Chapter 2
Applications

Abstract Sections 2.1–2.4 present numerical illustrations and applications of the theory developed in Chap. 1. Section 2.1 presents reliability calculations for H_4 network. The network has 16 nodes, 32 edges and two sets of terminals, T_1 and T_2. An edge $e = (a, b)$ in state *up* provides high communication speed between a and b. If this edge is in state *mid*, the $a \leftrightarrow b$ communication goes with reduced speed; *down* state for an edge means that this edge is erased. Edge state is chosen randomly and independently according to probabilities p_2, p_1, p_0 for *up, mid* and *down* state, respectively. System *UP* state is defined as the existence of high speed communication between nodes of T_1 and existence of a path of operational edges between any pair of nodes of T_2. We present data on network reliability and characterize the ternary D-spectrum numerically and graphically. Section 2.2 considers a stochastic source-terminal flow network problem for a dodecahedron network. In this network, an edge $e = (e, b)$ is a *pair* of directed links for $a \to b$ and $b \to a$ flows. Each link has capacity 6, 3, or 0 for *up*, *mid* and *down* state, respectively. The network has two *DOWN* states, *DOWN*1 and *DOWN*2, for the flow less than L_1 or L_2, respectively ($L_1 > L_2$). We present data on network reliability for various values of edge state probabilities $\mathbf{p} = (p_2, p_1, p_0)$. Section 2.3 is an example of a rectangular grid network with 100 nodes and 180 edges. Components subject to failure are the nodes. If a node is *down* all edges adjacent to it are erased and the node gets isolated. If a node is in *mid* state, it has only horizontal or vertical edges, depending on the position of the node. For this network we calculate the probability that the largest connected node set (an analogue to a "giant" component) has less than L nodes. Section 2.4 analyzes edges importance data for H_4 network. Section 2.5 deals with networks having independent and *nonidentical* components. Component i is in state *up, mid* and *down*, with probability $p_2^{(i)}, p_1^{(i)}$ and $p_0^{(i)}$, respectively. In this situation, different failure sets with the same number of components in *up, mid, down* have different probabilistic weights and it is not possible to use the ternary spectrum technique. For calculating network reliability, we present an efficient and accurate Monte Carlo method based on a modification of M.V. Lomonosov's evolution algorithm [3, 4]. Its action is illustrated by examples of a flow network and a grid network.

I. Gertsbakh et al., *Ternary Networks*, SpringerBriefs in Electrical and Computer Engineering, DOI: 10.1007/978-3-319-06440-6_2,

Keywords Dodecahedron network · Hypercube network · Flow-network · Grid network · Giant component · Component importance · Nonidentical components · Evolution algorithm

2.1 Hypercube Network

We consider fourth order hypercube H_4. It has 16 nodes and 32 edges, see Fig. 2.1. Each edge, independently of other edges, can be in three states: *up*, *mid* and *down* with respective probabilities p_2, p_1 and p_0, $p_2 + p_1 + p_0 = 1$. States *up* and *mid* provide high and medium connection speed, respectively. Edge *down* state means loss of connection. We say that node set T_1 is *strongly connected* if any pair of nodes from this set is connected by a path consisting only of edges providing high speed connection. We say that node set T_2 is *weakly connected* if any pair of nodes in this set is connected by a path of operational edges. We define *UP* state of our system as the presence of strong connection between nodes $T_1 = \{0, 3, 7, 12\}$ *and* weak connection between nodes of $T_2 = \{1, 4, 6, 15\}$, see Fig. 2.1. System *DOWN* state is absence of strong connection for T_1 *or* absence of weak connection for T_2, or both.

Our main tool for computing $P(DOWN)$ is the ternary D-spectrum. For this purpose we used Monte Carlo procedure, the algorithmic details of which were described in Sect. 1.6. Now let us mention some specific properties of the ternary spectrum in this example. First, it is easy to check from Fig. 2.1 that the minimal number of edges providing strong connection for T_1 equals 6. Therefore, if we have $r \leq 5$ edges *up* and other edges in *mid*, the system is *DOWN*. Thus, the ternary spectrum starts with $F_6(x), x = 1, 2, ..., 26$. In the course of simulation it is revealed that $r_{\max} = 28$, i.e. if 29 or more edges are *up*, the system can not fail. Therefore the ternary spectrum is the collection $F_6(x), F_7(x), ..., F_{28}(x)$. Figure 2.2 presents a sample of graphs of $\{F_i(x)\}$.

