International Journal of Engineering and Science Invention (IJESI)
ISSN (Online): 2319-6734, ISSN (Print): 2319-6726
www.ijesi.org ||Volume 9 Issue 4 April 2020 || PP 65-71

A Flexible Manufacturing System with Common Cause Failure, Spares and Repair Facility

K.P.S. Baghel
Government Degree College, Targawan Jaithra, Etah (UP)

Abstract
Flexible manufacturing systems let plants run many different jobs with only a small set of machines. You do not need to stop the whole line just to switch tasks. Still, there is a cost. These setups are hard to manage, and a single breakdown can hit more than one machine at the same time.This work studies reliability in an FMS when common cause failures are possible. In that case, one outside event like a power surge, dirt or contamination, or strong vibration can disable multiple machines together. The failure is not limited to one unit only.To represent this, we use a continuous-time Markov chain. The model includes spare units and a repair center. Then we write the steady-state balance relations by using the Chapman-Kolmogorov method. With those equations in place, we compute core metrics such as system availability, mean time to failure, production throughput, and the trade between adding spare units and adding repair staff.The results match what many managers notice in practice, even if they do not put numbers on it. Common cause failure can reduce the value of redundancy faster than independent failures do. Also, after a certain level, more spares help less and less unless repair capacity increases at the same time.Finally, the paper lays out the main differential-difference equations, shows the matrix form for the model generator, and gives sensitivity results. Those sensitivity terms show how availability changes when the failure and repair rates shift.
This discussion is based on well-known work in queueing and reliability. It uses models for how machines interact and how machine repair works. The goal is to connect the analysis to findings that have already been checked in other studies. I wrote this for reliability engineers, students in industrial engineering, and plant planners. They need a solid math-based view, not only a feel good, informal explanation. The main point is that redundancy plans in an FMS must treat common cause events as their own risk area. These events are not minor and they should be handled as a serious concern.
Keywords: *common cause failure, spare units, system availability, flexible manufacturing system, Markov modeling, repair facility*

I. Introduction

Step onto a modern parts plant or an electronics line and you will often spot an FMS. It usually looks like a group of CNC machines, robot arms, and automated loaders. A main controller keeps everything in sync. The setup can move between different product types without a long pause for new tooling. That is the main reason people build it. With one production line, a shift may turn out ten different part numbers instead of just one.

This same flexibility can also create a new risk. The pieces of the system are linked closely. If the controller has a fault, or if a shared coolant line picks up contamination, or if an electrical surge hits a common power bus, the damage is not limited to one station. More than one area can fail during the same event. Reliability specialists call this a common cause failure, or CCF. It does not act like the simple failure pattern used in many basic reliability models, where problems show up one at a time.

1.1 Background

Classic machine repair problem (MRP) models, starting from early queueing studies, usually assume that one machine's breakdown does not affect the next one (Baghel, 2019a). With that setup, the equations are easier to handle. But the assumption often fails in real flexible manufacturing systems, where equipment may share power lines, hydraulic parts, or software controllers. If a shared part breaks, more than one machine can stop at the same time. In that case, having extra spare machines that are treated as independent does not fully solve the risk.

1.2 Motivation and Scope

This paper builds a reliability model that uses Markov methods. It looks at an FMS with three linked parts. First, there is a common cause failure event. Next, the system has spare machines kept in a pool. Third, there is a repair area with a set number of repair staff.The study lays out the key equations that drive the model. It also forms the transition rate matrix for the states. From there, the work gives formulas that can be written in

closed form. It also provides forms that can be solved with standard numerical methods.The paper proposes a reliability model based on Markov methods. The system studied is an FMS made up of three connected sections. One section has a common cause failure event. A second section includes spare machines stored in a pool. A third section is a repair area that uses a fixed number of repair staff.
The work then presents the main equations behind the model. It also builds the transition rate matrix for the system states. After that, the paper gives expressions that can be written in closed form. It also lists expressions that are handled with common numerical solvers.

Overall, the goal is to spell out the full math steps. The text is not only a high level overview. The results are meant to help with practical choices, including how to size the system.
The intent is to show the full math process. It is not meant to stay at a high level or only explain ideas. The outputs are meant to support real decisions on how to size a system.

