A combined Monte Carlo and cellular automata approach to the unreliability analysis of binary network systems

E Zio*, M Librizzi, and G Sansavini
Department of Nuclear Engineering, Polytechnic of Milan, Milan, Italy

The manuscript was received on 15 March 2007 and was accepted after revision for publication on 25 July 2007.

DOI: 10.1243/1748006XJRR66

Abstract: In the present work, cellular automata are combined with Monte Carlo sampling to solve two critical problems related to the unreliability assessment of complex networks composed of nodes interconnected by binary arcs: the identification of the minimal cut sets of the network and the computation of the Fussell–Vesely importance measures assessing the criticality of each arc with respect to the network unreliability. The effectiveness of the method is tested on two literature case studies.

Keywords: network systems, reliability assessment, Monte Carlo simulation, cellular automata, minimal cutset identification, Fussell–Vesely importance measures

1 INTRODUCTION

The modern technological society is increasingly based on critical distributed systems and infrastructures made up of many elements physically and logically interconnected in a very complex way. Typical examples of such distributed technological systems are those employed in transportation, telecommunication, gas and oil distribution, water supply, electric power, and heat transfer. These systems perform essential services for much of the world population, so that it becomes of primary importance to identify the associated criticalities and vulnerabilities accurately in order to be able to design adequate protections against their failure. These needs are increasingly perceived in a world where deregulation of services is favoured and where there has been a worrying increase in terrorism and sabotage threats.

The first of those issues, i.e. the deregulation of the service market, aims at a more effective usage of the network systems through the introduction of a competitive framework, with the result of operating at limit conditions and thus increasing the associated risk. This situation calls for greater dependability of the systems and a higher adaptability to cope with the dynamism of the demands.

The second issue, regarding sabotage actions, implies widening the scope of risk assessment in order to take such scenarios into account in a systematic way.

In spite of the potentially more critical vulnerabilities of network systems, due to the intrinsic distributed character, the methods for analysing the safety, reliability, and availability of network systems have received far less attention than those for process, production, and storage systems. The main reasons are that the latter methods generally involve greater capital expenditure and the associated accidental events might have more severe consequences, causing localized damage to the neighbouring community, the environment, and the plant. On the other hand, distributed systems are relatively cheaper and they can suffer failures and accidents with lower associated consequences, generally shared among users ‘distributed’ both geographically and socially.

In the present work, a methodology, based on a combination of cellular automata (CA) and Monte Carlo (MC) sampling, is proposed to identify the minimal cut sets (MCSs) of a binary network, i.e. a network where arcs can only be in fully working or fully failed states, while computing its two-terminal unreliability.

The determination of the MCS is fundamental for the identification of system criticality and vulnerability because it allows the determination of those component configurations critical for system failure. In principle, for those systems whose success criterion is determined by the connectivity between a source

*Corresponding author: Department of Nuclear Engineering, Polytechnic of Milan, LASAR, Via Ponzio 34/3, Milan 20133, Italy. email: enrico.zio@polimi.it
node and a target node, the MCS identification may be accomplished by depth-first procedures [1–3]. Yet, these approaches lead to NP-hard (non-deterministic polynomial-time hard) problems, requiring cumbersome and mathematically intensive methods of solution. Furthermore, in many real cases, modification of an existing network is required for expansion or reinforcement planning or occurs inadvertently due to link failures. In such cases, the standard algorithms entail recomputing the network connection and reliability.

A ranking of arcs according to their criticality in contributing to network unreliability has also been performed by comparing their Fussell–Vesely importance measure values [4]. Among the various component importance measures available [5], the Fussell–Vesely measure has been chosen because it adequately represents the contribution of a component to the system unreliability in terms of the maximum fractional decrement in unreliability achievable when that component is always available; as such it is, for example, most appropriate for ranking the system components with respect to their maintenance priority.

CA have been recently proposed to verify directly network connectedness [6]. They form a general class of mathematical models that are appealingly simple and yet capture a rich complexity of behaviour of complex systems [7]. As such, CA have been used to study and model many real complex systems – including fluids, neural networks, molecular systems, ecological systems, economical systems, and network systems. CA offer a significant computational potential due to their spatially and temporally discrete nature, characterized by local interaction and an inherently parallel form of evolution.

