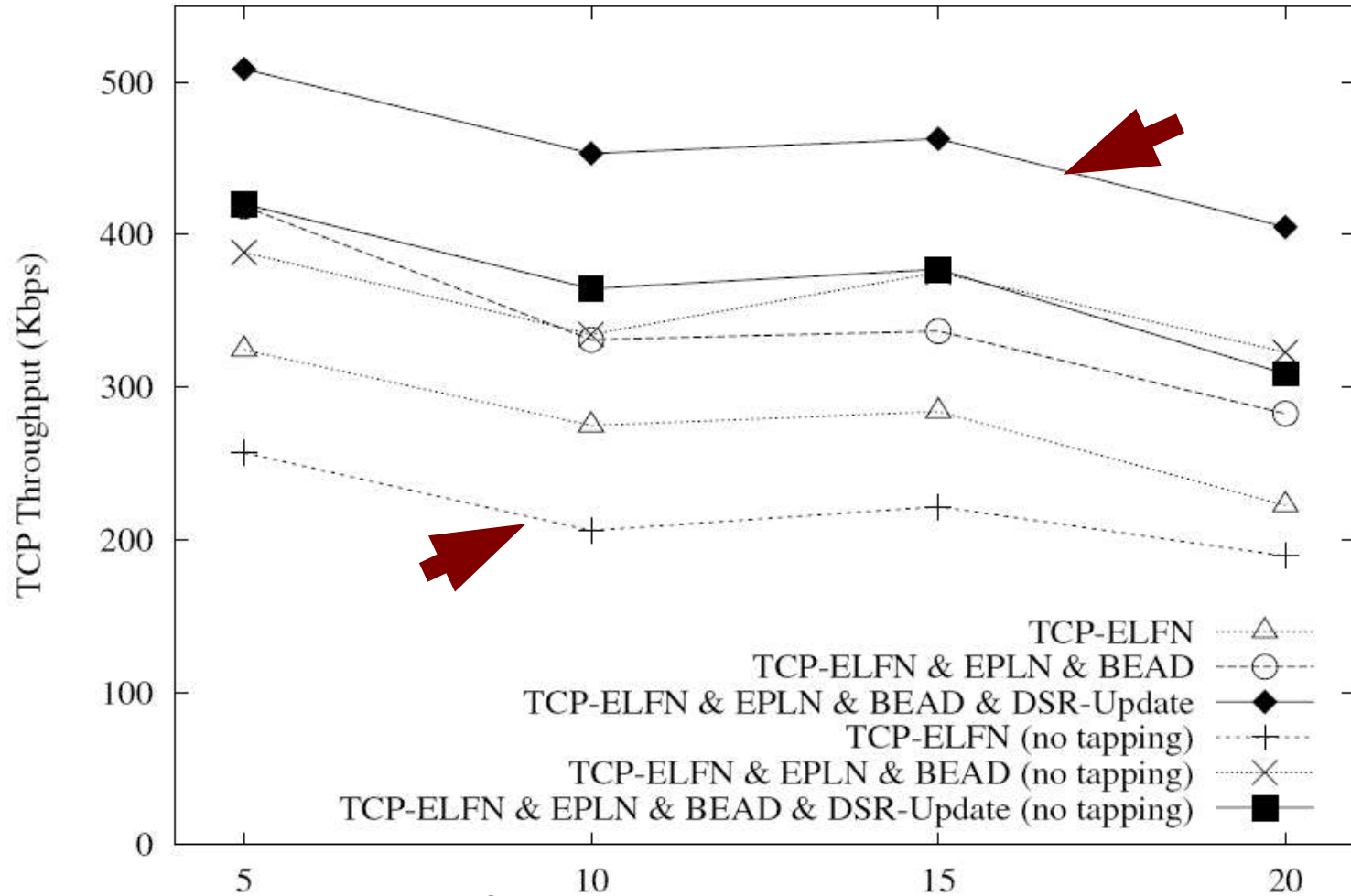


## Cross-Layer Interaction

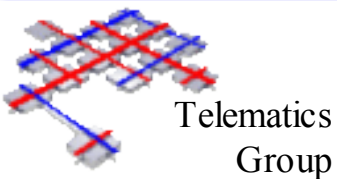
- BEAD: routing protocol attempts to retransmit ACK for lost ACK without cross-layer information exchange
- Cross-layer interaction in EPLN:
  - Network layer sends ICMP packet to TCP for every lost packet
  - ICMP packet contains Sequence Number AND information whether packet is lost (packet could aswell only be salvaged by intermediate node)