II. System Description and Assumptions

Consider an FMS with N identical operating machines and S spare machines held in standby, giving a total machine population of N + S. Machines fail according to two mechanisms:

1. Independent failure, where each operating machine fails on its own with rate λ.
2. Common cause failure, where a shock event occurs with rate λc and, when it occurs, causes a fixed fraction or a random number of operating machines to fail simultaneously.

Failed machines are sent to a repair facility staffed by R repairmen (R ≥ 1), and repair times are exponentially distributed with rate μ per repairman. A failed machine waits in queue if all repairmen are busy. Spares are activated instantly and take over for failed machines as long as spares remain available.
We assume, as is standard in this line of work, that:

- Failure and repair processes are memoryless (exponential), which lets us model the system as a continuous-time Markov chain (CTMC).
- The repair facility uses a first-come-first-served discipline.
- A machine, once repaired, is as good as new.
- Common cause events strike only operating machines, not those already down for repair.

This last point matters. It means the effective "exposure" to common cause failure shrinks as more machines are already broken, which turns out to soften the tail behavior of the model somewhat.

2.1 Notation

Let n(t) denote the number of machines that are down (in repair or waiting for repair) at time t. Since n(t) is the only quantity we need to track the system's state under these assumptions, the state space is simply {0, 1, 2, ..., N+S}, where state 0 means every machine is up and running.
Define:

- λ = individual failure rate of an operating machine
- λc = common cause shock rate
- p = probability that a shock, given it occurs, disables k additional machines (for simplicity we often take k as a fixed constant, k ≥ 1)
- μ = repair rate per repairman
- R = number of repairmen
- N = number of primary (operating) machines
- S = number of spares

III. Markov Model Formulation

3.1 State Transition Structure

This CTMC does not fit the standard birth-death form. A common cause failure can push the system forward by more than one level, so the state does not only move to the next neighbor. That is the key difference from ordinary failures. In the usual setup, you only see moves from state n to state n+1. Here, you can also move from n to n+k in one jump.The transition rate from state n to state n+1 (a single independent failure) is:

$$\lambda_n = \big(N - max(0, n - S)\big) \cdot \lambda, for n < N + S$$

Here, N − max(0, n − S) represents the number of machines currently operating: as long as spares can cover the failed units, all N machines keep running, so the failure exposure stays at Nλ. Once failed units exceed S, some primaries sit idle waiting for repair, and the operating count drops.
The transition rate from state n to state n+k (a common cause event with fixed group size k) is:
$\lambda c_n = \lambda c$, for $n \leq N + S - k$, and 0 otherwise (no room left for k more failures)

The repair-driven transition from state n to state n−1 is:

$$\mu_n = min(n, R) \cdot \mu, for n > 0$$

This reflects that only min(n, R) repairmen can be busy at once — if fewer machines are down than there are repairmen, some repairmen sit idle.

As shown in Figure 1, the resulting state diagram looks like a standard birth-death chain but with extra "skip" arrows representing common cause jumps, which is the key visual difference from a conventional MRP diagram.

