International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 01 | Jan - 2025 SJIF Rating: 8.448 ISSN: 2582-3930

Reinforcement learning resource allocation optimisation for network reliability

Anup Kumar Yadav[1], Kalpna Sachan[2], Harimohan Soni[3]

[1]Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.
[2]Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.
[3]Mechanical Engineering Department, Bansal Institute of Science and Technology, Bhopal (M.P.), India.

---***---

Abstract - The power grid, telecommunications, water supply, and transportation are all examples of networks that provide vital societal functions but are vulnerable to disruption. Based on this, we take a look at a sequential choice problem where an initial network is improved over time (for example, by adding or enhancing edge dependability) and incentives are added depending on the network's all-terminal reliability. Time, money, and labour are all limited resources that determine what can be done within each time frame. To fix this, we used a Deep Reinforcement Learning (DRL) approach in OpenAI-Gym with Stable Baselines. Based on the current state of the network and the budget, a Proximal Policy Optimisation (PPO) method was used to identify the edge that needed improvement or to incorporate a new edge. The network's all-terminal dependability was calculated using a reliability polynomial. We evaluated numerous network configurations with different starting link reliability, additional link reliability, node amounts, and budget frameworks to understand the model's behaviour in different conditions. Last but not least, we go over the key points from our set of controlled tests.

Key Words: Deep reinforcement learning, proximal policy optimisation,reliability, PPO.

1.INTRODUCTION

In every aspect of our life, we make use of many networks. It is essential to ensure that networks function properly in order to ensure that everyone is able to go about their daily lives. As an illustration, people make use of a network of roads in order to drive, a network of telephones in order to communicate with one another, and a network of electrical outlets in order to power elements within their homes. All of these are only some of the major networks that are relied on by everyone worldwide. In the event that they cannot be trusted, a great number of people will suffer harm.

A great number of people have had difficulties as a result of the numerous infrastructure failures that have occurred in the past. The blackout that occurred in the Northeast in 2003 is an example that is well-known. There were fifty million people in the United States and Canada who were affected by this blackout, which was caused by a malfunction with the software, as stated by History.com [13]. During Hurricane Katrina, the levees in Louisiana failed, which is another example of infrastructure networks failing. The accident that occurred at Three Mile Island in Pennsylvania is another example of infrastructure networks failing. The disaster that occurred at Three Mile Island was caused by a fault that occurred in a particular component of the plant. This issue caused the entire system to become inoperable. According to the United States Nuclear Regulatory Commission [44], this malfunctioning component was the cause of a partial nuclear meltdown that impacted at least tens of thousands of people living in the vicinity of the plant. According to Pruitt [34], the levees in Louisiana were not prepared to handle the overwhelming volume of water that Hurricane Katrina brought. As a consequence of the pressure, they broke, which resulted in a significant amount of New Orleans being flooded. Simply put, these are only a handful of the many reasons why the infrastructure of the United States of America frequently fails. People who are affected by infrastructure breakdowns are in a great deal of suffering because these failures can occasionally result in serious injuries or even death. It is made abundantly evident by these examples how essential it is to maintain the security of crucial networks.

At its most fundamental level, a network is nothing more than a collection of nodes that are connected to one another via edges. Numerous objects that we use on a daily basis are connected to one another in the form of a network. Table 1 offers a visual representation of the many categories of networks that Newman [32] discusses.

Due to the massive size of the networks, determining their reliability is very difficult. The difficulty of conducting efficient analysis grows in direct proportion to the size of the network. Assessing network reliability efficiently is beyond the capabilities of current approaches. There have been previous investigations into the difficulties of building trustworthy networks as well. Because it necessitates ongoing evaluation of network dependability, this issue is more complex.

2. Dependability of Networks

Networks can be classified in various ways, aiding in the identification of those warranting further examination. Ball [4] addresses the many levels of complexity in network dependability models. The level of complexity is significantly contingent upon the quality of the network's connectivity. A network dependability issue may be classified as 2-terminal, k-terminal, or all-terminal. An exception to the k-terminal problem is present in the two-terminal and all-terminal cases. In the k-terminal reliability problem, there exists a single root node (s) and k terminal nodes. The probability that each node in the k-terminal system is connected to the root node generally represents the system's reliability. Reliability in a two-terminal reliability model is defined as the probability that the two nodes in the system are connected. This model consists of only two nodes. A fully connected network has each node linked to every other node. Reliability is the probability of a fully interconnected system. The computational complexity of determining the system's dependability escalates with the rise in the number of nodes and links in the network, with all-terminal reliability issues being the most challenging.

