Optimal Reliability Allocation

Yashwant K. Malaiya
malaiya@cs.colostate.edu

Department of Computer Science
Colorado State University

Reliability Allocation Problem

- Allocation the reliability values to subsystems
  - to minimize the total cost
  - while achieving the reliability target.
- Widely applicable
  - Software systems
  - Electrical systems
  - Mechanical systems
- Implementation choices
  - Discrete
  - Continuous

Reliability Allocation in Software

- A software system consists of many functional modules
  - Some reused, probably with lower defect densities
  - Some are new, with higher defect densities
  - Some are invoked more often
- To increase reliability
  - Additional testing
  - Replicated using n-version programming?
- What is the best strategy?

Optimal Reliability Allocation

- System composed of subsystems:
  - Subsystem cost a function of reliability
  - System reliability depends on subsystems
  - Failure rate as a reliability measure
- Commons systems: series and parallel
- Software system reliability
  - Fractional execution time
  - Lagrange multiplier: closed form optimal solution
  - Parameter dependence: size, defect density
- Apportionment & general approach

Colorado State University
Knowledge to Go Places

Problem Formulation

- System S has subsystems SSi, i = 1, ..n.
- Each subsystem SSi has a specific functionality (i.e. It is modeled as a Series System)
- Several choices with same functionality, but differently reliability levels.
  - $C_i = f_i(R_i)$
- Minimize system cost $C_s = \sum_{i=1}^{n} C_i = \sum_{i=1}^{n} f_i(R_i)$
- Subject to $\text{target system reliability } R_{ST}$

$$\leq \text{achieved reliability } R_s$$

Colorado State University
Knowledge to Go Places

Cost minimization problem

$$\text{Minimize} \quad C_s = \sum_{i=1}^{n} C_i = \sum_{i=1}^{n} f_i(R_i)$$

$$\text{Subject to} \quad R_{ST} \le R_s$$

$$\text{For a series system } \mathrm{R_S} = \prod_{i=1}^{n} R_i$$

$$\text{thus} \quad R_{ST} \le \prod_{i=1}^{n} R_i$$

Problem: Achieve a reliability equal to or better than the target values, while minimizing the overall cost.

Colorado State University
Knowledge to Go Places

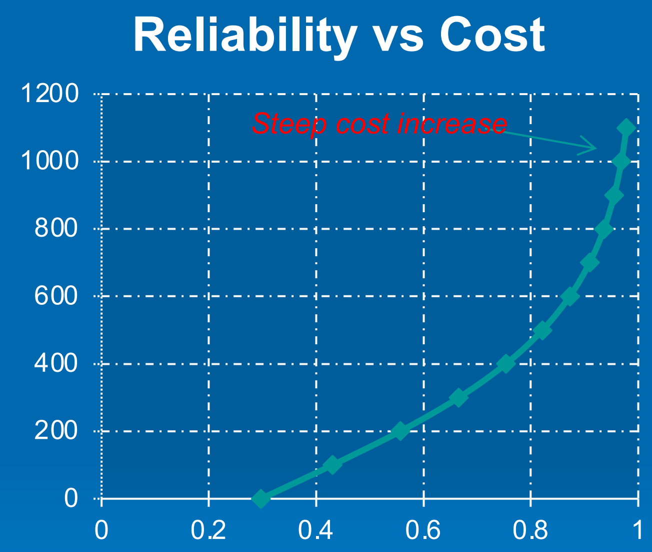
Subsystem implementation choices

- Subsystem can be made more reliable by extending a continuous attribute
  - diameter of a column in building
  - time spent for software testing.
- Different vendors implementations of SSi at different costs.
- Multiple copies of SSi to achieve higher reliability.
  - double wheels of a truck
- Number of copies is constrained between one and a practical number because of implementation issues.

The Cost function

Cost function fi should satisfy these three conditions:

- f_i is a positive function
- f_i is non-decreasing, thus higher reliability will come at a higher cost.
- f_i increases at a higher rate for higher values of R_i



Mettas A, Reliability allocation and optimization for complex systems. Pro Ann Reliability and Maintainability Symp, January 2000, 216-221

Colorado State University
Knowledge to Go Places

In terms of failure rate

- Taking log of both sides of the constraint, and since $R_i(t) = e^{-\lambda_i t}$

$$\ln(R_{ST}) \leq \sum_{i=1}^{n} \ln(R_i) \qquad \lambda_{ST} \geq \sum_{i=1}^{n} \lambda_i$$

- Stating cost as a function of failure rate

$$C_S = \sum_{i=1}^{n} C_i = \sum_{i=1}^{n} f_i(\lambda_i)$$

Colorado State University
Knowledge to Go Places

In terms of failure rate: SRGM

- exponential software reliability growth model

$$\lambda_i(d) = \lambda_{0i} \exp(-\beta_i d)$$

  - d is testing time
  - λ_{0i} depends on initial defect density
  - β_i depends inversely on program size

- Restating it as Cost function

$$d(\lambda_i) = \frac{1}{\beta_i} \ln\left(\frac{\lambda_{0i}}{\lambda_i}\right)$$

Assumes constant development cost, thus neglected

Colorado State University
Knowledge to Go Places

Series and Parallel Systems: linearlization

- Constraint Linearization simplifies the calculations.
- Series system $$\ln(R_{ST}) \le \sum_{i=1}^{n} \ln(R_i)$$
- Parallel system: log of *unreliabilities*

$$R_{ST} \le 1 - \prod_{i=1}^{n}(1 - R_i) \qquad \ln(1 - R_{ST}) \ge \sum_{i=1}^{n}(\ln(1 - R_i)$$

  - Elegbede: If cost function satisfies 3 properties given above, the cost is optimal if all parallel components have the same cost.