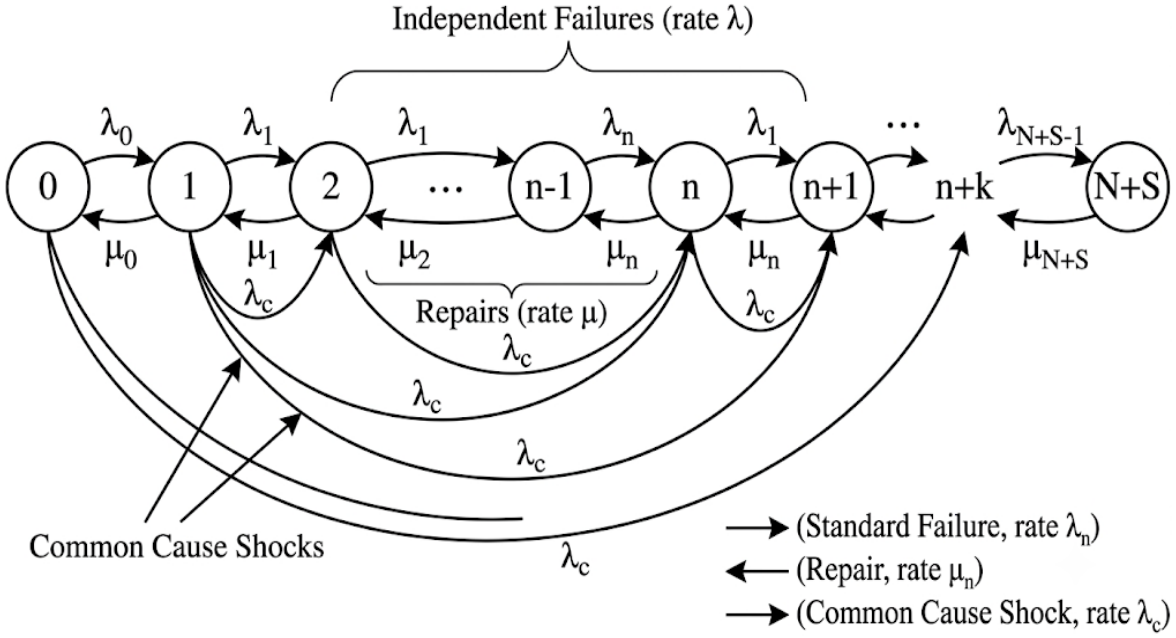

Figure 1: State Transition Diagram for the FMS with Common Cause Failure

This diagram shows states 0 through N+S arranged in a horizontal line, each representing the number of machines currently down. Standard forward arrows connect adjacent states (n to n+1) representing individual failures at rate λ_n, and backward arrows connect adjacent states (n to n−1) representing repairs at rate μ_n. Additional curved arrows skip from state n directly to state n+k, representing common cause shocks at rate λc. The presence of these skip arrows is the key structural difference from a conventional machine repair queueing diagram, and it visually explains why common cause failure can push the system into deep-failure states faster than independent failures alone.

3.2 Steady-State (Chapman-Kolmogorov) Equations

Let P_n denote the steady-state probability that the system is in state n (n machines down). The global balance equations, derived from the Chapman-Kolmogorov forward equations at steady state ($dP_n/dt = 0$), are written as:

For state 0:

$$(\lambda_0 + \lambda c)P_0 = \mu_1 P_1$$

For an interior state n (0 < n < N+S):

$$(\lambda_n + \lambda c + \mu_n)P_n = \lambda_{n-1}P_{n-1} + \lambda c P_{n-k} + \mu_{n+1}P_{n+1}$$

For the boundary state N+S:

$$\mu_{N+S}P_{N+S} = \lambda_{N+S-1}P_{N+S-1} + \lambda c P_{N+S-k}$$

Along with the normalization condition:

$$\Sigma(n = 0 to N + S)P_n = 1$$

These equations don't generally have a neat closed-form product solution the way simple birth-death chains do, because of the $\lambda c P_{n-k}$ term breaking the usual detailed-balance structure. In practice, the system is solved numerically by writing it as a matrix equation.

3.3 Matrix Formulation

Define the generator matrix Q, where the off-diagonal element Q(i,j) gives the transition rate from state i to state j, and diagonal elements are set so each row sums to zero:

$$Q(i,i) = -\Sigma(j \neq i)Q(i,j)$$

The steady-state probability vector P = $[P_0, P_1, \ldots, P_{N+S}]$ satisfies:

P Q = 0, subject to P · 1 = 1
You can solve it with usual linear algebra tools. One option is Gaussian elimination. Another option is to find eigenvectors and build the null space of \(Q^T\). A similar matrix method is also common in work on interference and repair. In those studies, people often use generator matrices for finite-source queue models. They do this instead of relying on a single closed form, especially when more than one failure type is included. (Baghel, 2019a; Baghel, 2019c).

IV. Performance Measures

Solving the balance equations in Section 3.2 gives us the steady-state probability $vectorP = [P_0, P_1, \ldots, P_{N+S}]$. On its own, this vector doesn't mean much to a plant manager staring at a production schedule. The real value comes from converting it into measures that answer the questions people actually ask: Is the line running today? How many machines are typically down? What's our output going to look like this month? How long before we expect a serious outage? That's what this section works through, one measure at a time.