## Evaluation (I) – TCP throughput vs mobility



**TCP-ELFN & EPLN & BEAD & DSR\_Update  
outperform other optimizations!**



## Evaluation (II) – slow starts vs mobility

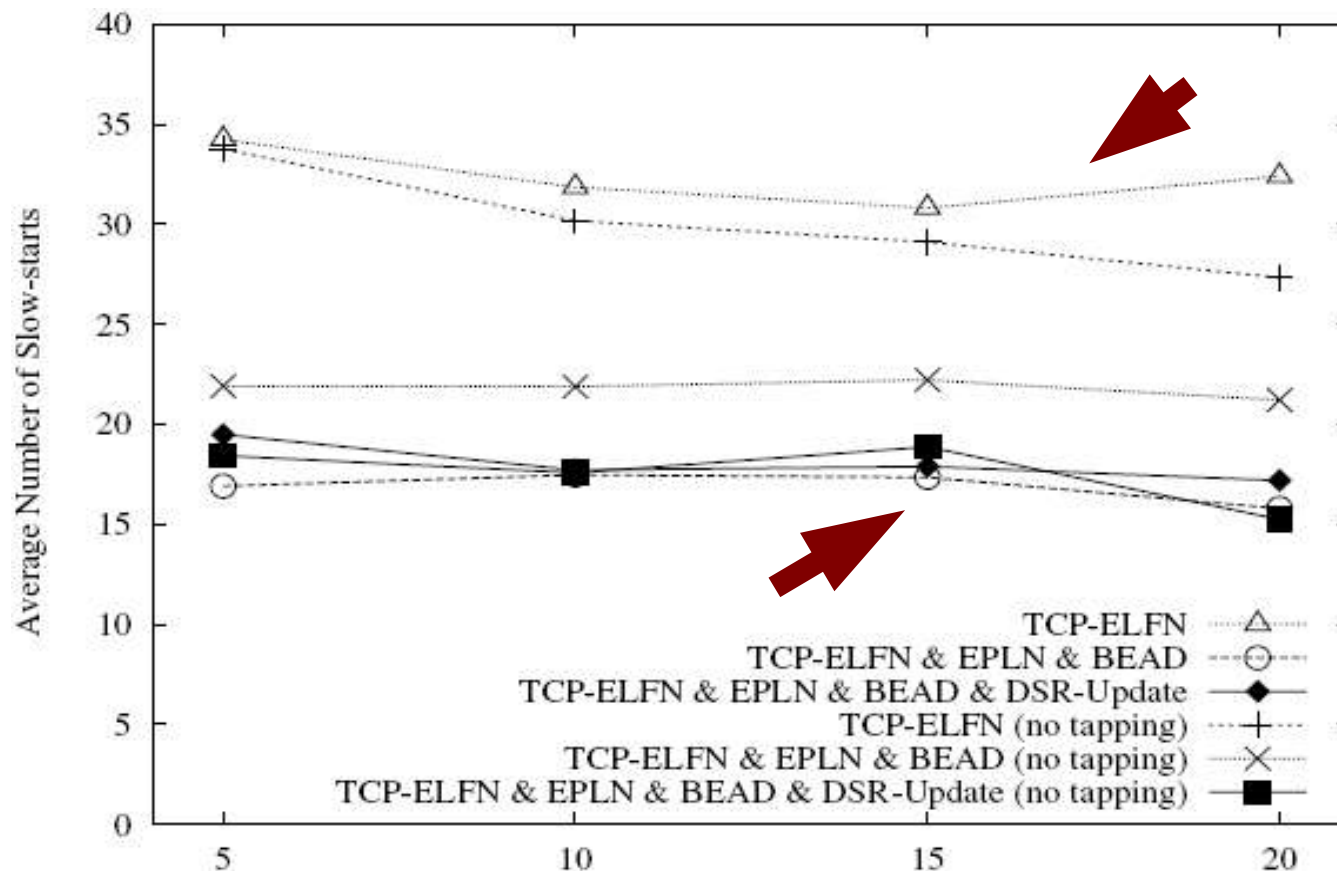
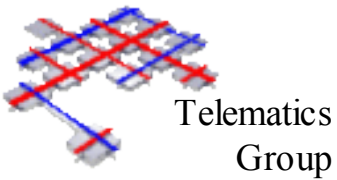


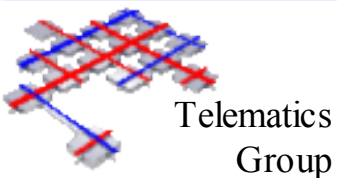
Fig.8 Average number of slow starts v. mobility (m/s) [2]

Significant reduction in slow starts with all optimizations enabled!



