Is There Such a Thing as Software Reliability?

Dick Hamlet

Department of Computer Science
Portland State University
Portland, OR 97207
hamlet@cs.pdx.edu
The probability $R(N)$ that sth. will operate according to specification $N$ times in a row
The probability $R(N)$ that sth. will operate according to specification $N$ times in a row

- The specification is essential – defines *failure*
- Large enough $N$ always means $R(N) = 0$
- How to measure (estimate) $R$?
- Confidence in measurements?
Life Testing Toasters

1. Start with a new (one-slice) toaster \( i \)

2. Toast 1, 2, ..., \( N_i \) pieces of bread until toaster \( i \) fails

3. Repeat 1 – 2 for \( i = 1, 2, ..., m \)
Life Testing Toasters

1. Start with a new (one-slice) toaster $i$
2. Toast 1, 2, ..., $N_i$ pieces of bread until toaster $i$ fails
3. Repeat 1 – 2 for $i = 1, 2, ..., m$
4. Reliability estimate $\hat{R}(j)$ is the fraction of $m$ for which $N_i \geq j$
Life Testing Toasters

1. Start with a new (one-slice) toaster $i$

2. Toast 1, 2, ..., $N_i$ pieces of bread until toaster $i$ fails

3. Repeat 1 – 2 for $i = 1, 2, ..., m$

4. Reliability estimate $\hat{R}(j)$ is the fraction of $m$ for which $N_i \geq j$
   - If out of 100, one toaster fails first at 379 slices and one lasts longest to 420 slices, then
     - $\hat{R}(1) = \hat{R}(2) = ... = \hat{R}(378) = 1.0$
     - $\hat{R}(379) = 0.99$, ..., $\hat{R}(419) = 0.01$
     - $\hat{R}(420) = \hat{R}(421) = ... = 0$
Life Testing Toasters

1. Start with a new (one-slice) toaster \( i \)

2. Toast 1, 2, ..., \( N_i \) pieces of bread until toaster \( i \) fails

3. Repeat 1 – 2 for \( i = 1, 2, ..., m \)

4. Reliability estimate \( \hat{R}(j) \) is the fraction of \( m \) for which \( N_i \geq j \)

5. An estimate of the mean runs to failure (MRTF) is the average \( \bar{N} \) of \( N_i \)
Life Testing Toasters

1. Start with a new (one-slice) toaster $i$

2. Toast 1, 2, ..., $N_i$ pieces of bread until toaster $i$ fails

3. Repeat 1 – 2 for $i = 1, 2, ..., m$

4. Reliability estimate $\hat{R}(j)$ is the fraction of $m$ for which $N_i \geq j$

5. An estimate of the mean runs to failure (MRTF) is the average $\bar{N}$ of $N_i$

6. Calculate standard error $\sigma$ of the $N_i$

7. Confidence that the actual MRTF is within the interval [$\bar{N} - 2\sigma$, $\bar{N} + 2\sigma$] is roughly 95%
Assumptions Required for Life Testing (Toasters)

1. Toaster behavior is continuous
2. Toasters have no systematic cause of failure
3. Each toasting run is independent of the others (Bernoulli trials)
4. Test circumstances duplicate actual toasting

Then the measured MRTF and confidence are accurate predictions
Assumptions Required for Life Testing (Toasters)

1. Toaster behavior is continuous
2. Toasters have no systematic cause of failure
3. Each toasting run is independent of the others (Bernoulli trials)
4. Test circumstances duplicate actual toasting

Then the measured MRTF and confidence are accurate predictions

Interaction between 2 and 4:
- Gap in screen guarding the heating element (2 false)
- Tests use sliced bread but usage is for fat bagels (4 false)
- Bagel protrudes through the gap and burns out heating element long before the predicted MRTF
Can’t Wait for Failure?

Five years of breakfast is about 3000 slices

• Time for each toaster to fail is about $\bar{N}$, the MRTF

• 3,000 slices can be toasted in about 5 days (@2 min each – overheating violates assumption 4!)

• But suppose the MRTF is actually 30,000?!
Can’t Wait for Failure?

Five years of breakfast is about 3000 slices

- Time for each toaster to fail is about $\bar{N}$, the MRTF
- 3,000 slices can be toasted in about 5 days (@2 min each – overheating violates assumption 4!)
- But suppose the MRTF is actually 30,000?!

What can be predicted from runs that do not fail?

- Confidence $C$ that the failure probability is below $f_{\text{max}}$ based on $T$ runs: $C = 1 - (1 - f_{\text{max}})^T$
- For toasters, 95% confidence in a MRTF of better than 3000 requires $T \approx 9000$
- For $T = 3000$:

<table>
<thead>
<tr>
<th>$MRTF$</th>
<th>95%</th>
<th>75%</th>
<th>63%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2200</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4300</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Objections to *Software* Reliability

- Software need *never* fail: $R(N) = 1.0$ for all $N$
Objections to *Software* Reliability

- Software need *never* fail: $R(N) = 1.0$ for all $N$
  OK, predict a large enough MRTF from no failures
Objections to *Software* Reliability

- Software need *never* fail: \( R(N) = 1.0 \) for all \( N \)
- Software failure isn’t probabilistic – each run is deterministic
Objections to Software Reliability

- Software need never fail: \( R(N) = 1.0 \) for all \( N \)
- Software failure isn’t probabilistic – each run is deterministic

The minefield analogy
Objections to Software Reliability

- Software need never fail: $R(N) = 1.0$ for all $N$
- Software failure isn’t probabilistic – each run is deterministic

The minefield analogy
Objections to *Software* Reliability

- Software need *never* fail: $R(N) = 1.0$ for all $N$
- Software failure isn’t probabilistic – each run is deterministic

The minefield analogy
Objections to *Software* Reliability

- Software need *never* fail: $R(N) = 1.0$ for all $N$
- Software failure isn’t probabilistic – each run is deterministic

*The minefield analogy*
Objections to Software Reliability

- Software need never fail: $R(N) = 1.0$ for all $N$
- Software failure isn’t probabilistic – each run is deterministic

The minefield analogy

12 rocks thrown with no explosion
⇒ 70% confidence that 10 steps are safe

<table>
<thead>
<tr>
<th>Mindfield</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>input space</td>
</tr>
<tr>
<td>mines</td>
<td>failure inputs</td>
</tr>
<tr>
<td>path</td>
<td>usage profile</td>
</tr>
<tr>
<td>rocks thrown</td>
<td>tests executed</td>
</tr>
<tr>
<td>explosions</td>
<td>failures</td>
</tr>
<tr>
<td>steps on the path</td>
<td>runs</td>
</tr>
</tbody>
</table>
Objections to \textit{Software} Reliability

- Software need \textit{never} fail: $R(N) = 1.0$ for all $N$
- Software failure isn’t probabilistic – each run is deterministic
- Assumptions of life testing are false
Objections to *Software* Reliability

- Software need *never* fail: \( R(N) = 1.0 \) for all \( N \)
- Software failure isn’t probabilistic – each run is deterministic

- Assumptions of life testing are false
  1. Toaster behavior is continuous
  2. Toasters have no systematic cause of failure
  3. Each toasting run is independent of the others (Bernuilli trials)
  4. Test circumstances duplicate actual toasting
Conclusions

Theory is lacking

- How important is continuity?
- Is MRTF well defined for software?
- Study minefield simulations?