The computational complexity of determining the dependability of an all-terminal network escalates exponentially with the number of interconnected nodes and links, as identified by Provan & Ball [33] as a #P-complete problem. A multitude of methods for assessing network reliability have been utilised to facilitate the analysis of this problem. These strategies can provide an exact figure or a general assessment of reliability. Artificial neural networks exemplify an approach using boundary computation, optimisation, exponential and polynomial time methods, state enumeration, and Monte Carlo simulation. This section explores the mechanisms of these approaches and analyses their previous studies to delineate their strengths and drawbacks.

Ball et al. [5] delineates precise methodologies for computing network reliability, encompassing algorithms that operate in polynomial time for certain network classes and those that function in exponential time for broad networks. They additionally address alternative techniques, including constraints on network reliability and Monte Carlo simulation. Gaur et al. [15] not only delineated the constraints of several network dependability methodologies but also offered comprehensive descriptions of several techniques, including neural networks, state enumeration, and minimal cut enumeration.

A minimal cut set is a collection of system components whose failure results in the failure of the system (Su et al., [42]). To prevent system failure, minimum cut sets exclude any supplementary subsets of cuts. To assess the network's reliability, minimal cut enumeration procedures initially calculate the reliability of each minimum cut set and subsequently utilise these sums to ascertain the network's dependability. A definitive method to assess a network's reliability is by cut enumeration. Although it functions effectively for smaller networks, its limitations become evident rapidly. According to Gaur et al. [15], cut enumeration is the preferred way for addressing reliability issues between two terminals. The number of cut sets increases exponentially as the network expands. Consequently, calculating all possible combinations for two-, k-, and all-terminal dependability is laborious. To assess the network's reliability, defined as the proportion of states in which the network functions correctly, Monte Carlo simulation (MCS) methods randomly selected states for analysis.

Karger [25] employed MCS to simulate edge failures and ascertain whether the network failed due to the randomly selected edge. In his view, a significant issue with the MCS approach is its sluggishness when the probability of failure is low. Cardoso et al. [8] employed neural networks alongside Monte Carlo simulation in their investigation of structural dependability. The computation of dependability via MCS can be time-consuming, as it allows for the analysis of just one network structure at a time. To resolve this, they combined neural networks with MCS, resulting in decreased computation time and enhanced accuracy in dependability evaluations.

Artificial neural networks (ANNs) replicate the structures of neural networks utilised by the human brain. To acquire knowledge from experiences, the components of artificial neural networks communicate both sequentially and concurrently, much to the brain. A training set comprising inputs and corresponding known outputs is employed to facilitate this learning. The principal applications of artificial neural networks (ANNs) include control, pattern recognition, optimisation, associative memory, and prediction (Jain & Mao, 2021). To evaluate the reliability of the network, Srivaree-ratana et al. [41] utilised an artificial neural network. Their research encompassed training the artificial neural network utilising diverse topologies and link dependabilities. The optimal network topology was subsequently identified by employing the ANN to forecast the network's dependability about the topology and link reliabilities. Ultimately, they ascertained the exact reliability of each topology through its use. By comparing their estimation against a precise method and an upper bound established by a polynomial time approach, they demonstrate that their estimation performs effectively in practice, albeit with significant computational expense.

Establishing constraints on the network's reliability offers an alternative approach to both precise and approximate reliability assessments. The limit-finding method is less computationally expensive than alternative approaches; yet, it lacks precision as it only provides bounds rather than a definitive reliability. Sebastio et al. [37] devised an approach to ascertain the limitations on the reliability of a two-terminal network. The algorithm enables the user to select the duration for execution. Their approach considers minimum cuts and minimum pathways. A minimum route is a collection of interconnected nodes in a network that would stay connected even if one connection were severed.

To reduce the dependency between the upper and lower boundaries, their method identifies the most critical minimal paths and cuts inside the network. Bounds have an additional use, as discussed by Satitsatian and Kapur [36]. To ascertain the exact reliability and its bounds, they identified a bottom limit for the network's dependability. To achieve reduced dependability with minimal computational effort, they devised a method to identify a subset of lower boundary points. Ramirez-Marquez and Rocco [35] proposed an innovative method for addressing all-terminal network reliability allocation problems (RAP). Their aim in addressing this difficulty was to determine the ideal network cost while considering reliability.

