A Simple Monte-Carlo Method for Estimating the Continuous-State Two-Terminal Network Reliability at Required Demand Level

WEI-CHANG YEH
Department of Industrial Engineering and Management Engineering
National Tsing Hua University
P.O. Box 24-60, Hsin-Chu, 300
TAIWAN

Abstract: - The reliability at required demand level *d* (M2R$_d$) is usually selected as the most important index of two-terminal multi-state networks (MSNs) whose arcs have independent, discrete, limited and multi-valued random capacities. To evaluate M2R$_d$ is a NP-hard problem and is too costly to obtain through traditional techniques. Up to now, only one Monte-Carlo Simulation (RC$_{MCS}$) is proposed to evaluate M2R$_d$. Moreover, RC$_{MCS}$ not only requires to overcome NP-hard problems to know all minimal multi-state cuts (*d*-MCs) in advance, but also its replications all need an exponential number of comparisons. A simple polynomial-time Monte-Carlo Simulation (Yeh$_{MCS}$) is proposed in this article to estimate M2R$_d$ without finding any *d*-MCs. Yeh$_{MCS}$ can also solve the reliability (C2R$_d$) for the continue-state networks (CSN) which is a novel generation of MSN. The estimators of Yeh$_{MCS}$ are compared with RC$_{MCS}$ and exact solutions. The analysis indicates that Yeh$_{MCS}$ is more practical, efficient and effective for most cases from the proposed experiments.

Key-Words: - Reliability, Binary/Multi/Continuous-State Network, Monte Carlo Method (MCS), Minimal Path/Cut Sets (MPs/MCs), *d*-MP/*d*-MC, NP-hard

1 Introduction

In recent years, network reliability theory has been applied extensively in many real-world systems such as oil/gas production systems [1], computer and communication systems [2,3], power transmission and distribution systems [4], transportation systems [5], etc. System reliability thus plays important roles in our modern society. The reliability is recommended to be measured and evaluated through performances of the systems which can be modeled as stochastic networks or into fault trees first.

Each arc of a binary-state network (BSN) has only operated or failed two states [1-10]. The BSN reliability evaluation approaches exploit a variety of tools for system modeling and reliability index calculation. Among the most popular tools are network-based algorithms founded in terms of either minimal cuts (MCs) or MPs [5,7-18]. A MP/MC is a path/cut set such that if any edge is removed from this path/cut set, then the remaining set is no longer a path/cut set. However, both the problems in finding all MCs/MPs and computing the exact reliability in terms of the known MCs/MPs are also NP-hard. Numerous Monte Carlo Methods (MCSs) have been developed for relatively large BSN reliability analysis [21-23].

In MSNs, each arc has several possible independent, limited and discrete capacities and may fail [11-19]. Hence, MSNs are more practical and reasonable than BSNs in many real-life situations [11-19]. For example, Figure 1 is a a multi-state network (also called the bridge network) with $V=\{s,t,a,b\}$, $E=\{e_1,e_2,e_3,e_4,e_5,e_6\}$, and Figure 2 is the network induced by $X=(2,2,1,1,1,1)$ from Figure 1. If each arc of Figure 1 has only operative or failed two states, then it is a binary network. Otherwise, it is a multi-state network, e.g. the (capacity) states of e_1 are 0, 1, 2, and 3.

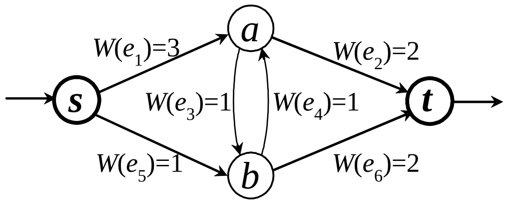

Fig 1. The bridge network

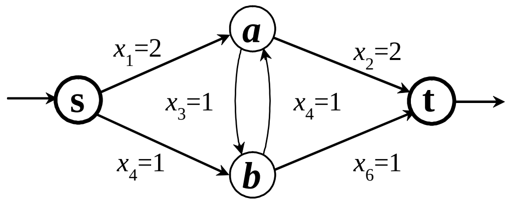

Fig 2. The network induced by (2,2,1,1,1,1) from Fig 1.

Analysis of BSNs extends to MSNs has already become a popular subject in the reliability problem.

