Reliability Block Diagram - Simulate or Calculate?

Peng Liu* Rajneesh Rajneesh†

Abstract

Reliability Block Diagram is a visual interface to model a system's reliability. A system consists of multiple components, each with a different reliability characteristics, arranged according to a specific design. It is desirable to estimate a system's reliability to know its performance measure. System's reliability has been studied for decades but it is known to be computationally intensive. Even with the current computing power, the computation of a system's reliability has not improved much for complex networks; for example, telecommunication systems, electric power utility systems etc. We will discuss the system's reliability calculation using symbolic expression and Monte Carlo simulation. Both methods have their merits and demerits. In the current release of our software, we implemented the exact computation using symbolic expression. The feature to compute a system's reliability by Monte Carlo simulation is through a non-parametric distribution specification. We will present the use of that feature in this work.

Key Words: Reliability, System Reliability, Exact Calculation, Simulation, Hybrid Calculation

1. Introduction to System Reliability Computation

Reliability is a perspective, which describes the quality of some product. The measurement of reliability is often the time to an event when a product is not useable, or needs to be replaced or repaired. The time measurement is a random variable, whose nature can be described by some probability functions, such as Weibull, Lognormal, etc. The notation $R(t)$ is often used in the literature, which is the probability that the event will not occur before time t. The larger $R(t)$ is at a given time t, the more reliable that product is at that time. Those well known functions are often used, when the product of interest is treated as an indivisible entity, or a black box whose inside is not of interest. Many products can be treated in that way. For example, a cast iron part, which may break, or wear out; or a light bulb, which may burn out.

The reliability of many other products, which are usually called systems, is more understandably depended on the reliability of individual parts that aggregate the system. Furthermore, it is more interesting to know that the form of dependencies among individual parts also contributes to the reliability of the system. Describing the reliability of those system using simple probability functions, such as Weibull or Lognormal, is usually not realistic, except a handful of special cases. Many times, a closed form probability function may not even exist.

Figure 1: Series System

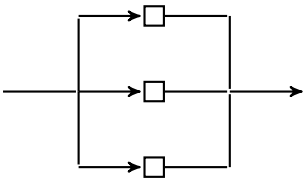

Figure 2: Parallel System

*SAS Institute Inc., 100 SAS Campus Dr, Cary 27513
†SAS Institute Inc., 100 SAS Campus Dr, Cary 27513

There are two basic types of dependence forms, series and parallel; see Figure 1 and Figure 2. The series dependence is a form that all the parts are mutually dependent, such that the failure of any individual will lead to the failure of the entire system. Therefore, the series dependence system is not more reliable than any individual parts, the reliability of which equals $\prod R_i(t)$, the product of individual reliabilities. The parallel dependence, on the other hand, is a form that all the parts are independent, such that the system will not fail, unless all individual parts fail. Therefore, the parallel dependence system is more reliability than any individual parts, the reliability of which equals $1 - \prod(1 - R_i(t))$.