Colorado State University
Knowledge to Go Places

Reliability Allocation for Software Systems

- a block i is under execution for a fraction x_i of the time where $\Sigma x_i = 1$
- Reliability allocation problem

$$\text{Minimize} \quad C = \sum_{i=1}^{n} \frac{1}{\beta_i} \ln\left(\frac{\lambda_{0i}}{\lambda_i}\right)$$

$$\text{subject to } \lambda_{ST} \geq \sum_{i=1}^{n} x_i \lambda_i$$

Solution using Lagrange multiplier

- solutions for the optimal failure rates

$$\lambda_1 = \frac{\frac{\lambda_{ST}}{x_1}}{\sum_{i=1}^{n} \frac{\beta_1}{\beta_i}} \qquad \lambda_2 = \frac{\beta_1 x_1}{\beta_2 x_2} \lambda_1 \qquad \cdots \qquad \lambda_n = \frac{\beta_1 x_1}{\beta_n x_n} \lambda_1$$

- optimal values of test times d_1 and d_i, $i \neq 1$

$$d_1 = \frac{1}{\beta_1} \ln \left(\frac{\lambda_{10} x_1 \sum_{i=1}^{n} \frac{\beta_1}{\beta_i}}{\lambda_{ST}} \right) \qquad d_i = \frac{1}{\beta_i} \ln \left(\frac{\lambda_{i0} \beta_i x_i}{\lambda_1 \beta_1 x_1} \right)$$

Colorado State University
Knowledge to Go Places

Observations: Software reliability allocation

- A reused subsystem has a higher reliability because of past testing causing $\lambda_i \geq \lambda_{i0}$ and hence negative d_i.
  - Solution: apply allocation problem only to modules with positive d_i.
- If x_i is proportional to the subsystem code size, then optimal values of the post-test failure rates $\lambda_1, \ldots \lambda_n$ are equal.

Colorado State University
Knowledge to Go Places

An Illustration (next)

- Five blocks software blocks I = 1 to 5.
- Parameters β and λ_{i0} values are based on what we know about the relationship between parameters and software size, defect density.
- X_i is presumed to be proportional to software size. d_i is the additional testing time.
- Analysis using Excel Solver obtains the optimal solution: note that final λi is same for all blocks.
  - Closed form solution will yield the same result.
  - Equal testing or testing only the block with most defects will not be optimal.

Ex: Optimal: Software with 5 blocks

$\lambda_{ST} \leq 0.04$

Block	B_1	B_2	B_3	B_4	B_5
Size KSLOC	1	2	3	10	20
Ini Defect density	10	10	10	15	20
β_i	4.59×10^{-3}	2.30×10^{-3}	1.53×10^{-3}	4.59×10^{-4}	2.30×10^{-4}
λ_{i0}	0.046	0.046	0.046	0.069	0.092
x_i	0.028	0.056	0.083	0.278	0.556
Optimal λ_i	0.04	0.04	0.04	0.04	0.04
Optimal d_i	30.1	60.1	90.2	1184	3620

- Top 2 rows: problem construction, middle 3 The Problem, bottom 2 the solution.
- Observation: Optimal when all modules have the same failure rate!

Colorado State University
Knowledge to Go Places

Ex: Equal testing

$\lambda_{ST} \leq 0.04$

Block	B_1	B_2	B_3	B_4	B_5
Size KSLOC	1	2	3	10	20
Ini Defect density	10	10	10	15	20
β_i	4.59×10^{-3}	2.30×10^{-3}	1.53×10^{-3}	4.59×10^{-4}	2.30×10^{-4}
λ_{i0}	0.046	0.046	0.046	0.069	0.092
x_i	0.028	0.056	0.083	0.278	0.556
λ_i	0.146	0.003	0.01	0.08	0.15
Equal d_i	1109.4	1109.4	1109.4	1109.4	1109.4

- If Total test time is equally distributed for all 5 blocks, system will have significantly higher failure rate of 0.055 per unit time

Ex: Testing only B5

$\lambda_{ST} \leq 0.04$

Block	B_1	B_2	B_3	B_4	B_5
Size KSLOC	1	2	3	10	20
Ini Defect density	10	10	10	15	20
β_i	4.59×10^{-3}	2.30×10^{-3}	1.53×10^{-3}	4.59×10^{-4}	2.30×10^{-4}
λ_{i0}	0.046	0.046	0.046	0.069	0.092
x_i	0.028	0.056	0.083	0.278	0.556
λ_i	0.146	0.003	0.01	0.08	0.15
Equal d_i	0	0	0	0	5547

- If Total test time is allowed only for block B5, system will have higher failure rate of 0.043 per unit time

Illustration using excel

- See Excel sheet relallocationexamples.xls
- Try changing entries.

Common Apportionment rules

- Equal reliability apportionment:
  - At end they all individually have failure rate equal to target failure rate for the system
- Complexity based apportionment
  - test time apportioned in proportion to the software size
- Impact based apportionment:
  - A component executed more frequently, or more critical, should be assigned more resources

Reliability Allocation for Complex Systems

- An iterative approach
  - Design the system using functional subsystems.
  - Perform an initial apportionment of cost or reliability attributes based on suitable apportionment rules or preliminary computation.
  - Predict system reliability.
  - Is reallocation feasible and will enhance the objective function. If so, perform reallocation.
  - Repeat until optimality is achieved.
  - Does this meets objectives? If not, return to step 1 and revising the design at a higher level..

Colorado State University
Knowledge to Go Places

Conclusions

- Reliability allocation: consider how cost varies with reliability.
- Software testing:
  - cost $\propto$ log(1/failure rate)
  - $\beta 1 \propto$ size
- Reliability allocation in systems with replicated subsystems can encounter correlated failures and thus would need a more careful modeling.

Colorado State University
Knowledge to Go Places