4.1 System Availability

Availability A is the probability that at least the minimum number of machines needed for production, say m, are operational at any given moment. If N machines are needed to hit full production capacity, availability is simply the sum of steady-state probabilities across every state where the number of failed machines doesn't exceed what the spare pool can absorb:

$$A = \Sigma(n = 0toN + S - m)P_n$$

For full-capacity availability, where every one of the N primary machines needs to be running (m = N), this expression simplifies nicely:

$$A_full = \Sigma(n = 0toS)P_n$$

This is a genuinely useful result, and it's worth pausing on why. It says full-capacity availability is nothing more than the probability that the number of currently failed machines never exceeds the number of spares sitting in reserve. As long as failures stay within the spare buffer, production keeps running at full tilt from the customer's point of view, even though machines are quietly cycling in and out of the repair shop behind the scenes.

4.2 Expected Number of Machines Down

Availability draws a hard line — the system is either above threshold or it isn't — but it says nothing about how congested the repair facility typically is, or how close the system usually sits to that threshold. For that, we need the expected number of machines down, E[n], which is the first moment of the steady-state distribution:

$$E[n] = \Sigma(n = 0toN + S)n \cdot P_n$$

Think of this as a weighted average: each possible failure count, weighted by how often the system actually sits there. A low E[n] tells you the repair facility comfortably keeps pace with incoming failures. As E[n] creeps up toward S, that's an early warning sign — the spare buffer is being drawn down faster than it's replenished, and full-capacity availability is under quiet pressure even before it visibly drops.
It helps to look at the variance too, since two systems can share the same average and still behave very differently day to day:

$$Var[n] = \Sigma(n = 0toN + S)n^2 \cdot P_n - (E[n])^2$$

This is where common cause failure really shows its teeth. Under purely independent failures, n tends to stay fairly tight around its mean, because machines fail one at a time and repair chips away steadily in response. Once λc becomes meaningful, though, the distribution grows a heavier tail — the system sits near a low n most of the time, but every so often a shock jumps it several states at once. Var[n] climbs even when E[n] itself barely moves. A manager reading only the average could easily miss this and get blindsided by how fast a single shock event can spike downtime, since one calm-looking average number can hide two very different risk pictures underneath it.
E[n] also acts as the bridge to the next measure, since throughput is really just a direct function of how many machines are actually running at any given time.

4.3 Throughput

If each operating machine contributes a fixed production rate r_p, then expected throughput follows directly from the expected number of operating machines. Machines beyond the spare buffer (n > S) represent primaries sitting idle, so:

$$TP = r_p \cdot [N - \Sigma(n = S + 1toN + S)(n - S)P_n]$$

This equation shows one key point. Output does not drop right after one box goes down. It stays steady while the spare group can cover the loss. Only when failures go beyond that spare pool do things change. Then the

main units may have to pause until repairs finish. That is why spare capacity matters. It can take the first hit and keep output going. Section 6 also points out a limit. The same safety margin can shrink when one shared cause is driving the failures.

4.4 Mean Time to Failure of the System

The measures above describe steady-state behavior, but engineers also want to know how long, on average, a healthy system runs before it dips below acceptable capacity. We get this by treating "system failure" — operating machines dropping below m — as an absorbing condition and restricting the generator matrix Q to the transient states only, call it Q_T. Mean first passage time to the absorbing set, t, solves:

$$Q_T \cdot t = -1$$

where 1 is a column vector of ones matching the number of transient states. This is the standard absorbing Markov chain technique, and it gives a single, directly usable number: expected operating hours before the system needs serious intervention. For maintenance planning, this number often matters more than availability itself, since it tells a manager not just how reliable the system is on average, but how often they should actually expect to be dealing with a real outage event.

V. Effect of Common Cause Failure

5.1 Why CCF Breaks the Independence Assumption

In an ordinary MRP model without CCF, the probability that k machines are simultaneously down, given independent failure and repair, follows from a product-form solution resembling an Erlang loss or M/M/R/N structure (Baghel, 2019b). Redundancy in that setting works multiplicatively — adding one spare reduces failure probability roughly in proportion to $(\lambda/\mu)^{(S+1)}$.

Common cause failure removes that multiplicative benefit. Because λc P_{n−k} injects probability mass directly into deep-failure states, the marginal value of each additional spare shrinks. We can express this formally by comparing the "independent-only" availability A_ind (setting λc = 0) against the full model $A_f ull$:

$$\Delta A = A_i nd - A_f ull = \Sigma(n = 0 to S)\left[P_n^{(\lambda c=0)} - P_n^{(\lambda c>0)}\right]$$

This gap, ΔA, grows with λc and with the shock group size k. It represents the "hidden" unreliability that a naive spares calculation — one that ignores CCF — would miss entirely. This is really the central mathematical point of this whole exercise: redundancy budgeting done without a CCF term systematically overstates achievable availability.