2. Complexity of System Reliability Computation

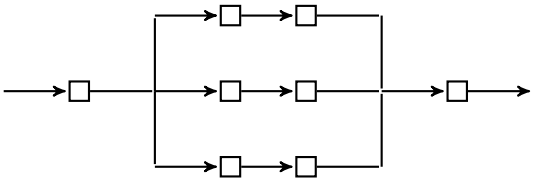

Figure 3: Series-Parallel System

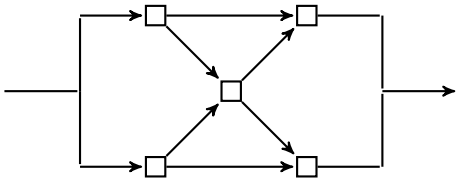

Figure 4: Bridge System

Beyond the basic forms, closed forms often exist, at least in theory, under certain conditions. To exist, the state of any part cannot depend on any randomly occurred events, such as the failure of another part. In some situations, the system is a composition of series and parallel sub-systems (Figure 3), such that the reliability function of the system is a composition of two forms. An algorithm exists to identify whether the system is decomposable in that way. After decomposing the system, the calculation of the system reliability is of the order of N, the number of parts. However, other systems cannot be decomposed into either series or parallel. A well known example is a bridge system (Figure 4). The reliability function of a bridge system cannot be written in either of the above two forms. A technique is to write the reliability function of the system as the sum of two conditional reliability functions, which are conditioned on two mutually exclusive events. For the bridge system, the two mutually exclusive events are the bridge is working or not. Conditioning on the two given events, the system becomes series-parallel decomposable. Therefore, a bridge system's complexity is of the order $2(N - 1)$. Although the analysis is elegant, it is less realistic to identify the key parts in very large systems. Meanwhile, as the number of key parts increases to make the system decomposable, the complexity order increases dramatically as well. Suppose K out of N parts are the key parts, such as the remaining system with $N - K$ parts is decomposable, the complexity order is $2^K(N - K)$, which may not be small.

When the complexity of the above computation reaches the limit of existing computational power, simulation is mostly suggested in the practice. By simulation, we mean simulating failure times of individual components, then inspect each failed component, starting from the earliest failed one, until we find the one, whose failure will cause the system to be down.

In some situations, simulation if probably the only method to solve the problem. One of such situations is when the characteristics of a component can be altered by randomly occurred events then the closed forms usually do not exist. An example is a cold standby system. In a cold standby system, all components are assembled exactly the same as a parallel dependence system. However, some components are identified as standbys, and

remain inactive, until some active component fails. When the total number of active components falls under a requirement, and there are no standbys to activate, the whole system fails. The reliability function of such a system involves integrals, which has to be evaluated numerically or by using simulation if a closed form does not exist. The computational complexity will be linear to the number of samples that one wants to simulate. However, for all simulation approaches, the curse of dimensionality is the evil. That is the dimension of the simulation is the number of components of the system. The larger the dimension of a simulation, the more samples are necessary to be drawn in order to expose the most events. Especially if some time-to-failure events are critical but rare, many simulations may be required to discover them. For example, early failures due to rare malfunction parts.

All we have mentioned above are called exact solutions to computing reliability function of a system. Even simulation gives approximate results, their discrepancy from exact solutions diminish as the simulation runs longer. In literature, there were other approaches, either also exact, or approximated but cannot be improved to increase precision. We are not considering them in this current work.

3. Considerations in Software Implementation

As we have discussed, we have two options to implement a software to compute the system reliability function exactly. One is brute force calculation, if a closed form exists. The other one is simulation if a closed form does not exist, or the brute force approach is too expensive. The brute force calculation brings ease to calculations besides the system reliability. Those calculations include the density function of the time to system failure, the hazard function of the time to system failure, the Birnbaum's component importance, and so on. All those calculations can take the advantage of the continuity and differentiability of the closed form reliability functions. On the other hand, simulation results can very well describe the time to system failure, they are not direct measurements to the quantities that those other functions need to describe. Indirect methods must be used to convert the simulation results to obtain estimates of other functions, which are certainly approximate again. We think it is graceful if user has a choice whenever possible. The following two subsections describe the frameworks that can be implemented.

3.1 A Framework of Brute Force Calculation

The framework is prepared for the worst scenario that the system is not series-parallel decomposable, and it cannot be easily identified similar to the bridge system example. For that situation, all approaches that we have seen require the computational complexity of $O(2^N)$. Approaches include conditional probability, minimum tie sets, and minimum cut sets. The conditional probability is much easier to implement, in order to obtain exact solutions. We choose that approach in our work.