In the context of network reliability analysis, CA operate in a way basically to mimic traditional graph methods, such as depth-first or breadth-first search [1–3]. Combined with MC sampling and simulation, they have been shown capable of [6, 8–10]:

(a) verifying the existence of the connection between source and terminal targets in a network of nodes;
(b) solving the problem of (a) after a connectivity change, without the need of recomputing the whole network;
(c) evaluating the network reliability and availability.

In the present work, the combination of CA and MC sampling is further exploited to identify the network MCSs with respect to two-terminal reliability and to rank the network arcs according to their Fussell–Vesely importance. Several system configurations are sampled with MC sampling and for each one, the CA verifies the connection between source and target node. A further advantage of the approach is that it can efficiently be used in a changing environment where the connectivity of the links is modified. This computational advantage is of utmost importance in MC reliability evaluations, where a large number of connectivity evaluations are typically needed. Yet, the full computational potential offered by the proposed approach will be exploited if the analysis is carried out on parallel machines.

The proposed approach is tested on two cases in the literature [11] of increasing complexity. The paper is organized as follows. In section 2 some basic concepts on CA are presented. Section 3 summarizes their application in the procedure for testing the existence of source–target connectedness. In section 4, CA and MC sampling are combined for a direct assessment of the unreliability of a network system. Section 5 describes a novel procedure for identifying the MCSs of a network system and computing the Fussell–Vesely importance measure of its constituents. The results obtained on two literature case studies are also presented [11]. Finally, in section 6 some conclusions are drawn.

2 BASICS OF CA COMPUTING

CA are a class of spatially and temporally discrete mathematical systems characterized by local interaction and an inherently parallel form of evolution which unfolds on a discrete lattice of cells L. Each cell is a finite automaton which can assume one of a finite number of discrete values in a local value space S = {0, 1, 2, ..., k−1}. With reference to a three-dimensional lattice, the state at the discrete time t of the cell i,j,l, of coordinates x_{i,j,l} with i, j, l ∈ Z, is described by the state variable s_{i,j,l}(t).

In the original version, all cells are assumed to bear the same properties (homogeneous CA). The generic cell i,j,l interacts only with a fixed number n of cells that belong to its predefined local neighbourhood N_{i,j,l}. At the next discrete time t + 1, the cell i,j,l updates its state s_{i,j,l}(t + 1) according to a transition rule ϕ: S^n → S which is a function of the state variables at time t of the n cells in N_{i,j,l} viz

\[ s_{i,j,l}(t + 1) = \phi(s_{rs}(t), \quad rsp \in N_{i,j,l}) \quad (1) \]

In principle, there is no restriction on the size of the neighbourhood, except that it is the same for all cells. However, often in practice it is made only of adjacent cells because if it is too large, the complexity of the rule may be unacceptable (complexity usually grows exponentially fast with the number of cells in the neighborhood). Notice that the functional form of the rule is assumed to be the same everywhere in the lattice, i.e. there is no space index attached to ϕ: differences between what is happening at different locations are due only to differences in the values of the state variables of the local neighbourhood, not to the update rule. The rule is also homogeneous in time. One ‘iteration step’ of the dynamical evolution of the
CA is achieved after the simultaneous application of the rule $\phi$ to each cell in the lattice $L$.

Since the number of cells in a lattice has to be finite for practical purposes, boundary conditions must be assigned for the definition of the neighbourhoods of the cells at the lattice borders. Clearly, a cell belonging to the lattice boundary does not have the same neighbourhood as the internal cells. In order to define the behaviour of these cells, a different evolution rule can be considered which defines the appropriate neighbourhood. In this case, the information of the cell being, or not, at the boundary must be coded in the cell itself and depending on this information a different rule is selected. This approach allows defining several types of boundary with different behaviours.

