FAULT MODELS

• Much work has been done on fault models. The discussion is based on the paper:

  • Thambidurai, P., and You-Keun Park, "Interactive Consistency with Multiple Failure Modes", Reliable Distributed Systems, Volume, Issue, 10-12 Oct 1988 Page(s):93 - 100. (Only read up to Section 3).

  • There is an interesting follow-up paper "Verification of Hybrid Byzantine Agreement Under Link Faults" by P. Lincoln and J. Rushby that addresses a problem in the algorithm of Thambidurai and Park.
FAULT MODELS

• Benign versus Malicious
  • Benign
    • error is self-evident
    • component does not undergo incorrect state transition during failure
  • examples:
    • omission fault
    • crash fault
    • timing fault
    • data out-of-bound
FAULT MODELS

• Malicious
  • not self-evident to all non faulty receivers
  • can behave in two ways
  • symmetric
    • received identically by all processors
  • asymmetric
    • no restrictions of fault => anything goes

• Fault frequency
  • worse case every fault could behave asymmetric
  • best case all faults are benign
  • what is the best assumption for your system?
FAULT MODELS

• Fault Taxonomy

Fault

Benign

Malicious

Symmetric

Asymmetric

• Relationship & Probability of Occurrence
FAULT MODELS

- Lamport Model
  - assumes that every fault is asymmetric

\[ N \geq 3t + 1 \]
\[ r' \geq t + 1 \quad \text{or} \quad r \geq t \quad \text{rebroadcasts} \]

- Meyer + Pradhan 87
  - differentiates between malicious and benign faults

\[ N > 3m + b \]
\[ r > m \]
\[ m = \text{number of malicious faults} \]
\[ b = \text{number of benign faults} \]
FAULT MODELS

• Thambidurai + Park 88
  • difference between malicious faults
    • symmetric faults
    • asymmetric faults
  • result:
    \[ N > 2a + 2s + b + r \]
    \[ r \geq a \]

• \( a = \text{asym.}, \ s = \text{sym.}, \ b = \text{benign}, \ r = \text{rounds} \)
• in general \( a_{\text{max}} < s_{\text{max}} < b_{\text{max}} \)
• or \( \lambda_a << \lambda_s << \lambda_b \)
• saves rounds and hardware
FAULT MODELS

- Advantages of multi-fault model
  - 1) more accurate model of the system
    - less “overly conservative”
  - 2) resulting reliabilities are better
    - custom tailor recovery mechanisms
- Example:
  - consider Byzantine solution using OM() algorithm
  - assume N = 4, 5, 6
  - still, only one fault is covered using the OM algorithm
  - moreover, the system reliability degrades
    - N = 6 results in worse reliability than N = 4
    - one is better off to turn the additional processors off!
- see paper Tha88, page 98, table 1
## Fault Models

Source: Tha88

<table>
<thead>
<tr>
<th>Model</th>
<th>$N$</th>
<th>$P(\text{Failure})$</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>BG</td>
<td>4</td>
<td>$6.0 \times 10^{-8}$</td>
<td>1 arbitrary</td>
</tr>
<tr>
<td>BG</td>
<td>5</td>
<td>$1.0 \times 10^{-7}$</td>
<td>1 arbitrary</td>
</tr>
<tr>
<td>BG</td>
<td>6</td>
<td>$1.5 \times 10^{-7}$</td>
<td>1 arbitrary</td>
</tr>
<tr>
<td>UM</td>
<td>4</td>
<td>$6.0 \times 10^{-8}$</td>
<td>1 arbitrary, $b = 0, s = 0$</td>
</tr>
<tr>
<td>UM</td>
<td>5</td>
<td>$1.0 \times 10^{-11}$</td>
<td>1 arbitrary, $b = 1, s = 0$</td>
</tr>
<tr>
<td>UM</td>
<td>6</td>
<td>$2.0 \times 10^{-11}$</td>
<td>1 arbitrary, $b = 0, s = 1$</td>
</tr>
<tr>
<td>UM</td>
<td>6</td>
<td>$1.1 \times 10^{-15}$</td>
<td>1 arbitrary, $b = 2, s = 0$</td>
</tr>
</tbody>
</table>

Table 1: Reliability data for Example 1
**FAULT MODELS**

Source: Tha88

![Table 2: Resiliency of a System based on the Unified Model (minimum number of processors required)](image)

<table>
<thead>
<tr>
<th></th>
<th>( a = 0 )</th>
<th>( a = 1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( s ) 0 1 2 3</td>
<td>( s ) 0 1 2 3</td>
</tr>
<tr>
<td></td>
<td>( b = 0 ) 4 6 8</td>
<td>( b = 0 ) 4 6 8</td>
</tr>
<tr>
<td></td>
<td>( b = 1 ) 3 5 7 9</td>
<td>( b = 1 ) 5 7 9</td>
</tr>
<tr>
<td></td>
<td>( b = 2 ) 4 6 8 10</td>
<td>( b = 2 ) 6 8 10</td>
</tr>
<tr>
<td></td>
<td>( b = 3 ) 5 7 9 11</td>
<td>( b = 3 ) 7 9 11</td>
</tr>
<tr>
<td></td>
<td>( b = 4 ) 6 8 10 12</td>
<td>( b = 4 ) 8 10 12</td>
</tr>
<tr>
<td></td>
<td>( b = 5 ) 7 9 11 13</td>
<td>( b = 5 ) 9 11 13</td>
</tr>
<tr>
<td></td>
<td>( b = 6 ) 8 10 12 14</td>
<td>( b = 6 ) 10 12 14</td>
</tr>
</tbody>
</table>
FAULT MODELS

• 3) smarter degradation
  • we can specify the number of rounds
  • example using $N = 11$
    • let subscript $\text{max}$ denote the maximum number of faults covered, assuming this is the only type of fault occurring.
      • if $r = 1$ then $a_{\text{max}} = 1$ or $s_{\text{max}} = 4$
      • if $r = 2$ then $a_{\text{max}} = 2$ or $s_{\text{max}} = 4$
        why? $s_{\text{max}} = 4$ => $N > 2 \times 4 + 2 = 10$
        $s_{\text{max}} = 5$ => $N < 2 \times 5 + 2 = 12$
  • requirements for success
    • good estimate of fail rates $\lambda_a$, $\lambda_s$, $\lambda_b$
      • typically $\lambda_a << \lambda_s << \lambda_b$
    • good estimate of recovery rates $\rho_a$, $\rho_s$, $\rho_b$
      • typically $\rho_a < \rho_s < \rho_b$
AGREEMENT ALGORITHMS

• Incomplete Interconnections
  • Lam82, Dol82
  • agreement only if the number of processors is less than 1/2 of
    the connectivity of the system’s network.
• Eventual vs. Immediate Byz. Agreement (EBA, IBA)
  • recall interactive consistency conditions IC1, IC2
  • an agreement is immediate if in addition to IC1 and IC2 all
    correct processors also agree (during the round) on the
    round number at which they reach agreement.
  • otherwise the agreement is called eventual
    • each processor has decided on its value, but cannot
      synchronize its decision with that of the others until some
      later phase.
  • Thus, agreement may not always need full t+1 rounds