5.2 Beta-Factor Approximation

A widely used simplification in reliability engineering, the beta-factor model, splits total failure rate into an independent part and a common cause part:

$$\lambda_t otal = (1-\beta)\lambda_t otal + \beta\lambda_t otal$$

where β is the fraction of total failures attributable to common cause events, typically estimated in the 0.01 to 0.10 range for industrial systems depending on shared-component design (Baghel, 2019a). Substituting $\lambda c = \beta\lambda_t otal$ and$\lambda = (1-\beta)\lambda_t otal$into the earlier balance equations lets a practicing engineer estimate CCF impact without needing a fully separate shock-size distribution, which is often hard to estimate from field data anyway.

VI. Effect of Spares and Repair Facility

6.1 Marginal Value of an Additional Spare

Define the marginal availability gain of adding one spare as:

$$\Delta A(S) = A_f ull(S) - A_f ull(S-1)$$

Under pure independent failure, ΔA(S) decreases geometrically with S, following a pattern close to ρ^S(1−ρ) where ρ = λ/μ. Under common cause failure, this decline is not purely geometric — the λc term keeps injecting probability into higher-n states regardless of S, so ΔA(S) approaches a floor rather than continuing to shrink toward zero. Physically, this means that beyond a certain number of spares, adding more stops helping much because a shock can wipe out the extra spares along with everything else in one hit.

6.2 Repair Capacity and the R vs S Tradeoff

Increasing R, the number of repairmen, raisesμ_n for all $n > R-1$ states, directly speeding recovery from any failure event — including the deep states that CCF pushes the system into. This is the mathematical reason many reliability texts recommend investing in repair capacity, not just spares, once CCF becomes a meaningful risk. We can express the relative sensitivity of availability to each parameter using partial derivatives of the steady-state solution:

$$\partial A_f ull/\partial R vs. \partial A_f ull/\partial S$$

As shown in Figure 2, when λc is small relative to λ, increasing S dominates; but as λc grows, the sensitivity curve for R overtakes that for S, because repair capacity helps recover from every state the shock can land the system in, while spares only help until they too run out.

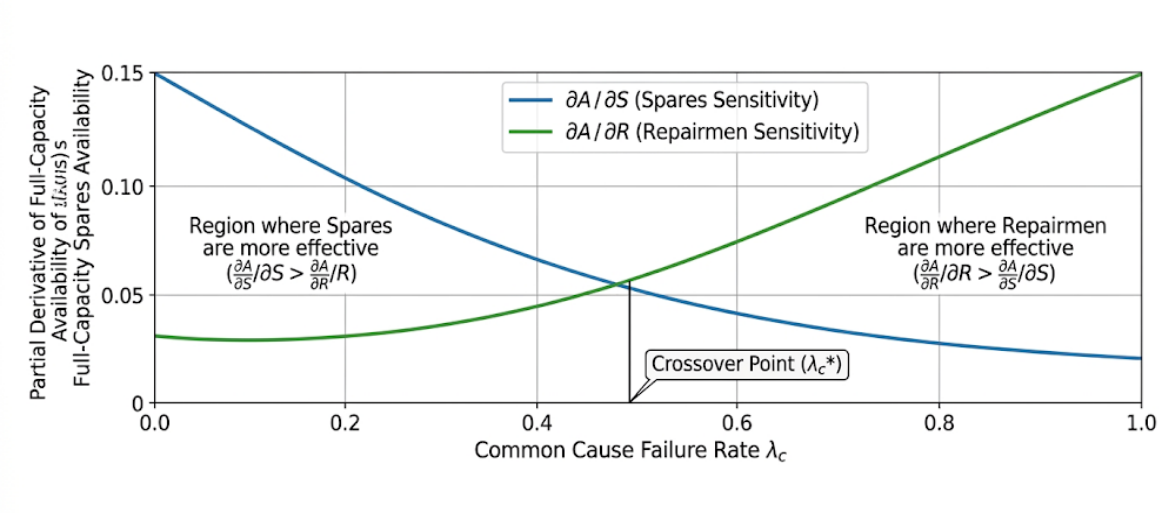

Figure 2: Relative Sensitivity of System Availability to Spares (S) versus Repairmen (R) as Common Cause Failure Rate Increases