An alternative to having different rules at the boundaries of the system is to extend the neighbourhood of the cells at the boundary. For instance, a commonly used type of boundary condition, termed 'periodic', consists of 'gluing' together the corresponding cells on opposite borders. For example, in a one-dimensional lattice of $M$ cells, the $M+1$th state variable, $s_{M+1}$, is identified with $s_1$ so that the lattice space wraps around itself into a circle. Other types of boundary condition can be defined by extending the lattice beyond its boundaries, adding a set of virtual cells.

CA offer a promising modern horizon for physical computations. Further details on the theory and applications behind this approach can be found in the specialized literature, for example references [7] and [12] to [14].

### 3 VERIFYING THE EXISTENCE OF SOURCE–TARGET CONNECTEDNESS BY CA

Consider a network of $n_n$ interconnected binary nodes whose function is to deliver a given throughput from a source node $S$ to a target node $T$. Ascertaining the connectivity of the network from source to target would require knowledge of the system cut or path sets or a depth-first procedure [1–3]. The required approach may be cumbersome and mathematically intensive, therefore in this paper the CA modelling paradigm is used; this paradigm offers the simplicity of the local interaction rules and its inherent parallelism for effectively addressing this complex computational problem.

Referring to an acyclic network with directed arcs (like that shown in Fig. 1), let each node $i$ be mapped into a spatial cell of a CA whose neighbourhood $N_i$ is the set of cells (i.e. network nodes) that provide their input to it (Table 1). The state variable $s_i$ of cell $i$ is binary, assuming the value of 1 when node $i$ is operating (active) and of 0 when not operating (passive). The transition rule governing the evolution of cell $i$ consists of the application of the logic operator OR ($\lor$) to the states of the nodes in its neighbourhood

$$s_i(t + 1) = s_p(t) \lor s_q(t) \lor \ldots \lor s_r(t), \quad p, q, \ldots, r \in N_i$$

(2)

where $t$ is the iteration step.

According to this rule, a cell is activated if there is at least one active cell in its neighbourhood, i.e. if it receives input from at least one of its connected nodes. At the beginning of the evolution, the source $S$ is the only active cell. The successive application of the transition rule to each node of the network generates the paths of transmission through the network. The computation ends when either the target node is activated or the process stagnates. Since the longest possible path from the active source $S$ to $T$ activates all the remaining $n_n$–1 nodes, the $S$–$T$ connection can be computed in $O(n_n)$ iterations, i.e. the computation ends after $n_n$–1 steps or the process stagnates.

The basic algorithm proceeds as follows [6]:

1. $t = 0$
2. Set all the cells state values to 0 (passive)
3. Set $s_i(0) = 1$ (source activated)
4. $t = t + 1$
5. Update all cells states by means of rule (2)
6. If $s_T(t) = 1$, stop (target activated). Else
7. If $t < m - 1$, go to 4. Else
8. $s_T(m-1) = 0$ (target passive): there is no connection path from $S$ to $T$

The above algorithm is also applicable to undirected networks.

![Fig. 1 Example network](example-network.png)

### Table 1 Neighbourhoods of the nodes in the example network of Fig. 1

<table>
<thead>
<tr>
<th>Cell</th>
<th>Neighbour</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Source)</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4 (Target)</td>
<td>1, 2, 3</td>
</tr>
</tbody>
</table>
4 DIRECT RELIABILITY ASSESSMENT OF A NETWORK SYSTEM BY A COMBINATION OF CA AND MC

Let \( G = (n, n_a) \) represent a stochastic binary network connecting a source node \( S \) to a sink node \( T \), with \( n \) being the number of nodes and \( n_a \) the number of arcs. For simplicity, but with no loss of generality, the former are considered infallible, i.e., always in their design state, whereas the generic arc \( ji \) connecting node \( j \) to \( i \) can be in one of two states, success or failure. The arc \( ji \) state variable, \( w_{ji} \), defines the operational state of the arc: \( w_{ji} = 1 \) for success and \( w_{ji} = 0 \) for failure. The corresponding occupancy probabilities are \( p_{ji} \), i.e., the probability of a successful connection from \( j \) to \( i \) (i.e. if they are also verified), in which case they are deleted and their associated counters are assigned to the counter of the newly found.

