Distributed Systems
Master of Science in Engineering in Computer Science

AA 2020/2021

LECTURE 2B: LINKS

Links

Used to model the network component of a distributed system

Connects pairs of processes

Messages exchanged between processes over a link are unique

Link Abstractions under crash failure assumption

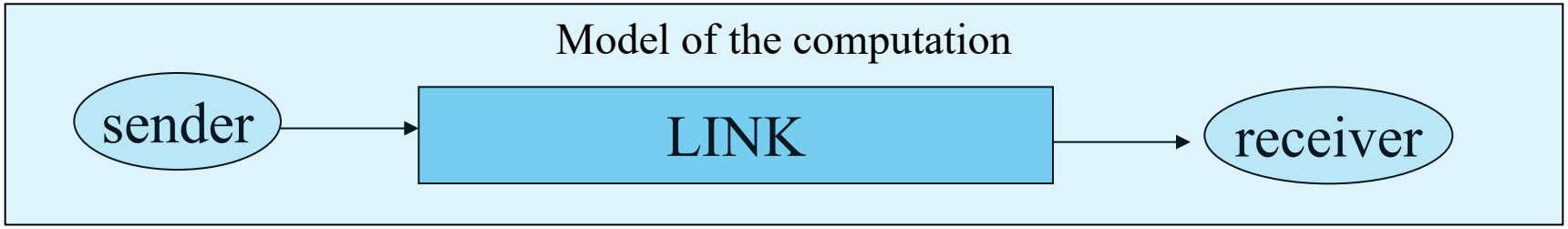
1. *fair-loss links* (captures the basic idea that messages might be lost but the probability for a message not to be lost is nonzero).

2. *Stubborn links*

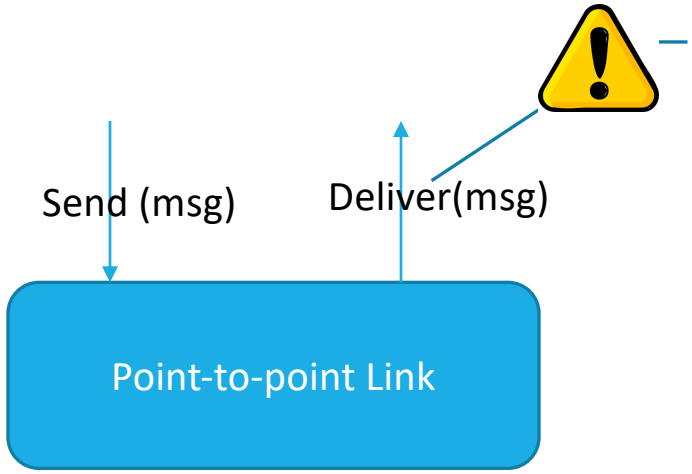
3. *Perfect links*

System Model

- Two processes (sender and receiver)
- Messages
  - can be lost
  - experience an unpredictable time to reach the destination
- Processes
  - can crash
  - The time taken by each process to execute an operation is bounded (such a bound can be unknown)



A generic link interface



Pay Attention!
Deliver is different from receive

A message is typically received at a given port of the network and stored within some buffer, and then some algorithm is executed to make sure the properties of the required link abstraction are satisfied, before the message is actually delivered.

Fair-loss Point-to-Point Link: Specification

Module 2.1: Interface and properties of fair-loss point-to-point links

Module:

Name: FairLossPointToPointLinks, **instance** *fll*.

Events:

Request: $\langle$ *fll*, *Send* | q, m $\rangle$: Requests to send message m to process q.

Indication: $\langle$ *fll*, *Deliver* | p, m $\rangle$: Delivers message m sent by process p.

Properties:

FLL1: *Fair-loss:* If a correct process p infinitely often sends a message m to a correct process q, then q delivers m an infinite number of times.

FLL2: *Finite duplication:* If a correct process p sends a message m a finite number of times to process q, then m cannot be delivered an infinite number of times by q.

FLL3: *No creation:* If some process q delivers a message m with sender p, then m was previously sent to q by process p.

Fair-loss Point-to-Point Link: Issues

The sender must take care of the retransmissions if it wants to be sure that a message m is delivered at its destination.

Stubborn Link

The specification does not guarantee that the sender can stop the retransmission of each message

Quiescient Implementation

Each message may be delivered more than once.

Perfect Link

Stubborn Point-to-Point Link: Specification

Module 2.2: Interface and properties of stubborn point-to-point links

Module:

Name: StubbornPointToPointLinks, **instance** *sl*.

Events:

Request: $\langle$ *sl*, *Send* | q, m $\rangle$: Requests to send message m to process q.

Indication: $\langle$ *sl*, *Deliver* | p, m $\rangle$: Delivers message m sent by process p.

Properties:

SL1: *Stubborn delivery:* If a correct process p sends a message m once to a correct process q, then q delivers m an infinite number of times.

SL2: *No creation:* If some process q delivers a message m with sender p, then m was previously sent to q by process p.

Stubborn Point-to-Point Link: Implementation

Algorithm 2.1: Retransmit Forever

Implements:
StubbornPointToPointLinks, **instance** *sl*.