Their developed process comprises three stages as follows: Initially, generating network configurations; subsequently, assessing each network's reliability through MCS; thirdly, imposing penalties on networks that do not meet the reliability criteria; and finally, ranking the networks from highest to lowest performance. Their algorithm revealed solutions that were, at most, 21% less costly and, at minimum, 7% more economical than those previously identified in the literature. Yeh et al. [46] proposed a method for utilising Monte Carlo Simulation in resource allocation. Their proposed approach was Movable Cluster Swarm Optimisation (MCS-PSO). Their objective was to meet reliability thresholds while minimising component expenses. Their methodology surpassed MCS independently in terms of efficiency and reliability estimation. Others have proposed alternative methods for establishing reliable networks.

Mettas [30] examined the component-level reliability allocation issue for generic systems. From his research, we may deduce the essential system dependability criteria for the reliability of various components. Both Jan et al. [22] and AboElFotoh & Al-Sumait [1] utilised methods to analyse network topology. Their objective was to identify the optimal topological configuration of links that minimised costs while satisfying the essential requirements for network stability. Jan et al. [22] employed a branch-and-bound decomposition method. Their methodology decomposes the network into smaller problems that their branch and bound algorithm can manage, based on the quantity of linkages. AboElFotoh and Al-Sumait [1] successfully addressed the same issue using an artificial neural network (ANN)..

2.1 Training with Positive Reinforcement

The many different approaches to machine learning are partitioned into distinct categories based on the learning process of the algorithm. One of these procedures is known as reinforcement learning. According to Zhang [47], reinforcement learning is dependent on a training data set that includes both positive and negative reinforcement in order to educate a computer to respond effectively to new situations. This is necessary in order to train the computer to respond appropriately. Because of this input, the machine is now able to perform the task more effectively the next time it is used. The algorithm for reinforcement learning, which selects an action at random, is the one that decides the value of an action. Both the value of achieving a new state and the value of receiving an immediate reward are factors that contribute to the value of the acts. Learning the value of optimal state/action combinations through repeated application of this technique is the objective of reinforcement learning, which aims to maximise the overall reward by learning the value of these combinations.

Reinforcement learning (RL) is a technique that can be useful in a wide range of different industries. The gaming industry was one of the first to adopt real-time gaming. In a video game, the player frequently assumes control of a character and makes decisions over what actions to do. RL considers the behaviours to be the product of the choices that they make. Through the utilisation of reinforcement learning (RL) to test out a wide range of situations, the researchers were able to teach the system to select the most appropriate action in each and every circumstance. According to Lin et al. [27], the two video games that were investigated were Flappy Bird and Breakout. In order to train both games, a neural network and reinforcement Q-learning were utilised in conjunction with one another. as well as one that does not use a neural network. In comparison to the situation in which a neural network was not utilised, the amount of time required to train the model was dramatically reduced.

The application of reinforcement learning is effective in a variety of contexts, not just the use of networks. The deep reinforcement learning model that Yang et al. [45] developed was created with the purpose of investigating different strategies for the distribution of computer network resources. Their key purpose was to guarantee the dependability of the system from the beginning to the end without fail. In order to ensure that the channels of the system did not fall short of the quality standards that they had established, they utilised a Q-learning algorithm to provide assistance to the system in the process of resource allocation. It was concluded that the Q-learning strategy was successful in their research after around one hundred of the training attempts were completed. Gottesman et al. [17] carried out research on the application of RL and other types of artificial intelligence in healthcare systems. The decisions that are made about the timing of particular tasks within a healthcare facility have a direct impact on the health of the patients. RL can provide assistance to healthcare personnel in deciding the most appropriate course of treatment for a patient based on their baseline state by analysing the results of past decisions. This is possible when the training has been completed. The implementation of RL in the medical field has made it possible to optimise the treatment sequences for patients.