The method is based on the theory that the probability that a system is working can be written as a sum of probabilities that the system is working given a specific component is working or not, denoted by the following equation.

$$\Pr[S] = \Pr[S|X_i] + \Pr[S|\overline{X}_i],$$

where S is the event that the system is working; X_i is the event that the component X_i is working; $\overline{X}_i$ is the event that the component X_i is not working. Either $\Pr[S|X_i]$ or $\Pr[S|\overline{X}_i]$ can be conditioned on another component X_j, which can be further broken down. In terms of computer science, the calculation forms a pattern of a binary tree. The data structure and algorithms associated to that methods have been maturely developed in computer science,

which will facilitate the implementation. It is worth noticing if we can observe that the system will not work under certain conditions, such as X_i is not working, the calculation does not need to break further down on that part of the branch. In that situation, the calculation is not a full binary tree, which requires less calculation than the full one. In the implementation, we expect the worst.

3.2 A Framework of Simulation

Depending on whether the states of components are mutually dependent, simulation can be extremely simple or dauntingly sophisticated. Because there are many forms of dependence in practice, in different applications, it is not possible to develop a universal framework. there are a lot of software on the market that address some of the frameworks. A brief description about a framework for simulation is described here to illustrate a big picture.

1. At time zero, simulate the life of all running components.
2. Remove the component that fails the first from the system. Test whether the system is working. If not, go to Step 5.
3. Change the states of dependent components of the failed component.
4. Simulate the life of all running components. Because some components have been working through that time point, the simulation will simulate the remaining life of those components. Go to Step 2.
5. Record the time as the system failure time. Go to Step 1, and repeat the process for desired amount of iterations.
6. Collect the simulated system failure times. Those depict the reliability of the system.

It is the third step that can be filled with boxes of tools for all kinds of dependence scenarios.

3.3 A Mixed Implementation

Our implementation is flexible to allow user to take advantages from both approaches whenever it is possible or desired. The overall structure of the implementation is under the brute force calculation, whose calculation can be optionally switched to simulation at a near future time. Indeed, any exactly calculation that is possible under brute force calculation can be optionally obtained via simulation, but not vice verse. For parts whose reliability must be calculated by simulation, we isolate the part as a subsystem, whose reliability function $R(t)$ is specified as a non-parametric distribution. The simulation that obtains that non-parametric distribution is encapsulated.

4. A Warm Standby Example

We illustrate our approach in this hypothetical example (Figure 5). This reliability block diagram models an air exhaustion system. The system has two identical sets of controls, C1 and C2, both are active. The purpose of that is to avoid unexpected electronic or mechanical difficulties, such as a malfunction chip or a broken switch on a control board. The system also has two blowers, B1 and B2. One is activated and runs under full capacity. The other one is also activated, but runs under a very low capacity, which will run at full speed if the other one fails. The reason of doing so is to reduce the stress of turning the backup

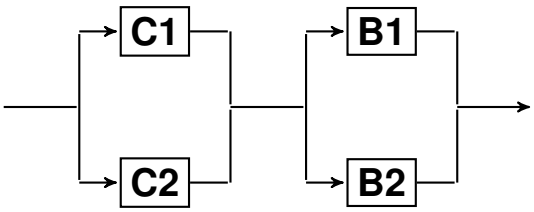

Figure 5: Two-Blower Exhaustion System

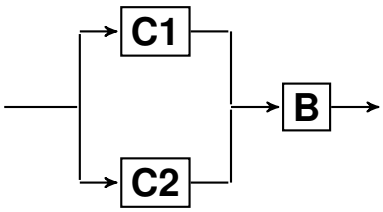

Figure 6: Two-Blower Exhaustion System

blower to full speed from zero speed, which is more likely to fail, if it is not gradually speed up. The system will not operate if both controls fail or both blowers fail. Therefore, the system can be modeled as a series diagram of controls and blowers. Two controls can be represented by a parallel sub-diagram. In this example, they are represented by a single parallel block. Two blowers cannot be trivially represented. We represent the two blowers by a single block, whose reliability function is represented by a non-parametric distribution (Figure 6).