Data on system *DOWN* probability is presented in Table 2.1. We see that for $p_2 \leq 0.5$ and p_1 in the range $[0.1, 0.4]$, the system has low reliability—$P(DOWN) \geq 0.4$. To provide reliability of about 0.85 or higher, p_2 should exceed 0.6 and p_1 must not exceed 0.15.

We will often characterize the accuracy of estimating *DOWN* probability by so-called relative error (RE). It is calculated as follows. The whole simulation experiment based on M replications for estimating the ternary spectrum and for calculating $P(DOWN) = \ell$ is repeated 10 times. Let the result of the i-th experiment be denoted by ℓ_i. Denote by S the estimate of the standard deviation computed as

$$S = [\sum_{i=1}^{10} (\ell_i - \ell_0)^2/9]^{0.5}, \tag{2.1}$$

where $\ell_0 = \sum_{i=1}^{10} \ell_i/10$. Then the RE is defined as

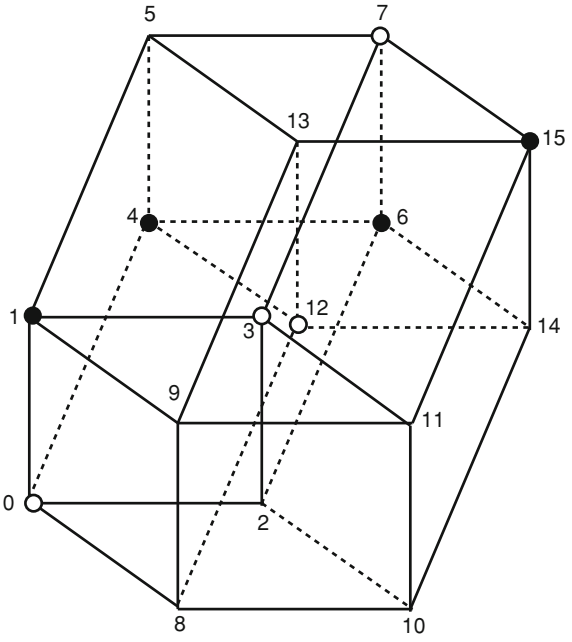

Fig. 2.1 Hypercube of order four. $T_1 = \{0, 3, 7, 12\}$, $T_2 = \{1, 4, 6, 15\}$

$$RE = \frac{S}{\ell_0}. \tag{2.2}$$

Our calculations reveal that for M =100,000 and $p_2 = 0.6$, $p_1 = 0.3$, the $RE \approx 0.01$. It means that the absolute error in estimating the *DOWN* probability is about $\pm$**0.002**. For $p_2 = p_1 = 0.3$, $P(DOWN)$ is close to 1, and the $RE \approx 0.0003$, which means that the absolute error is about $\pm$**0.0003**.

2.2 Flow Network

The network in this example is a dodecahedron graph shown on Fig. 2.2. Nodes 1 and 10 are the source and the sink, respectively. The edges incident to the source $s = 1$ allow the flow to be directed only outward as $(s \mapsto 2)$, $(s \mapsto 16)$, $(s \mapsto 5)$, and all edges incident to sink $t = 10$ allow the flow to go only in the direction of the sink. All other edges shown on Fig. 2.2, represent a *pair* of independent links, allowing the flow to go in opposite directions. For example, there are *two* directed edges connecting nodes 2 and 15: $(2 \mapsto 15)$ and $(15 \mapsto 2)$. There are in total $30 + 24 = 54$ edges in this flow network (Fig. 2.3).

