Failure, connectivity, and disconnection detectors

Lynda Temal and Denis Conan

Mobilité et Ubiquité 2004
2nd June 2004
Outline

1. Motivations and objectives for terminal mobility ........................................... 3
2. Distributed system model .................................................................................... 5
3. Failure detection .................................................................................................... 8
4. Connectivity detection ......................................................................................... 12
5. Disconnection detection ....................................................................................... 16
6. Mixing and using detectors .................................................................................. 20
7. Conclusion and future work .................................................................................. 23
1 Motivations and objectives for terminal mobility

Disconnection management: Need for service continuity

♦ Voluntary disconnection: Save battery, minimise the probability of unexpected disconnections...

♦ Involuntary disconnection: Move out of the reach of a base station...
1.1 Disconnection and failure

- (Acceptable/Required) Disconnections $\neq$ (Unacceptable/Tolerated) Failures
  - Detectors (failure, connectivity, and disconnection), and consensus
    - Disconnection management $\neq$ Fault tolerance
2 Distributed system model

2.1 Partially synchronous model .......................................................... 6
2.2 Failure and disconnection models .................................................... 7
2.1 Partially synchronous model

- Time to deliver a message or to execute a process step is bounded
- Bounds are unknown
- Bounds exist but after a Global Stabilisation Time (GST) also unknown
  - Alternation of unstabilised and stabilised periods of time
- Fully connected network of hosts with reliable links (e.g., using TCP)
- Existence of primitives for reliably broadcasting messages [Hadzilacos and Toueg, 1994]
- One process per host and global virtual clock (only to ease the presentation)
2.2 Failure and disconnection models

- **Failure model:**
  - Processes fail by crashing, *e.g.* premature halting (fail silent model)
  - Once crashed (faulty), a process does not recover
  - There always exists one correct (non-faulty) process

- **Disconnection model:**
  - Voluntary or involuntary disconnections
  - Every process starts while being connected
## 3 Failure detection

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Role</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Properties</td>
<td>10</td>
</tr>
<tr>
<td>3.3 Algorithm HB</td>
<td>11</td>
</tr>
</tbody>
</table>
3.1 Role

- **Consensus**: a fundamental paradigm of fault-tolerance [Chandra and Toueg, 1996]

- **Impossibility** to reach a consensus in an asynchronous distributed system [Fischer et al., 1985]
  - Due to the impossibility to state whether a process or a link is very slow or faulty

- **Unreliable failure detectors** encapsulate the partial asynchrony
3.2 Properties

- **Completeness** measures the safety and **accuracy** measures the liveness

In the following: \(\mathcal{HB}\): Strong completeness and eventual weak accuracy
3.3 Algorithm HB

[Aguilera et al., 1997]: Strong completeness and eventual weak accuracy

```plaintext
for every process p:
  initialisation:
    for all q ∈ neighbour(p)
      \( D_p[q] \leftarrow 0 \)  \{vector of received heartbeats numbers\}
  cobegin:
    || task 1: repeat forever
      for all q ∈ neighbour(p)
      send(HEARTBEAT) to q
    || task 2: upon receive(HEARTBEAT) from q
      \( D_p[q] \leftarrow D_p[q] + 1 \)
  coend
```

- No timeouts and no set of faulty processes
- Add a ‘simple’ time-dependent algorithm that builds the set of faulty processes
4 Connectivity detection

4.1 Role ................................................................. 13
4.2 Property and proposed solution ................................. 14
4.3 State diagram and algorithm ................................. 15
4.1 Role

- Resource management is interpreted for connectivity management
  - Battery
  - Network bandwidth, per network interface
  - Memory

- When the level of availability of a local resource is insufficient, then disconnect
4.2 Property and proposed solution

- Prevention of the “Ping-Pong” effect
  - Avoid too frequent state transfers and log flushings
- \textit{CD}: Hysteresis [Conan et al., 2002, Conan et al., 2003]
4.3 State diagram and algorithm

Algorithm $CD$

"resV" means "level of resource availability"

"c", "p", and "d" mean "connected", "partially connected", and "disconnected", respectively
5 Disconnection detection

5.1 Role and principle of the solutions ......................................................... 17
5.2 Properties ................................................................................................. 18
5.3 Algorithm DD ............................................................................................ 19
5.1 Role and principle of the solutions

- From ‘local’ connectivity to ‘distributed’ connectivity
  - Design of a distributed solution

- Alert (notify) remote processes when disconnecting and reconnecting
  - Take advantage of the connectivity information to send messages
    - DISCONNECT messages when transitioning from connected to disconnected
    - RECONNECT messages when transitioning from disconnected to connected
  - When not enough time to send these messages, assume middleware can store and forward
    - ‘Rapid’ disconnections cannot be seen
      - They may be seen as failures
5.2 Properties

- **In the following: DD:** Weak completeness and strong accuracy of disconnection
- **From weak to strong completeness** of disconnection using a reduction algorithm
  - Connected processes periodically broadcast the set of processes seen disconnected
5.3 Algorithm DD

for every process $p$ :

initialisation :

$\text{disc}_p \leftarrow \emptyset$

for all $q \in \Pi$

$N_p[q] \leftarrow 0$

{set of processes seen disconnected}

{vector of received disconnection/reconnection numbers}

cobegin :

|| task 1 : upon change mode notification

if $mode = \text{‘d’}$

for all $q \in \Pi \setminus \text{disc}_p$

send(DISCONNECT, $N_p[p]$) to $q$

$N_p[p] \leftarrow N_p[p] + 1$

else

for all $q \in \Pi$

send(RECONNECT, $N_p[p]$) to $q$

$N_p[p] \leftarrow N_p[p] + 1$

|| task 2 : upon receive(DISCONNECT, $n_q$) from $q$

if $q \notin \text{disc}_p \land N_p[q] < n_q$ then $\text{disc}_p \leftarrow \text{disc}_p \cup \{q\}$

$N_p[q] \leftarrow \max(N_p[q] + 1, n_q)$

|| task 3 : upon receive(RECONNECT, $n_q$) from $q$

if $q \in \text{disc}_p \land N_p[q] < n_q$ then $\text{disc}_p \leftarrow \text{disc}_p \setminus \{q\}$

$N_p[q] \leftarrow \max(N_p[q] + 1, n_q)$

coend
6 Mixing and using detectors

6.1 Failure detectors to complete connectivity detection ....................... 21
6.2 Disconnection detectors to improve failure detection and consensus .......... 22
6.1 Failure detectors to complete connectivity detection

**Motivations:**
- Take into account the **quality of the links**
  - End-to-end connectivity or end-to-end QoS

**Usage:**
- A failure detector per host/application/process/link
- Extract $\Delta D$ from $\mathcal{DD}$ and translate it on the hysteresis
- For a better accuracy of the measure, vary the heartbeat frequency in accordance with the QoS
6.2 Disconnection detectors to improve failure detection and consensus

Motivations:

- Improve the complexity in terms of messages
  - Do not broadcast heartbeats to processes seen disconnected

- Modify the semantics of the consensus:
  - Once disconnected, a process is excluded from the computation
    - A reconciliation is required before reintegrating disconnected processes
  - Other semantics can be specified in conjunction with architectural concerns
    - This is an area of research
7 Conclusion and future work

Failure Vs. Disconnection Vs. Partition

Disconnection algorithm and failure detector algorithm when partitioning

Disconnection Vs. Failure and partition in group communication

Architecture of distributed applications when partitioning
References