Suppose the reliability function of a control unit is an Exponential distribution, with mean time to failure 5000 hours. The control unit is considered as a very reliable component, the failure rate of which stays constant. And the reliability function of a blower running under full speed is a Weibull with the shape parameter $\beta = 3$, and mean time to failure 2000 hours. The parametrization that we refer to is $R(t) = \exp[-(t/\eta)^{\beta}]$. Therefore, the characteristic life parameter $\eta = 2240$. The blower is considered to be more likely to fail if it continuously runs longer and longer, which can be illustrated by a hazard function plot. The reliability function of a blower running under backup speed, which is much lower, is assumed to have the same shape parameter, but much longer mean time to failure, which is 10000 hours. That means the characteristic life parameter $\eta = 11200$.

The reliability function of the whole system is $(1 - (1 - R_C(t))^2) \times R_S(t)$, where $R_C(t)$ is the reliability function for a control unit, and $R_S(t)$ is the reliability function of the sub-system of two blowers. The calculation of $R_S(t)$ is described by the following equations.

$$\begin{aligned} R_S(t) &= \Pr[T \geq t] && (1)\\ &= \Pr[T_1 \geq t] + \Pr[T_1 < t \text{ and } T_2 \geq t] && (2)\\ &= R(t) + \int_0^t f(\tau)\Pr[T_2 \geq t|\tau]d\tau, && (3) \end{aligned}$$

where $R(\cdot)$ is the reliability function of the time to the failure of the running blower; $f(\cdot)$ is the density function of that event. Moreover, the part $\Pr[T_2 \geq t|\tau]$ inside of the integrand is conditioning on τ, the time to the failure of the first blower. The computation of $\Pr[T_2 \geq t|\tau]$ is not trivial. We give its expression here without further explanation.

$$\Pr[T_2 \geq t|\tau] = R(t - \tau + q(1 - R^*(\tau))),$$

where $q(\cdot)$ is the percentile function of the time to the failure of a running blower; $R^*(\cdot)$ is the reliability function of the time to failure of the standby blower under standby condition. The calculation of above expression can be done by a numerical integration, which is only feasible for some problems. The calculation can also be done by Monte Carlo integration, or simulation in another word. Simulation is more general, and universally feasible, as long as the computational resources allows. A sample of time to failure of the above two-blower subsystem can be done using the following steps:

1. Generate a random sample t_1 from Weibull$(x; \eta = 2240, \beta = 3)$ and a random sample t_2 from Weibull$(x; \eta = 11200, \beta = 3)$.

2. If $t_1 \geq t_2$, let $t = t_1$ be the time to the failure of the subsystem, then go to previous step. Otherwise, go to next step.

3. Compute the probability that the standby blower has survived through t_1, which is $1 - R^*(t_1)$.

4. Compute the equivalent life of the standby blower has been through if it ran under normal stress, which is $t^* = q(1 - R^*(t_1))$.

5. Generate a random sample t_2 from conditional distribution Weibull$(x; \eta = 11200, \beta = 3|x > t^*)$. Let $t = t_2$ be the time to the failure of the subsystem. Go to the first step and repeat the process.

5. Conclusion

We explored the choices of calculating system reliabilities. To achieve the precision, we can either use the brute force calculation, or simulation. We only have to use simulation if the closed form formula does not exist. We may prefer simulation, if the brute force calculation is too expensive. The choice between brute force and simulation does not need to be at the whole system level. A sub-system can be isolated to simulate, and weaved into the bigger system by deriving proper functions from the simulation results using a nonparametric distribution.

REFERENCES

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2008) "Introductions to Algorithms," *MIT Press*, Cambridge, MA.

Elsayed, E. (1996), "Reliability Engineering," *Addison Wesley*, Reading, MA.

Meeker, W. Q., and Escobar, L. A. (1998) "Statistical Methods for Reliability Data," *John Wiley & Sons, Inc.*, New York.

Nelson, W. (1982) "Applied Life Data Analysis," *John Wiley & Sons, Inc.*, New York.

Tobias, P. A., and Trindade, D. C. (2012) "Applied Reliability," *Chapman & Hall/CRC*.

West, D. B. (2006) "Introduction to Graph Theory," *Pearson Education, Inc.*, New Jersey.