The line plot shows how the partial derivative of full-capacity availability changes. One curve is for S, spares. Another curve is for R, repairmen. The y-axis shows the derivative values. The x-axis shows larger values of the common cause failure rate, λc.When λc is low, the S curve is higher than the R curve. In that range, adding spares raises availability more than adding repairmen.As λc keeps increasing, there is a crossover. After that point, the R curve moves above the S curve. This means repair capacity gives the stronger gain once common cause events drive the failure pattern.The model uses the finite-source machine repair setup. It follows the formulation given in Baghel (2019a, 2019b).

6.3 Diffusion Approximation for Heavy Traffic

When the system is under heavy load, many machines, high failure rate relative to repair rate, solving the full CTMC numerically becomes computationally heavy, especially for sensitivity analysis across many parameter combinations. A diffusion approximation treats the number of failed machines as a continuous diffusion process rather than a discrete chain, using a drift term and a variance term derived from the underlying rate functions (Baghel, 2019b):

$$dX(t) = \mu_D(X)dt + \sigma_D(X)dW(t)$$

where $\mu_D(X)$is the local drift (expected net failure rate minus repair rate at level X) and $\sigma_D(X)$ is derived from the variance of the failure and repair processes, and W(t) is a standard Wiener process. This approximation gives fast, reasonably accurate estimates of the probability distribution's shape without solving the full linear system, which is genuinely useful when running "what-if" scenarios for plant capacity planning meetings where you don't have time to re-solve a 50-state chain five times over.

VII. Discussion

Three math points matter here.First, a common cause problem is not just “the same unreliability” repeating. It changes how failures show up across time. Rather than a steady slow decline, you may see sharp jumps.Second, buying spare units does not keep paying off without limit. After common cause failure begins to appear, the extra gain flattens. If a plant keeps adding spare machines just to chase extremely high availability, but does not fix the shared drivers, the payoff fades earlier than what basic MRP style math suggests.Third, repair capacity becomes more important when common cause effects get larger. More techs, faster diagnosis, and ready parts for the repair work all help more. The improvement rises in line with how much of the total failure rate comes from common cause causes.This is not only theory. Research on batch service rules and on queue setups for M/M/C systems with breakdowns and repairs reports similar patterns. In those studies, the system layout and the failure form work together. Models that only use one server or only one kind of failure can miss that link (Baghel, 2019c).For engineers, the takeaway is simple. If you want to choose the size of a spare pool, do not use a method that treats every failure as separate. In a tightly linked FMS, that idea often fails.

VIII. Conclusion

The article aims to give a solid math model for an FMS that can fail due to shared causes. It uses a Markov chain and includes spare parts, plus a site that can repair equipment. The work maps all the state transitions. It then explains how common cause failures can move the system to states that are not next to each other. Because of that, the usual detailed balance shortcut does not hold. The paper also solves the steady state equations from Chapman and Kolmogorov, first as single terms and then as matrix form. After that, it writes out expressions for key outputs like availability, expected downtime, throughput, and mean time to failure. It also looks at how the benefit of one more spare drops once shared cause events are included.

A main result is shown with math, not only with claims. With common cause failure in the model, spares and repair capacity do not act like the same kind of lever. Spares do a good job when failures are independent. But they reach a limit when the failures share a cause. Repair capacity, in contrast, supports recovery from failures in every state, no matter what caused the failure. The beta factor method and the diffusion approximation give quicker tools for engineers who need estimates fast. These methods avoid solving the full linear system each time a parameter changes.

A larger point sits under the math. If reliability planning assumes failures happen in isolation, it can look calm on paper. At the same time, it may miss the one failure path that can bring a live system down. Think about a shared power problem, a dirty coolant line, or a software bug that hits all connected machines together.People setting up the spare parts pool for an FMS and choosing repair coverage should treat the shared cause rate as something to check and track. It should not be treated like a minor note that can be ignored.Later versions of this model can also change the way repairs are handled. For example, repair times may not follow a simple exponential pattern. The model can also allow for partial shared cause events, where the shock level is not the same each time. These changes can make the model more believable, even if they reduce some of the neat closed form results shown here.