The components subject to failure are the edges. All edges can be in three states: *up, mid,* and *down.* An edge in *up* has maximal flow capacity 6. An edge in *mid*

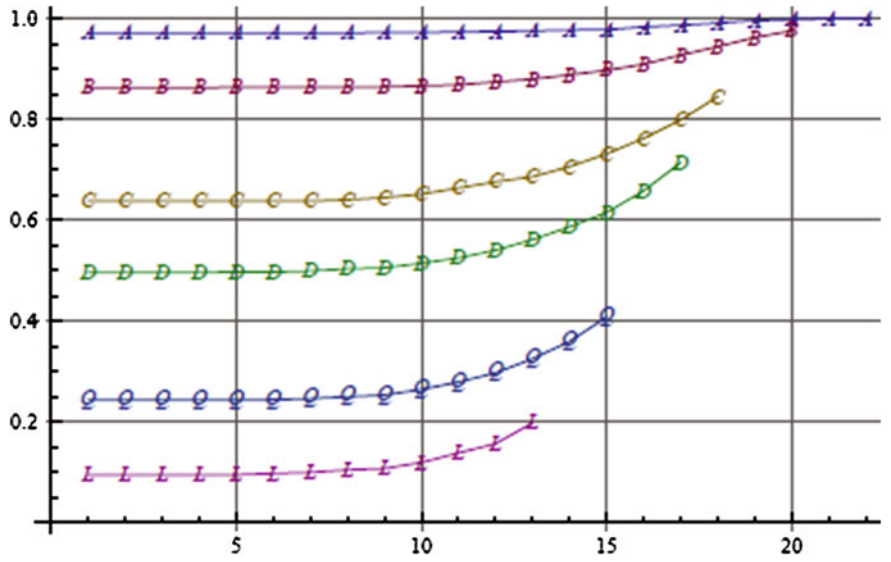

Fig. 2.2 Sample of $F_i(x)$. From *top* to *bottom*: $F_{10}, F_{12}, F_{14}, F_{15}, F_{17}, F_{19}$

Table 2.1 Network *DOWN* Probability *P(DOWN)*

p_1	$p_2 = 0.1$	$p_2 = 0.2$	$p_2 = 0.3$	$p_2 = 0.4$	$p_2 = 0.5$	$p_2 = 0.6$	$p_2 = 0.7$	$p_2 = 0.8$
0.1	0.9999	0.9976	0.9563	0.7790	0.4661	0.1887	0.0500	0.0073
0.2	0.9999	0.9957	0.9409	0.7412	0.4250	0.1660	0.0437	–
0.3	0.9959	0.9940	0.9312	0.7299	0.4099	0.1586	–	–
0.4	0.9928	0.9928	0.9267	0.7161	0.3981	–	–	–

state has maximal flow capacity 3, and an edge in *down* state has zero capacity (does not exist). The "parallel" edges, like (5, 6) and (6, 5) are *independent* with respect to their state. So, it may happen that edge (5, 6) will have capacity 6, and edge (6, 5)—capacity 3.

Obviously, the maximal flow equals 18. We will introduce two threshold values for the random flow $L_2 = 4.5$ and $L_1 = 10$ and define the following *DOWN* states for the network:

$$DOWN2 : \text{maximal flow}\, MF \leq 4.5,$$

$$DOWN1 : \text{maximal flow}\, MF \leq 10.$$

Suppose that all edges of the network are *up*, and therefore the network is in *UP* state and the $MF = 18$. Let edges start to fail in random order. There will be an instant when the *MF* will become smaller or equal 10. In that case we say that the network entered state *DOWN1*. Suppose the edges continue to fail and the *MF* becomes $\leq$4.5. Then, by our definition, the network will be in state *DOWN2*. So, the $MF \in [18, 10)$

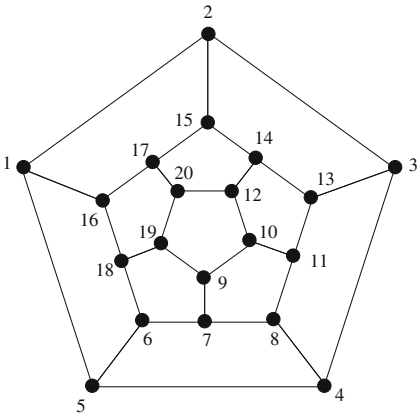

Fig. 2.3 The dodecahedron network. Nodes 1 and 10 are the source and the sink, respectively. All edges except incident to 1 and 10, represent a pair of independent edges

means the *UP* state, $4.5 < MF \leq 10$ means *DOWN1* state, and for $MF \leq 4.5$ the network is in *DOWN2* state.