The two-terminal MSNs reliability at required demand level d (M2R$_d$) is the probability that a demand of d units can be transmitted from the source to sink nodes through multi-state arcs. Generally, there are four main steps behind the tradition techniques for evaluating the M2R$_d$ as follows:

1. Find all MPs/MCs by treating the arc state to be binary.
2. Transfer MPs/MCs to d-MP/d-MC candidates using a time-consuming and very burdensome — Implicit Algorithm. A d-MP/d-MC candidate $X=(x_1, x_2, \ldots, x_{|E|})$ is a system vector, where x_i denotes the (current) capacity of the ith arc and $|E|$ is the arc number . If X is a d-MC candidate, then $\sum_{i=1}^{|E|} x_i = d$. If X is a d-MP candidate, then the maximal flow from source to sink nodes is equal to d under assumption that x_i denotes the maximal capacity of the ith arc for all i.
3. Verify all d-MP/d-MC candidates to find all (real) d-MPs/d-MCs.
4. Compute the M2R$_d$ in terms of d-MPs/d-MCs using some special and complicate method, e.g. the inclusion-exclusion method.

All of the four steps are NP-hard problems [12-19]; i.e. M2R$_d$ is NP-hard. Besides, the value of d needs to be a natural (nonnegative integer) number; otherwise the number of d-MCs/d-MPs is infinite. Therefore, the lack of generality limits the practical use of this model. MCS has been effectively used for analyzing relatively large BSN. However, the MCS has been underutilized for approximating the M2R$_d$ to reduce the computational burdens [20]. Nevertheless, for the M2R$_d$ problem, the best-known MCS (RC$_{\text{MCS}}$) proposed by [20] needs to overcome the above first three NP-hard problems to find all the d-MCs and only for natural d. The efficiency of the simulation methods is an important measure of evaluation. The need for a more efficient, practical and intuitive MCS to evaluate M2R$_d$ for any $d>0$ without knowing MP/MC/d-MP/d-MC in advance thus arises.

The main focus of this study is to develop a MCS for estimating the M2R$_d$ to completely overcome four NP-hard obstacles and the limitation of d discussed above. We also extend the MSN to CSN such that arcs have independent, continuous, bounded and random capacities, and using the proposed MSN to solve the two-terminal CSNs reliability at required demand level d (C2R$_d$), where $d>0$ is unnecessary to be a natural number.

MSN fails to characterize the actual system reliability behavior, which is a continuous-state. The CSN is a novel generation of MSN and is first proposed in the literature. To the author's best knowledge, Yeh$_{\text{MSC}}$ is also a new technique to evaluate M2R$_d$ and C2R$_d$. The CSN is more suitable than MSN when events/states are continuous, and more difficult to evaluate C2R$_d$ than to evaluate M2R$_d$. Furthermore, it is impossible to obtain C2R$_d$ through traditional techniques which all need d to be a natural number. To show the efficiency and effectiveness of Yeh$_{\text{MCS}}$, Yeh$_{\text{MCS}}$ is compared with RC$_{\text{MCS}}$ and the exact solution for the M2R$_d$ problem. The statistical properties of the proposed estimator are also analyzed.

2 The CSN and the Proposed MCS

The capacity level of each arc is discrete in MSN, e.g. $w_{11}=0$, $w_{12}=3$ and $w_{13}=6$. However, the capacity level of each arc is continuous in CSN, e.g. $w_{1k}=\ln k$. In the traditional techniques, d must be a natural number due to the multi-state characteristic of arcs. Therefore, C2R$_d$ can not be solved through the four steps mentioned in Section 1. Thus, the tradition techniques including RC$_{\text{MCS}}$ all fail to evaluate C2R$_d$. Yeh$_{\text{MCS}}$ is a simple approach especially for the large complex CSNs/MSNs. It is harder to evaluate C2R$_d$ than M2R$_d$. On the contrary, Yeh$_{\text{MCS}}$ bases on the max-flow algorithm only and is simpler than RC$_{\text{MCS}}$ which depends on the complicated theory of the d-MC. The main idea of the Yeh$_{\text{MCS}}$ is very simple: a repletion is successful if the max-flow in $G(V, E, X)$ is not less than d, where X is generated using a sequence of random numbers.

The max-flow problem is one of the core models of the network analysis. It has been well-researched since 1960, and it can be found in all textbooks related to Graph theory and/or Operations research. Ford and Fulkerson were the first to study the max-flow problem. Currently, the fastest known max-flow algorithm independently developed by King, Rao and Tarjan [24] and by Phillips and Westbrook [25] run in $O(|V|\cdot|E|\cdot \log_{\frac{|E|}{|V|\log|V|}} |V|)$. However, the fastest known max-flow algorithm requires some sophisticated data structure techniques [26]. Therefore, the simplest algorithm proposed by Ford and Fulkerson and revised in [27] with time complexity $O(|V|\cdot|E|^2)$ was adapted in Yeh$_{\text{MCS}}$ (see the STEPs 2-5 in Yeh$_{\text{MCS}}$ below). The basic idea of the Ford and Fulkerson approach is: find a s-t path, send flows via this path, update the arc capacities in this path, and repeat the above procedures until no s-t path exists. The revised Ford and Fulkerson approach is modified and emerged in the main calculation procedure of Yeh$_{\text{MCS}}$. However, if the amount of flows sent from nodes s to t is not less than d then the repletion is halted without going further to find the max-flow. The steps of Yeh_{MCS} to estimate the $\text{C2R}_d/\text{M2R}_d$ without knowing d-MPs/d-MCs for all $d>0$ are given as follows:

0. Let $r=1$ and $n=0$.

1. Let $d^*=0$, and generate a random number, say p_j, from an uniform (0,1) distribution, let $x_j=w_{jk}$ if $p_{jk} \le p_j < p_{j,k+1}$ for all $j=1,2,\ldots,|E|$.

2. Find a s-t path $P^* \in G(V,E,X)$ such that $X(e)>0$ for all $e \in P^*$. If no s-t path exists, go to STEP 6.

3. Let $\delta=\text{Min}\ \{X(e) \mid \text{for all } e \in P^*\}$, and $d^*=d^*+\delta$.

4. If $d^* \ge d$, then let $n=n+1$ and go to STEP 6.

5. Let $X(e)=X(e)-\delta$, $X(e^*)=X(e^*)+\delta$ and go to STEP 2, where e^* with the opposite direction of e for all $e \in P^*$.

6. If $r=m$, then let $R^{\#}=n/m$ and halt. Otherwise, let $r=r+1$ and go to STEP 1.

STEPs 2-5 are the main calculation procedure in Yeh_{MCS} and the time complexity is $O(|V|\cdot|E|^2)$ only. These STEPs can be improved furthermore if fastest known max-flow algorithms are adapted. Therefore, each replication can be executed in polynomial time, and it is much better than that of RC_{MCS} which need an exponential time for each replication. Thus, Yeh_{MCS} is simpler and more efficient than RC_{MCS} according to the time complexity for each replication even without considering that RC_{MCS} needs to know all d-MCs in advance. The statistical properties of the estimator obtained from Yeh_{MCS} are analyzed as follows:

Theorem 1. The estimated reliability value $R^{\#}$ obtained from Yeh_{MCS} is an unbiased and consistent estimator of the exact reliability R. Its variance is given by $R[1-R]/m$, where m is the replication number.

Theorem 2. If the relative error ε and the confidence interval $(1-\alpha)\%$ are given, then the total number of replications of the simulation must be taken at least

$$m \ge \frac{Z^2_{\alpha/2}}{2\varepsilon^2}.$$

Obvously, Yeh_{MCS} is more efficient than RC_{MCS} no matter all d-MCs are known in advance as disscussed before. However, The estimator obtained from either of Yeh_{MCS} or MC_{MCS} is unbiased and consistent of the exact reliability R with the same variance and is given by $R[1\text{-}R]/m$. Therefore, four bench examples with 13 distinctive cases are given to illustrate and validate Yeh_{MCS}, and compare the estimator quality obtained from Yeh_{MCS} with the estimator quality obtained from RC_{MCS}.

3 Performance and Comparisons

To investigate the effectiveness and efficiency of Yeh_{MCS}, four bench examples are considered. The BSN versions of these examples are frequently used as illustrative examples in the BSN reliabilities. Example 1 is called the bridge network (see Figure 1) with 4 nodes and 6 arcs. Example 2 is called the ARPA network (see Figure 3) with 5 nodes and 11 arcs. Example 3 is a median network (see Figure 4) with 12 nodes and 21 arcs with 110 MCs. These three examples are MSNs presented here to display the simplicity of Yeh_{MCS} to estimate M2R_d without finding all d-MCs [20]. Example 4 is a relatively larger CSN (see Figure 5) with 36 nodes, 57 arcs and 34241 MCs. It is utilized to demonstrate the ability of Yeh_{MCS} to evaluate C2R_d. Note that the application of RC_{MCS} for this size of network is an inefficient and burdensome task even example 4 is a BSN.