2.2. Advancements Made in Reliability

Improving a system's dependability from the beginning of its lifecycle—during design, development, and operation—is what reliability growth is all about. Running a system through its paces to identify its weak points and then modifying its design to make those areas less likely to repeat is the basic notion, with the end goal of making the system more dependable. By using reliability growth models, one can make design improvements that improve a system's dependability. One of the pioneers in studying reliability's evolution was Duane [12]. He found that the rate of reliability improvement during development was roughly the same for mechanical and electromechanical systems when he compared them. How long does it take for systems to learn to reliably predict future events? That was the main focus of his research. According to his research, there was a virtually linear relationship between the logarithm of the cumulative failure rate and the cumulative operational hours. Crow explored reliability in relation to the age of the system in additional detail [10].

The Army Material Systems Analysis Activity (AMSSAA) paradigm was one of his recommendations. In his book "The AMSAA Reliability Growth Guide," he proposed a nonhomogeneous Poisson process model with a Weibull intensity function to study age-dependent reliability. The book "summarised the benefits of reliability growth management in finding unforeseen deficiencies, designing improvements, reducing risk, and increasing the probability of meeting objectives" (Kurtz et al., [26]).

According to Cahoon et al. [7], reliability growth models for generic systems have three potential uses. As part of this, we will make enhancements to the system's reliability, track our progress towards those targets, and keep the project moving forward. Reliability growth models also finds practical application in system testing conducted by the Department of Defence (DoD). Defence Department personnel are trained to use one of two reliability growth models. Using nonhomogeneous Poisson processes (NHPPs), one type is system-level. Competitive risk, on the other hand, takes a sequential look at several failure types. It is possible to track the frequency and duration of failures using NHPP models. All of the competing risk models consider the system as a whole, rather than focussing on its individual components. Consequently, the system can't work unless all of its components are fully functional.

3. Basic Model Construction

Our challenge involves an initial network including n nodes and a specified set of n(n-1) edges. We examine a sequential decision problem including m time periods, where in each period t=1,2,…,m, we can execute restricted investments to include possible edges from the collection E={{i,j}:i=1,2,…,n−1;j=i+1,2,…,n} into the network and/or enhance the dependability of current edges. An edge {i, j} ∈E that has been included into the network and enhanced $z$ij times is presumed to possess dependability ki+$z$ij$l$ij, where kij and $l$ij are parameters.

Our objective is to optimise the total discounted reward accrued throughout the time intervals t=0, 1…, m−1, where the reward at time interval t is contingent upon the network's all-terminal dependability immediately after that interval. A predetermined budget $BBtt$ is allocated at the commencement of each time period t=0, 1…, m−1 and may be utilised for immediate activities or deferred for future usage. Parameters $cciiii$ and $ppiiii$ delineate the expense associated with adding an edge {i, j}∈E and enhancing an existing edge {i,j}∈E, respectively.

The network state s before to any time period is characterised by the tuple

$$s=(t,R,\beta),$$

where t∈ {0, 1,m−1} denotes the number of completed time periods, R is a |E|-vector indicating the reliability of each edge in the network, and β represents the remaining budget. Denote the components of R as $r$ij, {i, j} ∈E, where $r$ij=0 if the edge {i, j} has not been included into the network.

In state s= (t, R, β), an action is described as a= (X, Y) where X and Y are |E|-dimensional vectors. The vector X comprises of elements xij, where {i, j} ∈ E, with xij = 1 if the edge {i, j} is included into the network; 0 otherwise. The vector YY comprises elements $y$ij, {i,j}∈E, where $y$ij=1 indicates that the edge {i,j} has been enhanced; 0 otherwise. The operation with $x$ij=$y$ij=0 for every {i, j} ∈E signifies the decision to progress to the subsequent time period without augmenting or enhancing any supplementary edges. The viable actions in state s= (t, R, β) are delineated by the equations:

$$\sum_{\{i,j\}\in E}(x_{ij}+y_{ij}) \le 1 \tag{1}$$

$$\sum_{\{i,j\}\in E}(c_{ij}x_{ij}+p_{ij}y_{ij}) \le \beta \tag{2}$$

$$x_{ij}=0, \forall\{i,j\}\in E: r_{ij}>0 \tag{3}$$

$$y_{ij}=0, \forall\{i,j\}\in E: r_{ij}=0 \tag{4}$$

$$r_{ij}+k_{ij}x_{ij}+l_{ij}y_{ij} \le 1, \ \forall\{i,j\}\in E \tag{5}$$

Equation (1) says that we can only do one or zero actions during a given time period. Equation (2) says that the actions we do must be less than or equal to the amount we have left for the period. Keep in mind that any spending that wasn't used in time period bb can be carried over and used in later time periods. For this reason, it might be best to move on to the next period even if there are enough resources to do one of the other tasks. For action $x$ij=1 to be possible, equation (3) says that $r$ij=0. This means that an edge {i, j} ∈E can't be added if it's already in the network. For action $y$ij=1 to be possible, equation (4) says that $r$ij > 0. This means that an edge {i, j} in the network can only be made better if it was already there.
One of the three actions that are possible is to improve an edge (i.e., $y$ij=1 for some 19 {i, j} ∈E), add an edge (i.e., $x$ij=1 for some {i, j} ∈E), or choose to move on to the next time (i.e., $x$ij=0 ∀ {ii, jf). If you do something with $x$ij=1 or $y$ij=1 for some {i, j} ∈E, it doesn't mean we'll be in a new time period; it just means that the state variables R and β have changed. The fifth equation makes sure that an edge can't be given a reliability number higher than 1.

The state transition function is now defined as (t', R′, β') =g (s, a) for an action a= (X, Y) done in state s=(t,R,β). The new state is set by if $x$ij=$y$ij=0 ∀ {i, j} ∈E

$$t' = t+1, \tag{6}$$

$$R' = R, \tag{7}$$

$$\beta' = \beta + B_{t+1}. \tag{8}$$

Equation (6) indicates that the time period, t', following a state transition is one period subsequent to the preceding time period, t. Equation (7) asserts that the network's dependability remains constant throughout a state shift. Equation (8) indicates that the budget in the new state post-transition is determined by the residual budget from the prior state in conjunction with the fixed budget allocated for the new period.

4. The Initial Experiments and Analysis of the Model

The RL problem from [6] was used to train our model. It was written in Python using the OpenAI stable baselines and followed a standard approach. There are a number of reliable versions of reinforcement learning methods in the OpenAI package. There is a pre-trained RL agent in each application. This agent learns from what it sees, does, and is rewarded.
We used Python version 3.8.15 for our study. We used stable OpenMPI baselines to make sure that all methods could work. For stable baselines, we had to say how many training episodes our models should have. We used 5000 episodes for all of them.

A Maskable Proximal Policy Optimisation (M-PPO) method [18] was used for our tests. Based on the problem constraints, this method narrows down the action space to only actions that are possible. This means that for our problem, the only things that can be done are to add edges that aren't already in the network, make edges that are already in the network better, and make sure that all of these actions don't go over budget. A mask, which is a vector that keeps track of acceptable acts, is also used by the algorithm. This will restrict the activities that can be done with the XX and YY vectors.

We used an iterative method to build and study our network. To start, we made a simple model that only let one choice be made at a time. At first, this model only let you do one thing, which was to improve the reliability of current network edges by sending resources to them. This first model was made to make sure that the model and RL code were working right. Then, we made the model more complicated so that we could make more than one choice in each time and take more than one type of action (like adding new lines or improving existing ones).

The first model was mostly about making one of the network's edges better. This model looks at the benefits that are randomly given to each edge at the start and figures out the next best change to make. These benefits are shown as higher dependability that comes from upgrading to the chosen edge. With this update, the benefits for this edge will be worth less in the future.

5. Results and discussion:

To further investigate if the model's choices were influenced by the various reward ratios, we additionally examined them. Our research showed that the model improved more linkages when the ratio was favourable for short-term rewards and added more links when the ratio was favourable for long-term rewards. With respect to the rewards ratio, Figure 6 displays the trend. On multiple occasions, the model also discovered that networks with lower budgets or shorter time frames may obtain superior results. It is possible that these outcomes were caused by the training process of the model. More training sessions may have helped the model perform better on the larger networks. A five-period problem instance with a rewards ratio of 1:2 and a budget of $3,000 per period achieved a final all-terminal reliability of 1.0 in the five-node network with initial and new link reliabilities of 0.9/0.9. In contrast, the final all-terminal reliability for the seven-period problem instance with a rewards ratio of 1:2 and a budget of 3,000 per period was 0.9995. The all-terminal dependability of the ideal 7-node network is 0.9999. This dependability was achieved in an instance of the issue with seven periods, a rewards ratio of 1:2, a per-period budget of 7,000, and initial and new link reliability of 0.9/0.9. Nine links were added and six were improved by the model.