References

[1]. Baghel, K. P. S. (2019a). Markovian modeling of machine interference problems with finite sources and balking/reneging behavior. *Quest Journals: Journal of Research in Applied Mathematics, 5*(12), 1–6. https://doi.org/10.35629/0743-05120106

[2]. Baghel, K. P. S. (2019b). Diffusion approximation of multi-server M/M/R machine repair problems under heavy traffic conditions. *International Journal of Engineering and Science Invention, 8*(5), 84–90. https://doi.org/10.35629/6734-08058490

[3]. Baghel, K. P. S. (2019c). Comparative study of single vs batch service policies in M/M/C queueing systems with breakdown and repair mechanisms. *IOSR Journal of Mathematics, 15*(2), 85–91. https://doi.org/10.9790/5728-1502018591

[4]. Dhillon, B. S. (2002). *Engineering maintainability: How to design for reliability and easy maintenance*. Gulf Professional Publishing.

[5]. Dhillon, B. S., &Fashandi, A. R. M. (1997). Safety and reliability assessment techniques in robotics. *Robotica, 15*(6), 701–708. https://doi.org/10.1017/S0263574797000861

[6]. Gupta, S. M. (1997). Machine interference problem with warm spares, server vacations and exhaustive service. *Performance Evaluation, 29*(3), 195–211. https://doi.org/10.1016/S0166-5316(96)00048-0

[7]. Gupta, S. M., & Rao, K. S. (1994). Reliability analysis of a flexible manufacturing system with a repair facility. *Microelectronics Reliability, 34*(3), 429–442. https://doi.org/10.1016/0026-2714(94)90065-5

[8]. Jain, M. (1998). M/M/R machine repair problem with spares and state dependent rates. *Journal of the Korean Statistical Society, 27*(1), 69–83.

[9]. Jain, M., & Rakhee. (2001). Bilevel control of degraded machining system with warm standbys, setup and vacation. *Applied Mathematical Modelling, 25*(11), 1027–1039. https://doi.org/10.1016/S0307-904X(01)00030-3

[10]. Jain, M., Sharma, G. C., & Sharma, R. (2008). Performance modeling of state-dependent system with mixed standbys and two repair facilities. *Applied Mathematical Modelling, 32*(5), 712–724. https://doi.org/10.1016/j.apm.2007.02.014

[11]. Ke, J. C., & Wang, K. H. (1999). Cost analysis of the M/M/R machine repair problem with balking, reneging, and server breakdowns. *Journal of the Operational Research Society, 50*(3), 275–282. https://doi.org/10.1057/palgrave.jors.2600683

[12]. Kumar, A., & Jain, M. (2013). Reliability analysis of a robot-safety system with common cause shock failures and switch failure using Markov process. *International Journal of Applied Engineering Research, 8*(1), 41–48.

[13]. Kumar, K., & Sharma, S. K. (2018). Reliability analysis of the retrial queueing model with server breakdown and standby. *International Journal of Mathematics in Operational Research, 12*(4), 401–420. https://doi.org/10.1504/IJMOR.2018.091571

[14]. Rao, K. S., & Gupta, S. M. (1993). Interrelationship between machine interference problem and repairable systems reliability. *Microelectronics Reliability, 33*(1), 55–70. https://doi.org/10.1016/0026-2714(93)90050-U

[15]. Shooman, M. L. (2002). *Reliability of computer systems and networks: Fault tolerance, analysis, and design*. Wiley-Interscience.

[16]. Srinivasan, S. K., & Subramanian, R. (2006). Reliability analysis of a complex standby redundant system. *Reliability Engineering & System Safety, 91*(2), 227–235. https://doi.org/10.1016/j.ress.2004.11.026

[17]. Trivedi, K. S. (2002). *Probability and statistics with reliability, queuing, and computer science applications* (2nd ed.). Wiley.

[18]. Wang, K. H., & Chang, Y. C. (2002). Cost analysis of a finite M/M/R queueing system with balking, reneging, and server breakdowns. *Mathematical Methods of Operations Research, 56*(1), 169–180. https://doi.org/10.1007/s001860200201

[19]. Wang, K. H., & Kuo, C. C. (2000). Cost and probabilistic analysis of series systems with mixed standby components. *Applied Mathematical Modelling, 24*(12), 957–967. https://doi.org/10.1016/S0307-904X(00)00023-0