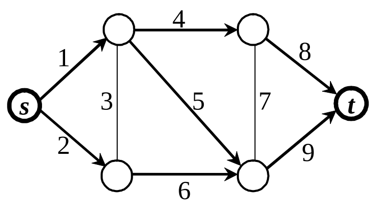

Figure 3. The ARPA network for Example 2

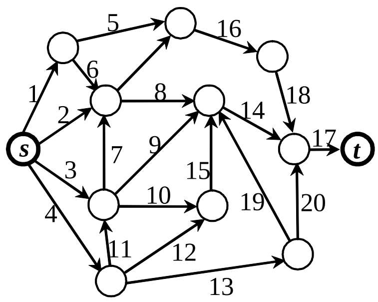

Figure 4. The network for Example 3

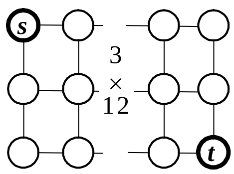

Figure 5. The network for Example 4

The required demand units (i.e. d) are 3, 10, and 5 for Examples 1-3, respectively [20]. All of the above information is obtained from [20] to get a fair comparison for the M2R_d problem. In example 4, suppose that the reliability of each arc has an exponential distribution with a mean reliability of $1/\lambda$, i.e. $R(x_i \le \tau)=1-e^{-\lambda\tau}$ for all i. The required demand units are 0.2, 0.4, …, 3.0 and the parameter λ are 0.1, 0.2, …, 1.0 for Example 4.

Yeh_{MCS} was coded with C^{++} and run on a Pentium 3 notebook with 1GHz to make comparisons at the same basis with RC_{MCS}. The running time unit was the second. The comparisons between Yeh_{MCS} and RC_{MCS} for the experiment results obtained from the three examples are presented in Tables 1-4. Table 1 gives the exact reliability and its estimations obtained by Yeh_{MCS} and RC_{MCS}, and also bounds obtained by MESP and MLQ.

For each of the cases, results are obtained through Yeh_{MCS} and RC_{MCS}, considering 50,000 runs. The variance of $R^{\#}$ listed in Table 1 is obtained using the following formula:

$$Var[R^{\#}]=\frac{R^{\#}(1-R^{\#})}{\text{the total repetition number}}.$$

The absolute relative error showed in Table 2 is based on the following equation:

$$Error[R^{\#}]=\frac{|R^{\#}-R_d|}{R_d}.$$

Table 1. Approximation results

Ex No.	Case No.	R	Bounds		Yeh_{MCS}		RC_{MCS}	
			MESP	MLQ	$R^{\#}$	$10^6 Var[R^{\#}]$	R^*	$10^6 Var[R^*]$
	1	.83099	.824680	.824935	.83059	2.81421	.83036	2.81725
	2	.67760	.659283	.662031	.67787	4.36725	.67674	4.37526
	3	.55374	.519458	.531444	.55385	4.94200	.55190	4.94613
	4	.49512	.439720	.469800	.49529	4.99956	.49366	4.99920
2	1	.91619	.913941	.913999	.91633	1.53339	.91542	1.54852
	2	.84439	.836295	.836799	.84417	2.63094	.84462	2.62474
	3	.71847	.696673	.700827	.71817	4.04804	.71862	4.04411
	4	.59907	.556872	.572389	.59907	4.80370	.60426	4.78260
3	1		.930707	.930728	.93900	1.14558	.94692	1.00525
	2		.910461	.910462	.91130	1.61665	.91200	1.60512
	3		.890943	.890967	.89884	1.81853	.89992	1.80128
	4		.536264	.555777	.66529	4.45358	.68354	4.32626

The quality of results are analyzed and considered in Table 2. There is no exact reliability for both Examples 3 and 4. Therefore, the corresponding absolute relative errors result from both RC_{MCS} and YEH_{MCS} related to Examples 3 and 4 all are not including in Table 2.

Table 2. Results of the absolute relative error* (in percentage) for Exs 1 and 2.

Example No.	Case No.	Yeh_{MCS}	RC_{MCS}	MESP	MLQ
1	1	.048135^	.075813	2.703074	2.297524
	2	.039994^	.126771	6.190145	4.025572
	3	.020768^	.331386	11.189207	5.113912
	4	.034335^	.294878	5.210440	3.041414
Average		.035808^	.207212	2.703074	2.297524
2	1	.015499^	.083826	0.245255	0.238925
	2	.026054^	.027239	0.958680	0.898992
	3	.041199	.021435^	3.033268	2.455092
	4	.000167^	.866511	7.043763	4.453577
Average		.020730^	.249752	2.820242	2.011646

^: the best among 4 methods.

For testing the efficiency of both simulation approaches, the running CPU times have been recorded and shown in Table 3. The running CPU times of RC_{MCS} listed in Table 3 do not include the exponential running time to obtain all *d*-MCs. Moreover, RC_{MCS} is impossible to estimate $C2R_d$, its running CPU time for Example 4 is not available.

Table 3. CPU time (in sec)

Example No.	Case No.	Yeh_{MCS}	$RC_{MCS}{}^*$
1	1	.160	.19
	2	.160	.17
	3	.128	.18
	4	.146	.16
2	1	.192	.56
	2	.256	.54
	3	.240	.52
	4	.272	.47
3	1	.536	23.994
	2	.457	23.053
	3	.558	22.762
	4	.727	17.445

* not including the exponential running time to obtain all *d*-MCs.