6. Conclusion

Our investigation focused on the construction of networks with five, seven, and ten nodes, all of which were constructed using the identical pricing for connection upgrade and addition. It is recommended that a number of different combinations of addition and improvement costs be studied in order to further investigate the findings that the model has drawn regarding the addition or enhancement of links. Due to the fact that our research was only focused on three different network sizes, which got increasingly computationally costly as the networks extended, additional research is required in order to develop a model that is capable of making decisions more quickly. It is possible that other sources of complexity were responsible for the prolonged run-times; nonetheless, one potential improvement would be to replace the current dependability-polynomial method for evaluating network reliability with an option that is more scalable for larger instances.

In addition, we established a great deal of assumptions concerning the reliability of the original link, the dependability of the new link, the amount that needs to be improved, the cost to add and enhance, the benefits, and some other things. In light of the fact that these assumptions imposed limitations on the outcomes of our experiments, it is recommended that in the future, more models with less stringent assumptions be developed.

REFERENCES

1. AboElFotoh, H. M., & Al-Sumait, L. S. (2001, December). A Neural Approach to Topological Optimization of Communication Networks, with Reliability Constraints. IEEE Transactions on Reliability, 50(4), 397-408.
2. Acevedo, P. E., Jackson, D. S., & Kotlowitz, R. W. (2006). Reliability Growth and Forecasting for Critical Hardware through Accelerated Life Testing. Bell Labs Technical Journal, 11(3), 121-135.
3. Ahmad, N., Bokhari, M., Quadri, S., & Khan, M. (2008). The exponentiated Weibull software reliability growth model with various testing-efforts and optimal release policy: A performance analysis. The International Journal of Quality & Reliability Management, 25(2), 211-235.
4. Ball, M. O. (1986, August). Computational Complexity of Network Reliability Analysis: An Overview. IEEE Transactions on Reliability, 35(3), 230-239.
5. Ball, M. O., Colbourn, C. J., & Provan, J. S. (1995). Network Reliability. In Handbooks in Operations Research and Management Science (Vol. 7, pp. 673-762).
6. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym. arXiv preprint arXiv:1606.01540.
7. Cahoon, J., Sanborn, K., & Wilson, A. (2021, November). Practical reliability growth modeling. Quality and Reliability Engineering International Special Issue: Advancing Statistical Methods for Testing and Evaluating Defense Systems, 37(7), 3108-3124.
8. Cardoso, J. B., de Almeida, J. R., Dias, J. M., & Coelho, P. G. (2008, June). Structural reliability analysis using Monte Carlo simulation and neural networks. Advances in Engineering Software, 39(6), 505-513.
9. Cinque, M., Cotroneo, D., Pecchia, A., Pietrantuono, R., & Russo, S. (2017). Debugging-workflow-aware software reliability growth analysis. Software: Testing Verification, and Reliability, 27(7).
10. Crow, L. H. (1975). Reliability Analysis for Complex, Repairable Systems. Aberdeen Proving Ground, Maryland: Army Material Systems Analysis Activity.
11. Dempsey, P. J., & Sheng, S. (2013, May). Investigation of data fusion applied to health monitoring of wind turbine drivetrain components. Wind Energy, 16(4), 479-489.
12. Duane, J. T. (1964, April). Learning Curve Approach to Reliability Monitoring. IEEE Transactions on Aerospace, 2(2), 563-566.
13. Editors, H. (2009, November 24). Blackout hits Northeast United States. Retrieved from History: https://www.history.com/this-day-in-history/blackout-hits-northeast-united-states