The transition rule governing the evolution of the generic cell \( i \) consists of the application of the logic operator OR (\( \lor \)) on the results of the logic operator AND (\( \land \)) applied to the states of the nodes in its neighbourhood and to the states of their connecting arcs with \( i 

\[ s_i(t + 1) = \{s_p(t) \land w_{pi}\} \lor \{s_q(t) \land w_{qi}\} \lor \ldots \lor \{s_r(t) \land w_{ri}\}, \quad p, q, \ldots, r \in N_i \] (3)

The network reliability \( p_{ST} \), i.e., the probability of a successful connection from \( S \) to \( T \), can then be computed (a) by MC (sampling a large number \( N \) of random realizations (MC trials) of the states of the connecting arcs), or (b) by CA (computing, for each realization, if a path from \( S \) to \( T \) exists: the ratio of the numbers of successful \( S \rightarrow T \) paths over the total number of realizations computed gives the network reliability).

The basic algorithm proceeds as follows:

1. \( n = 0 \)
2. \( n = n + 1 \) (\( n \)th MC trial)
3. Sample by MC a realization of the states of the connecting arcs \( w \)
4. Apply the previously illustrated CA algorithm for \( S \rightarrow T \) connectedness, to evaluate if there is a path from \( S \) to \( T \)
5. If a path exists, then update the counter of successful system states
6. If MC – iteration \( n < N \), go to 2. Else
7. Network reliability \( p_{ST} = \frac{\text{number of } S \rightarrow T \text{ successful paths}}{N} \)

5 IDENTIFYING THE MCS OF A NETWORK SYSTEM BY CA AND MC

Analogously to what is done in the procedure for estimating the network reliability (see section 4), at each trial the states of the network arcs are sampled by the MC algorithm from the cumulative distribution functions constructed from the known discrete occupancy probability distributions \( p_{ji}, q_{ji} \). In correspondence to each sampled system configuration, the source–target connectedness is verified by running the CA. If a failed configuration occurs, the following procedure is performed to identify the MCSs and compute their frequencies (Fig. 2).

1. The sampled, failed configuration is compared with the MCSs of lower order (the order of a cut set is the number of elements that compose it) already stored in a properly devised archive: if one of these MCSs is also verified, i.e. it is included in the sampled one, the counter associated to this cut set is incremented by one; otherwise,
2. the sampled configuration is compared with the cut sets of the same order in the archive to check if it is already present, in which case the associated counter is incremented by one; otherwise,
3. the sampled cut set is archived in the archive and it is compared with higher order cut sets in the archive to verify if it is included in any of them (i.e. if they are also verified), in which case they are deleted and their associated counters are assigned to the counter of the newly found.

Note that step 3 of the procedure guarantees that the cut sets kept in the archive at the end are minimal with respect to all cut sets configurations sampled. Of course, in principle the algorithm may not be exhaustive, depending on the finite size of the sample of network configurations. Indeed, the number of
trials necessary for the procedure to find the entire ensemble of MCSs depends on the complexity of the network. Yet, even a relatively low number of trials would allow identifying the most probable MCSs and, thus, the most critical vulnerabilities of the network system. At the end of the procedure, the criticality of each arc can be computed as the frequency of its occurrence, i.e. by dividing its corresponding counter by the number of MC trials performed. Furthermore, the criticality of each arc \( i \), in terms of the Fussell–Vesely importance measure with respect to network unreliability [4], can be computed as the ratio between the number of occurred cut sets containing \( i \) and the number of MC trials performed. This information can be of great practical aid to network designers and managers for tracing system bottlenecks and providing guidelines for effective actions of system improvement. The mathematical definition of the Fussell–Vesely importance measure is recalled in the Appendix. Furthermore, note that by recording in the archive the successful configurations sampled, the algorithm is also capable of determining the minimal path sets of the system.