By the above experiments, we showed that Yeh_{MCS} for $M2R_d$ is superior to RC_{MCS} in the estimator qualities. In Table 2, the absolute errors of estimators obtained from Yeh_{MCS} are much less than that obtained from RC_{MCS} in Examples 1 and 2 (except Case 3 of Example 2). The average simulation absolute error is also better than that obtained from RC_{MCS} in Examples 1 and 2. Therefore, Yeh_{MCS} is more effective than RC_{MCS}. Yeh_{MCS} is also more efficient than the RC_{MCS} which not yet includes the exponential time of calculations to search for *d*-MCs in advance (see Table 3). It is more evidence that the running time of RC_{MCS} is increasing more than 40 times from Examples 2 to 3 while the node number and arc number both are increasing less than 2.5 times only.

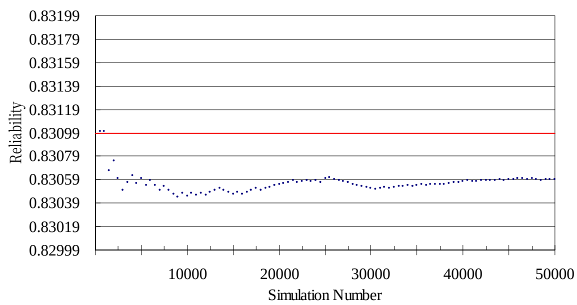

Fig 6a. The convergence of the Case 1 of Ex 1

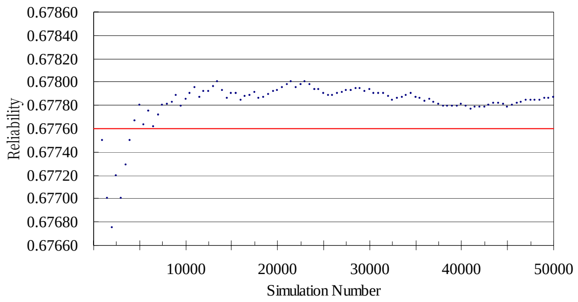

Fig 6b. The convergence of the Case 2 of Ex 1

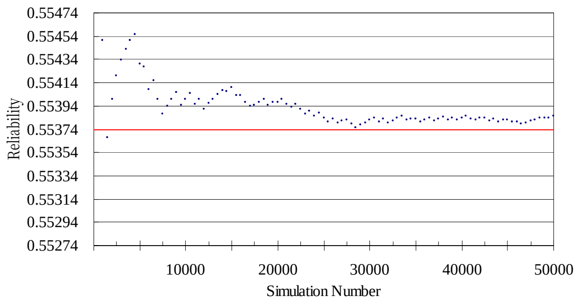

Fig 6c. The convergence of the Case 3 of Ex 1

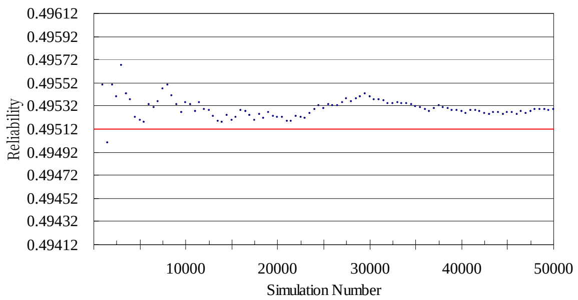

Figure 6d. The convergence of the Case 4 of Ex 1

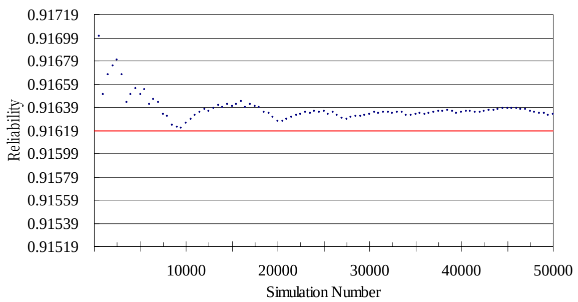