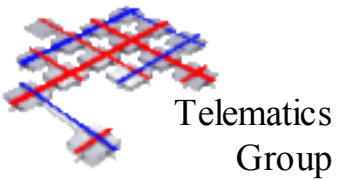
## Is it any good?

- Some facts:
  - TCP throughput more than doubled (210%)
  - TCP timeouts reduced by almost half (44%)
  - TCP slow-starts reduced by factor 15 (35-2)
  
- „Cross-layer information awareness is key to making TCP efficient in the presence of mobility“<sup>[2]</sup>
  
- Necessity for network layer to:
  1. inform TCP senders about lost packets
  2. retransmit ACKs for lost ACKs



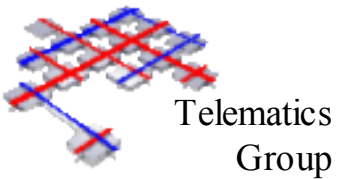
## Approaches for Wireless Wide-Area Networks

- Similar situation for TCP in GPRS-based WWANs
  - ...and also lower bandwidth
  - Observation: HTTP performance very bad! (only up to ~50% of max. speed)
  - **Although** TCP performance quite good (max. BW)
  - **REASON:** Inefficiencies in Application and Session Layer
- Find optimizations for WWANs in different layers



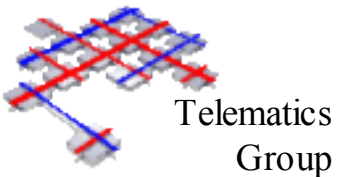
## Optimizations in different layers (I)

- Application layer:
  - HTTP pipelining (experimental!)
  - Dynamic content compression
- Session layer:
  - Varying number of TCP connections
  - URL/DNS-Rewriting



## Optimizations in different layers (II)

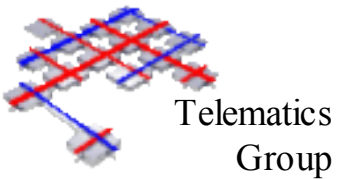
- Transport Layer:
  - TCP WWAN (Proxy, avoid excessive queues)
  - UDP GPRS (optimized for GPRS, prevent link stalls)
- Link layer:
  - Dynamically choose Forward Error Correction (FEC)



## Performance benefits

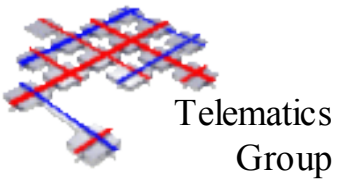
<b>Layer</b>	<b>Kind</b>	<b>Benefit</b>	<b>Reason</b>
Application	Dynamic Content Compression	11 - 17%	Amount of small files
	Pipelining	35 - 56%	Amount of small files
Session	Varying number of TCP connections	35 - 42%	Idle time waiting for reply to GET decreased
	URL/DNS-Rewriting	+ 5 - 9%	Only one DNS lookup needed
Transport	TCP WWAN	+ 5 - 13%	TCP timeouts decreased
	UDP GPRS	+ 7 - 14%	Less slow-starts
Link	FEC	+ 5 - 20%	Adjust FEC values

Tab. 1: Performance gains for various approaches in WWAN



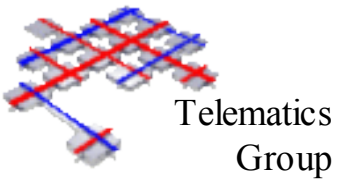
## Conclusions (I)

- TCP not prepared for challenges of mobility
- ...but performs quite well in GPRS
  - Link and Route failures biggest problem
  - Solutions:
    - **Caching** of routes
    - **Retransmission** of lost packets
- Exploiting cross-layer information provides large benefit (throughput doubled etc.)



## Conclusions (II)

- In WWANs:
  - Even more optimization possible
  - Different layers: Application, Session, Transport, Link
  - Combination of optimizations provide up to 70 % gain for performance (latency)



Thank you for your attention!  
Questions?

- [1] Dynamic Source Routing, <http://wiki.uni.lu/secan-lab/Dynamic+Source+Routing.html>
- [2] X. Yu, „Improving TCP Performance over Mobile Ad Hoc Networks by Exploiting Cross-Layer Information Awareness“, Proc. of the 10th MobiCom, Oct. 2004
- [3] R. Chakravorty et al. , „Performance Optimizations for Wireless Wide-area Networks: Comparative Study and Experimental Evaluation“, Proc. of the 10th MobiCom, Oct. 2004