Finally, the proposed approach is advantageous only if the number of trials carried out is smaller than the number of system states needed for the complete enumeration of the network. On the other hand, reducing the number of trials increases the estimation error, so that a compromise must be sought between accuracy and computing time.

5.1 A case study from the literature: the ARPA network

The proposed computational procedure has been validated first on a literature case study regarding a simple network named ARPA [11]. It has then been applied to the network depicted in Fig. 3 [11], where the arcs have been labelled sequentially for ease of reference and notation. All calculations have been performed by the Fortran code NUMA (network unreliability Monte Carlo analysis), developed at the Laboratorio di Analisi di Segnale ed Analisi di Rischio (LASAR, http://lasar.cesnef.polimi.it/) of the Department of Nuclear Engineering of the Politecnico di Milano.

The state occupancy probabilities are assumed equal for all arcs: \( p_{ij} = 0.75 \) for the functioning state and \( q_{ji} = 0.25 \) for the failed state. Table 2 reports the MCSs identified by computation and subsequently verified by manual enumeration. Since the failure probabilities are equal for all arcs, the frequencies of occurrence of MCSs of the same order are equal, within the MC estimation error. Obviously the most critical MCSs are those of lowest order, with only two arcs failed, which are the most probable. This would not be true, in general, if lower order cut sets were made up of more reliable components, to cope with their higher importance and criticality with respect to the vulnerabilities of the network.

For the same previous argument on the equality of the arc failure probability, the importance of the arcs is dependent only on their position in the network system, an arc near to the target or source node being more critical than those in the middle of the network – since a failure of the former is more likely to cause a network failure. This is, indeed, the finding of the computational procedure, as reported in Table 3. The network reliability value estimated based on 500 random samples of the system state and their comparison with the list of analytically identified MCSs is \( (8.260 \pm 0.152) \times 10^{-3} \), in agreement with the result obtained by the CA-MC computational procedure of section 4.

![Fig. 3 ARPA network](image-url)
5.2 A second case study from the literature [11]

The MCS identification procedure has been applied to the more complex network of Fig. 4 [11]. Again, the arcs have been labelled sequentially for ease of reference and notation. The state occupancy probabilities of each arc are reported in Table 4. Based on the identified MCSs, the application of the MC sampling procedure with $10^5$ MC trials gives an estimated network unreliability of $(6.621 \pm 0.080) \times 10^{-2}$, in agreement with the results of the CA–MC computational procedure of section 4. The procedure, run with $10^7$ MC trials, finds all 111 MCSs [11]. The most probable MCSs are presented in Table 5, in order of decreasing frequency. The first-order MCS is due to the structure of the network: the target node is connected to the remaining part of the network only through arc 17. This is a rather anomalous design since the entire network relies only on one component for its final connection to the target. Obviously, in any design, the presence of an MCS of order 1 should be avoided resorting to redundancy of connections; alternatively, its probability could be reduced closer to the values of higher-order cut sets by placing a very reliable component in such a critical position, so as to have a more balanced design [15].

Continuing with the analysis of the results, the use of more reliable components in low-order cut sets accounts for the fact that some order-3 MCSs are less probable than order-4, -5, or even -6 MCSs. For example, arcs 14, 16, and 20 in MCS 20 (see Fig. 5), which are gates to arc 17 and, thus, to the target node, are chosen to be very reliable; in contrast, arcs 2, 3, 5, 11, 12, and 13 in MCS 7 (see Fig. 6), which are coupled with redundant components, are less reliable.

Finally, in Table 6 the Fussell–Vesely importance measures of the arcs are given in decreasing order. Arcs near to the target and source nodes – such as arcs 2, 3, 4, 14, and 16 – are more present in the network of a case study from the literature [11]

![Fig. 4](image)

Table 4 State occupancy probabilities of the network arcs [11]