Fig 7a. The convergence of the Case 1 of Ex 2

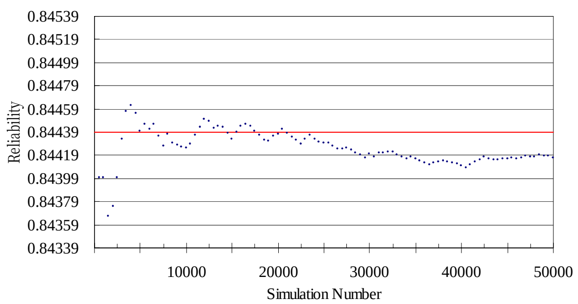

Fig 7b. The convergence of the Case 2 of Ex 2

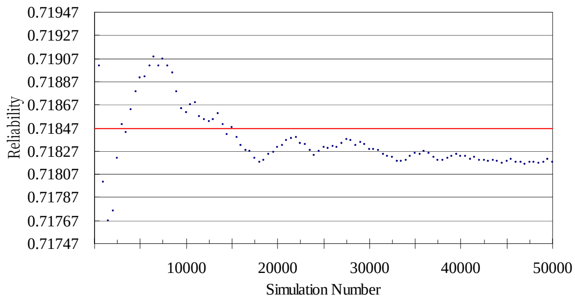

Fig 7c. The convergence of the Case 3 of Ex 2

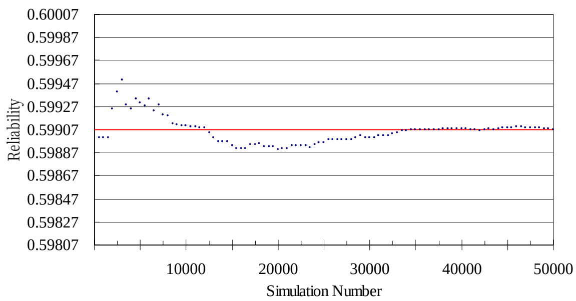

Fig 7d. The convergence of the Case 4 of Ex 2

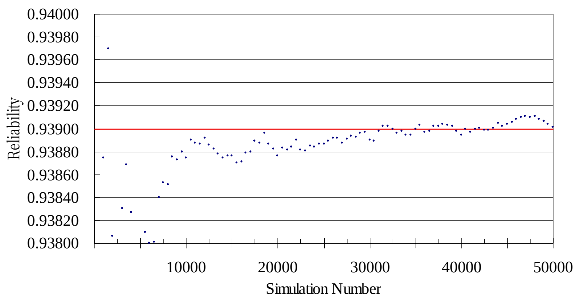

Fig 8a. The convergence of the Case 1 of Ex 3

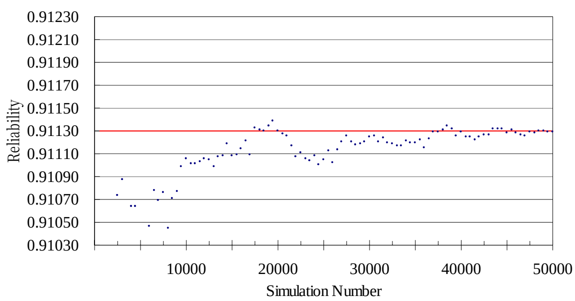

Fig 8b. The convergence of the Case 2 of Ex 3

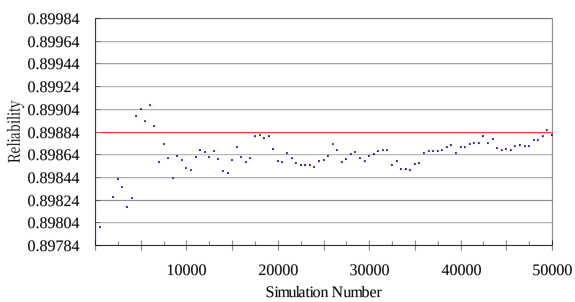

Fig 8c. The convergence of the Case 3 of Ex 3

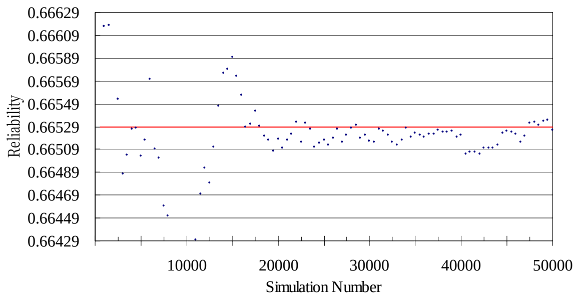

Fig 8d. The convergence of the Case 4 of Ex 3

Figures 6-8 demonstrate the trend of convergence of the estimator obtained from Yeh_{MCS} to provide an explicit perspective of how the approximations deviate at each run. These are made by the starting number of simulation runs at 500 and then later incrementing this number by 500 and conducting an independent simulation with 100 runs. The vertical axis represents the reliability values which are between the exact reliability+0.001 and the exact reliability-0.001. As pictured in the graphs, the estimators obtained from the proposed MCS converge rapidly.