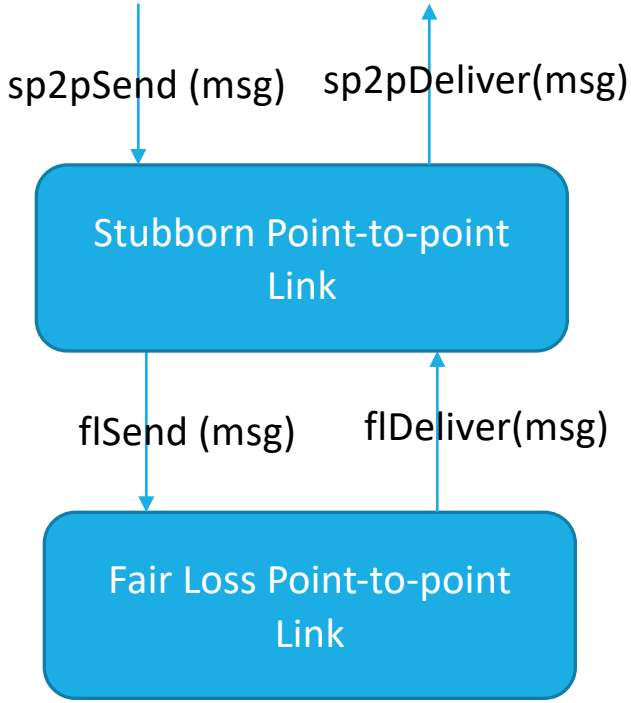
Uses:
FairLossPointToPointLinks, **instance** *fll*.

upon event ⟨ *sl*, *Init* ⟩ **do**
sent := ∅;
starttimer(Δ);

upon event ⟨ *Timeout* ⟩ **do**
forall $(q, m) \in sent$ **do**
trigger ⟨ *fll*, *Send* | q, m ⟩;
starttimer(Δ);

upon event ⟨ *sl*, *Send* | q, m ⟩ **do**
trigger ⟨ *fll*, *Send* | q, m ⟩;
sent := *sent* ∪ $\{(q, m)\}$;

upon event ⟨ *fll*, *Deliver* | p, m ⟩ **do**
trigger ⟨ *sl*, *Deliver* | p, m ⟩;



Perfect Point-to-Point Link: Specification

Module 2.3: Interface and properties of perfect point-to-point links

Module:

Name: PerfectPointToPointLinks, **instance** pl.

Events:

Request: $\langle$ pl, *Send* | q, m $\rangle$: Requests to send message m to process q.

Indication: $\langle$ pl, *Deliver* | p, m $\rangle$: Delivers message m sent by process p.

Properties:

PL1: *Reliable delivery:* If a correct process p sends a message m to a correct process q, then q eventually delivers m.

PL2: *No duplication:* No message is delivered by a process more than once.

PL3: *No creation:* If some process q delivers a message m with sender p, then m was previously sent to q by process p.

Perfect Point-to-Point Link: Implementation

Algorithm 2.2: Eliminate Duplicates

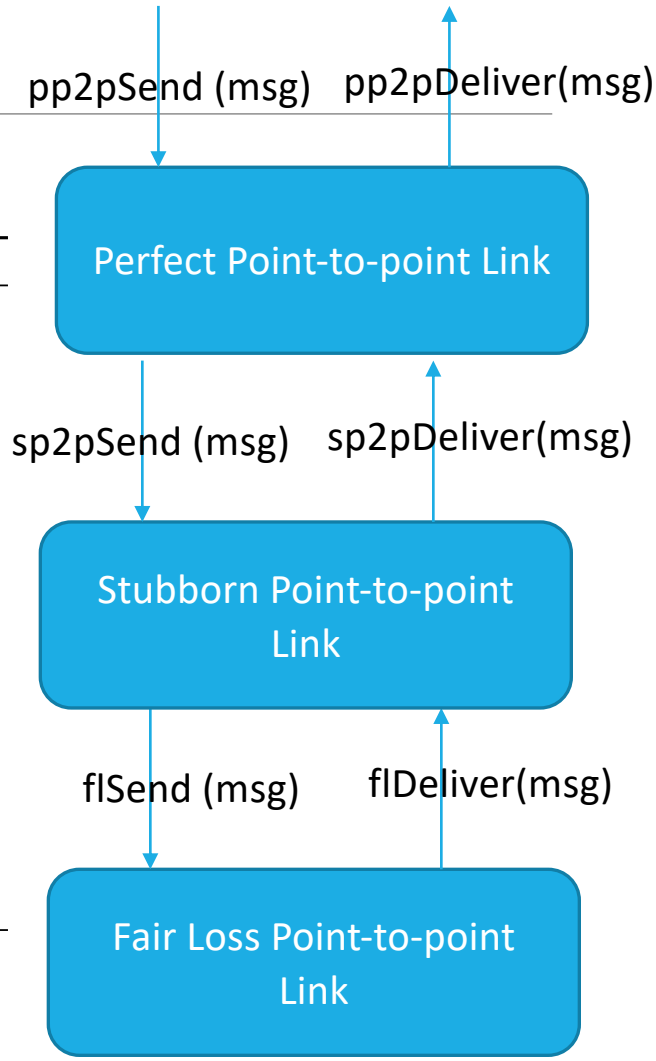
Implements:
PerfectPointToPointLinks, **instance** *pl*.

Uses:
StubbornPointToPointLinks, **instance** *sl*.

upon event ⟨ *pl*, *Init* ⟩ **do**
delivered := ∅;

upon event ⟨ *pl*, *Send* | *q*, *m* ⟩ **do**
trigger ⟨ *sl*, *Send* | *q*, *m* ⟩;

upon event ⟨ *sl*, *Deliver* | *p*, *m* ⟩ **do**
if $m \notin$ *delivered* **then**
delivered := *delivered* ∪ {*m*};
trigger ⟨ *pl*, *Deliver* | *p*, *m* ⟩;



Links: conclusion

Relation with existing protocols stacks

Performance issues