<table>
<thead>
<tr>
<th>Arc</th>
<th>State probability</th>
<th>Arc</th>
<th>State probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.8221</td>
<td>2</td>
<td>0.7086</td>
</tr>
<tr>
<td>3</td>
<td>0.7086</td>
<td>4</td>
<td>0.8156</td>
</tr>
<tr>
<td>5</td>
<td>0.6771</td>
<td>6</td>
<td>0.7542</td>
</tr>
<tr>
<td>7</td>
<td>0.6446</td>
<td>8</td>
<td>0.7973</td>
</tr>
<tr>
<td>9</td>
<td>0.8864</td>
<td>10</td>
<td>0.6171</td>
</tr>
<tr>
<td>11</td>
<td>0.6830</td>
<td>12</td>
<td>0.6272</td>
</tr>
<tr>
<td>13</td>
<td>0.6048</td>
<td>14</td>
<td>0.9494</td>
</tr>
<tr>
<td>15</td>
<td>0.8501</td>
<td>16</td>
<td>0.8797</td>
</tr>
<tr>
<td>17</td>
<td>0.9586</td>
<td>18</td>
<td>0.9537</td>
</tr>
<tr>
<td>19</td>
<td>0.5440</td>
<td>20</td>
<td>0.8893</td>
</tr>
<tr>
<td>21</td>
<td>0.8245</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Fig. 5](image)

Table 5 Main MCSs of the network of Fig. 4: the network of a case study from the literature [11]

<table>
<thead>
<tr>
<th>Order</th>
<th>MCSs</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2, 3, 4, 5</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>13, 14, 16</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>3, 4, 7, 16</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1, 2, 3, 4</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2, 3, 4, 16</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>2, 3, 5, 11, 12, 13</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>5, 13, 14, 21</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>7, 9, 13, 15, 16</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>4, 7, 9, 10, 16</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>4, 14, 16</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>13, 14, 18</td>
</tr>
</tbody>
</table>
Besides the role of the arc in the system logic, failure probability is, obviously, the other relevant factor for arc importance. Hence, arcs 13 and 5, although in an internal position of the network, are ranked more important than arc 14, which is closer to the target, because they are among the less reliable of the whole network. Furthermore, when arcs are connected in a series logic, e.g. arcs 16 and 18, it is obvious that the main contribution to unreliability is given by the less reliable one and, correspondingly, the importance measure ranks arc 16 higher than arc 18.

6 CONCLUSIONS

A challenging difficulty in the assessment of the safety, vulnerability, reliability, and availability characteristics of complex network systems resides in the identification of the MCSs. Classical approaches lead to NP-hard problems with cumbersome and mathematically intensive methods of solution. In this work, an efficient computational procedure, based on CA and MC sampling, has been proposed for identifying the MCSs of complex interconnected networks. The procedure also allows the ranking of the arcs in terms of their Fussell–Vesely importance. This information on the criticality of the arcs constituting the network may be exploited by the designers and managers of the system to deal with operation bottlenecks effectively and to establish proper actions of system improvement. The results from two literature case studies of increasing complexity have demonstrated the success of the proposed approach. However, the computational power of the proposed approach will be exploited in full if the analysis is carried out on parallel machines.

Future investigations are planned in order to verify the opportunity of using biased MC sampling techniques for improving the computational efficiency when analysing the vulnerabilities of highly reliable networks that would otherwise require a very large sample of system configurations.

REFERENCES

APPENDIX

Fussell–Vesely importance measure

Let:

\[ F(t) = \text{system failure probability or unreliability, or more generally risk, at time } t \]

\[ F_j^-(t) = \text{system failure probability or unreliability when component } j \text{ remains in the functioning state throughout the time interval } [0, t]. \]

It represents the maximum reduction in risk if component \( j \) is considered perfect, i.e. always in the functioning state.

The Fussell–Vesely importance measure \( FV_j(t) \) of component \( j \) at time \( t \) is defined as

\[
FV_j(t) = \frac{F(t) - F_j^-(t)}{F(t)}
\]

By definition, the Fussell–Vesely measure represents the maximum fractional decrement in risk \( F(t) - F_j^-(t) \) achievable when component \( j \) is always available. Note that the Fussell–Vesely importance measure of component \( j \) can be alternatively, and equivalently, defined as the ratio of the probability of the union of all the MCSs containing component \( j \) to the actual value of the risk.