Finally, the CPU running time for Example 4 using Yeh_{MCS} is less than 11 seconds from Table 4. No existing method can evaluate or even give the lower/upper-bounds for the $C2R_d$ problem. Therefore, the estimator obtained from Yeh_{MCS} provides valuable information for larger complex MSNs/CSNs.

Table 4. CPU time (in sec) for Example 4 under different combinations of α and *d*

		α									
		.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
d	.2	0.891	0.875	1.047	1.14	1.469	1.594	1.656	2.016	2.187	2.344
	.4	0.891	1.203	1.593	1.922	2.25	2.797	3.172	3.438	3.921	4.532
	.6	1.031	1.547	1.937	2.797	3.531	3.907	4.625	5.484	5.641	6.562
	.8	1.25	1.906	2.61	3.406	4.141	5.672	6.5	6.359	6.937	7.313
	1	1.344	2.281	3.328	4.516	5.39	7.813	6.969	7.656	8.312	8.266
	1.2	1.516	2.796	3.938	5.312	6.172	6.797	7.469	12.141	8.89	8.578
	1.4	1.985	3.14	4.86	5.718	6.922	7.25	7.813	8.094	8.718	9.047
	1.6	1.969	3.672	4.937	6.125	6.969	8.438	8.39	8.11	8.171	8.625
	1.8	2.079	3.937	6.594	7.062	7.672	8.235	8.234	9.312	8.391	8.859
	2	2.125	4.36	6.406	7.594	7.781	9.219	8.937	8.125	9.157	8.625
	2.2	2.5	4.531	6.125	7.359	8.203	9.219	8.141	8.797	8.031	8.656
	2.4	2.766	5.328	6.906	7.656	8.094	8.109	7.938	9	8.672	8.922
	2.6	2.875	6.093	7.172	8.172	9.516	8.453	9.156	8.703	9.813	9.469
	2.8	3.187	6.047	7.641	7.75	8.875	8.515	8.844	8.984	8.516	10.109
	3	3.438	5.703	8.594	8.625	8.875	8.172	9.609	8.453	8.516	9.812

4 Conclusion

In this study, a new MCS called Yeh_{MCS} is developed to evaluate the reliability of a novel generalized network called CSN. By extending the multi-state to continuous-state, CSN is a novel generation of MSN. It is more practical in many real-life situations, and existing method can not figure out its reliability. The exact computation of the binary/multi/continue-state network reliability is NP-hard [19]. Simulation is a valid approach to obtain fairly accurate approximations to the actual reliability in a reduced computational time. As we pointed out in Section 3, there are two major weak points in RC_{MCS}:

(1) all *d*-MCs need to be known in advance. Therefore,

- RC_{MCS} is more complicated and tedious using the *d*-MC concept.
- RC_{MCS} needs special techniques to overcome the NP-hard problem to obtain all *d*-MCs before it can be implemented.
- RC_{MCS} fails to estimate $M2R_d$ if any *d*-MC is unknown.
- If *d* is changed, RC_{MCS} is inapplicable before re-exploring entire new *d*-MCs.
- *d* can not be any positive number, otherwise no traditional techniques can find out all *d*-MCs.
- RC_{MCS} is not suitable in the CSN where *d* can be any positive number.

(2) RC_{MCS} requires to solve another NP-hard to decide whether a replication is successful.

Yeh_{MCS} is proposed to overcome all the above problems occurring in RC_{MCS} to meet the need for a more practical and efficient MCS. The proposed Yeh_{MCS} improves RC_{MCS} in the following six ways: (1) based only on the max-flow (the basic core in Graph Theory) instead of using the complicated concepts about the MC/MP/*d*-MP/*d*-MC, Yeh_{MCS} is simpler, (2) no need to know all *d*-MCs in advance, Yeh_{MCS} is more practical and reasonable (3) since *d* can be any positive number, Yeh_{MCS} is more useful and flexible and also can evaluate $C2R_d$, (4) without an exponential number of comparisons, Yeh_{MCS} is more efficient and reduces computational effort, (5) with better estimator quality as evidenced by the experiment results in Section 5, Yeh_{MCS} is more effective, (6) allowing the change of *d*, Yeh_{MCS} is also ideally suited to perform the sensitivity analysis to investigate the effect on the reliability if *d* takes on other possible values. Through the above discussion, the proposed Yeh_{MCS} is simpler, more practical, reasonable, useful, flexible, efficient and effective than RC_{MCS}.

References:

[1] T. Aven, “Availability evaluation of oil/gas production and transportation systems”,

Reliability Engineering. Vol. 18, 1987, pp. 35-44.

[2] K.K. Aggarwal *et al*., “A simple method for reliability evaluation of a communication system”, *IEEE Transactions on Communication*, COM-23, 1975, pp. 563-565.