14. Fries, A. (1993, June). Discrete Reliability-Growth Models Based on a Learning-Curve Property. IEEE Transactions on Reliability, 42(2), 303-306.
15. Gaur, V., Yadav, O. P., Soni, G., & Rathore, A. P. (2021). A literature review on network reliability analysis and its engineering applications. Journal of Risk and Reliability, 235(2), 167-181.
16. Goldbeck, N., Angeloudis, P., & Ochieng, W. Y. (2019, August). Resilience assessment for interdependent urban infrastructure systems using dynamic network flow models. Reliability Engineering & System Safety, 188, 62-79.
17. Gottesman, O., Johansson, F., Komorowski, M., Faisal, A., & Sontag, D. (2019, Jan). Guidelines for reinforcement learning in healthcare. Nature Medicine, 25(1), 16-18.
18. Huang, S., & Ontañón, S. (2020). A Closer Look at Invalid Action Masking in Policy Gradient Algorithms. arXiv preprint arXiv:2006.14171.
19. Hui, K.-P. (2005). Network Reliability Estimation. Doctoral Dissertation.
20. Hussin, M., Hamid, N. A., & Kasmiran, K. A. (2015, January). Improving reliability in resource management through adaptive reinforcement learning for distributed systems. Journal of Parallel and Distributed Computing, 75, 93-100.
21. Jain, A. K., & Mao, J. (1996). Artificial Neural Networks: A Tutorial. Computer, 31-44.
22. Jan, R.-H., Hwang, F.-J., & Cheng, S.-T. (1993, March). Topological Optimization of a Communication Network Subject to a. IEEE Transactions on Reliability, 42(1), 63-70.
23. Kanoun, K., Kaaniche, M., Beounes, C., Laprie, J.-C., & Arlat, J. (1993, June). Reliability Growth of Fault-Tolerant Software. IEEE Transactions on Reliability, 42(2), 205-219.
24. Kapur, P., & Garg, R. (1992, July). A Software-Reliability Growth-Model for an Error-Removal Phenomenon. Software Engineering Jornal, 7(4), 291-294.
25. Karger, D. R. (1999). A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. SIAM Journal on Computing, 29(2), 492-514.
26. Kurtz, J., Sprik, S., & Bradley, T. H. (2019). Review of transportation hydrogen infrastructure performance and reliability. International Journal of Hydrogen Energy, 12010-12023.
27. Lin, C.-J., Jhang, J.-Y., Lin, H.-Y., Lee, C.-L., & Young, K.-Y. (2019). Using a Reinforcement Q-Learning-Based Deep Neural Network for Playing Video Games. Electronics.
28. Lloyd, D. K. (1986). Forecasting Reliability Growth. Quality and Reliability Engineering International, 2, 19-23.
29. Mahmoodzadeh, Z., Wu, K.-Y., Droguett, E. L., & Mosleh, A. (2020). Condition-Based Maintenance with Reinforcement Learning for Dry Gas Pipeline Subject to Internal Corrosion. Sensors, 20(19).
30. Mettas, A. (2000). Reliability Allocation and Optimization for Complex Systems. Annual Reliability and Maintainability Symposium. 2000 Proceedings. International Symposium on Product Quality and Integrity, 216-221.
31. Milanovic, J. V., & Zhu, W. (2018, September). Modeling of Interconnected Critical Infrastructure Systems Using Complex Network Theory. IEEE Transactions on Smart Grid, 9(5), 4637-4648.
32. Newman, M. (2010). Networks: An introduction. Oxford, United Kingdom: Oxford University Press.
33. Provan, J. S., & Ball, M. O. (1983, November). The Complexity of Counting Cuts and of Computing the Probability that a Graph is Connected. SIAM Journal on Computing, 12(4), 777-788.
34. Pruitt, S. (2020, August 27). How Levee Failures Made Hurricane Katrina a Bigger Disaster. Retrieved from History: https://www.history.com/news/hurricane-katrina-levee-failures
35. Ramirez-Marquez, J. E., & Rocco, C. M. (2008, November). All-terminal network reliability optimization via probabilistic solution discovery. Reliability Engineering & System Safety, 93(11), 1689-1697.
36. Satitsatian, S., & Kapur, K. C. (2006, June). An Algorithm for Lower Reliability Bounds of Multistate Two-Terminal Networks. IEEE Transactions on Reliability, 55(2), 199-206.
37. Sebastio, S., Trivedi, K. S., Wang, D., & Yin, X. (2014, November 1). Fast computation of bounds for two-terminal network reliability. European Journal of Operational Research, 238(3), 810-823.
38. Sen, A., & Fries, A. (1998). Model Misspecification Effects Within A Family of Alternative Discrete Reliability-Growth Models. Lifetime Data Analysis, 65-81.
39. Serrano, W. (2019). Deep Reinforcement Learning Algorithms in Intelligent Infrastructure. Infrastructures, 4(3).
40. Shao, X., & Fang, X. (2016). Estimation in discrete reliability growth, a growth model with coefficient condition. Journal of Systems Science and Complexity, 1112-1122.