[3] M.A. Samad, “An efficient algorithm for simultaneously deducing MPs as well as cuts of a communication network”, *Microelectronic Reliability*, Vol. 27, 1987, pp. 437-441.

[4] W.J. Ke and S.D. Wang, “Reliability evaluation for distributed computing networks with imperfect nodes”, *IEEE Transactions on Reliability*, Vol. 46, 1997, pp. 342-349.

[5] P. Doulliez and E. Jalnoulle, “Transportation network with random arc capacities”, *RAIRO, Rech. Operations Research*, Vol. 3, 1972, pp. 45-60.

[6] J. Esary and F. Proschan, “Coherent structures of non-identical components”, Technometrics, Vol. 5, No. 2, 1963, pp. 191-209.

[7] T. Jin and D. Coit, “Network reliability estimates using linear and quadratic unreliability of minimal cuts”, *Reliability Engineering & System Safety*, Vol. 82, No. 1, 2003, pp.41-8.

[8] W.C. Yeh, “Search for MC in modified networks”, *Computers & Operations Research*, Vol. 28, No. 2, 2001, pp. 177-184.

[9] W.C. Yeh, “Search for Minimal Paths in Modified Networks”, *Reliability Engineering & System Safety*, Vol. 75, No. 3, 2002/3, pp. 389-395.

[10] T. Aven, “Some considerations on reliability theory and its applications”, *Reliability Engineering and System Safety*, Vol. 21, 1988, pp. 215-223.

[11] A. Lisnianski and G. Levitin, “*Multi-state system reliability. Assessment, optimization and applications*”, Singapore: World Scientific, 2003.

[12] W.C. Yeh, “A simple algorithm to search for all *d*-MPs with unreliable nodes”, *Reliability Engineering & System Safety*, Vol. 73, No. 1, 2001, pp. 49-54.

[13] W.C. Yeh, “A Simple Method to Verify All *d*-Minimal Path Candidates of a Limited-Flow Network and its Reliability”, *International Journal of Advanced Manufacturing Technology*, Vol. 20, No. 1, 2002/7, pp. 77-81.

[14] W.C. Yeh, “A simple approach to search for all *d*-MCs of a limited-flow network”, *Reliability Engineering and System Safety*, Vol. 71, No. 1, 2001, pp. 15-19.

[15] W.C. Yeh, “A simple MC-based algorithm for evaluating reliability of stochastic-flow network with unreliable nodes”, *Reliability Engineering & System Safety*, Vol. 83, No. 1, 2004/1, pp 47-55.

[16] W.C. Yeh, “A New Approach to the *d*-MC Problem”, *Reliability Engineering & System Safety*, Vol. 77, No. 2, 2002/8, pp. 201-206.

[17] Y. Lin, “Using minimal cuts to evaluate the system reliability of a stochastic-flow network with failures at nodes and arcs”, *Reliability Engineering & System Safety*, Vol. 75, 2002, pp. 41-6.

[18] Y. Lin, “A simple algorithm for reliability evaluation of a stochastic- flow network with node failure”, *Operations Research*, Vol. 28, 2001, pp. 1277-85.

[19] C.J. Colbourn, “*The combinatorics of network reliability*”, Oxford University Press, New York, 1987.

[20] E.J. Ramirez-Marquez and D.W. Coit, “A Monte-Carlo simulation approach for approximating multi-state two-terminal reliability”, *Reliability Engineering & System Safety*, Vol.87, No.2, 2005, pp. 253-264.

[21] G.S. Fishman, “*Monte Carlo – concepts, algorithms, and applications*”, New York: Springer-Verlag, 1996.

[22] W.C. Yeh, “A new Monte Carlo method for estimating network reliability”, *Proceedings of the 16th International Conference on Computers and Industrial Engineering*, Ashikaga, Japan; 1994.

[23] T.L. Landers, H.A. Taha, and C.L. King, “A reliability simulation approach for use in the design process”, *IEEE Transactions on Reliability*, Vol. 40, 1991, pp. 177-18l.

[24] V. king, S. Rao, and R.E. Tarjan, “A faster deterministic maximum flow algorithm”, *Journal Algorithms*, Vol.17, No.3, 1994, pp.447-474.

[25] S. Phillips and J. Westbrook, “Online load balancing and network flow”, *Proceeding of the 24th ACM Symposium on Theory of Computing*, 1992, pp.402-411.

[26] J. Edmonds and R.M. Karp, “Theoretical Improvements in Algorithm Efficiency for Network Flow Problems”, *Journal of ACM*, Vol. 19, 1972, pp. 248-264.

[27] L.R. Ford Jr. and D.R. Fulkerson, “*Flows in Networks*”, Princeton University Press, 1962.