We just described a ternary system which is not binary: it can be in three states. This extension does not complicate the formal part of our exposition. In this new situation we have not a single ternary D-spectrum, but *two* ternary D-spectra: one for the state $DOWN1$ and another—for state $DOWN2$. When calculating one of these spectra, we can completely ignore the presence of another.

Let us describe details the simulation of the above spectra in details. For each r we generated N =100,000 permutations. It guarantees quite good accuracy in computing network *DOWN* probabilities. For example, the estimate of $P(DOWN1)$ for $p_2 = 0.6, p_1 = 0.3, p_0 = 0.1$ is

$$P(DOWN1) = \mathbf{0.4407}$$

with relative error (RE) is about 0.002. It corresponds to an absolute error $\delta \approx 0.0009$. We calculated the same probability using crude Monte Carlo (CMC) based on 1,000,000 replications and the result was

$$P_{CMC}(DOWN1) = \mathbf{0.4416}.$$

Table 2.2 presents the *DOWN1* probabilities for a sample of seven different combinations of p_2, p_1, p_0 probabilities.

We see from this table that even the "most reliable" combination $p_2 = 0.7, p_1 = 0.2, p_0 = 0.1$ has rather large value of *DOWN1* probability:

$$P(DOWN1) = \mathbf{0.3243}.$$

Table 2.2 Flow network *DOWN1* and *DOWN2* probabilities

p_2	p_1	p_0	$P(DOWN1)$	$P(DOWN2)$
0.7	0.2	0.1	0.3243	0.0021
0.6	0.2	0.2	0.7042	0.1481
0.5	0.2	0.3	0.9274	0.4384
0.4	0.2	0.4	0.9908	0.7556
0.6	0.3	0.1	0.4407	0.2691
0.5	0.3	0.2	0.8064	0.1890
0.4	0.3	0.3	0.9654	0.5110

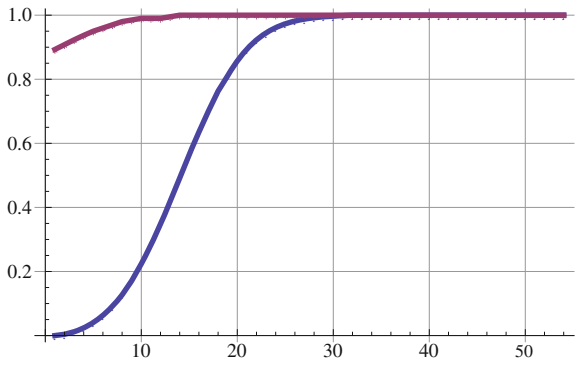

Fig. 2.4 $F_5(x)$ for $MF \leq 4.5$ (*lower curve*) and $MF \leq 10$ (*upper curve*)

If we change the *DOWN* definition to *DOWN2* meaning that $P(DOWN2) = P(MF \leq 4.5)$, the network becomes considerably more reliable, as we see from the right column of the Table.

The Monte Carlo estimation of ternary D-spectra for $MF = 4.5$ and $MF = 10$ (with 100,000 replications) took about 180 s of CPU time. It should be noted that the search of the anchor position in a permutation was made by using bisection algorithm, see [8].

For $MF = 10$, $r = 0$ and $r = 1$, the network is always *DOWN*. (This was called in Sect. 1.2 as existence permutations of type A). We remind that for no components in *up* and all components in *mid*, or only one component in *up* and remaining in *mid*, the network is always *DOWN*. It is easy to understand it: if all edges are in *mid*, the maximal flow can not exceed 9; the same is true if only one of the edges is in *up*. It turns out that $r_{\max} = 52$, i.e. when 53 edges are *up*, the system is always *UP*. The upper curves on Figs. 2.4 and 2.5 show the graphs of $F_5(x)$ and $F_{21}(x)$ for $MF = 10$.

For $MF = 4.5$ the principal behavior of the $F_r(x)$ curves is similar, see the lower graphs on Figs. 2.4 and 2.5. Contrary to the previous case, permutations of type A do not exist here. Permutations of type B appear for $r = 52$ and $r = 53$.

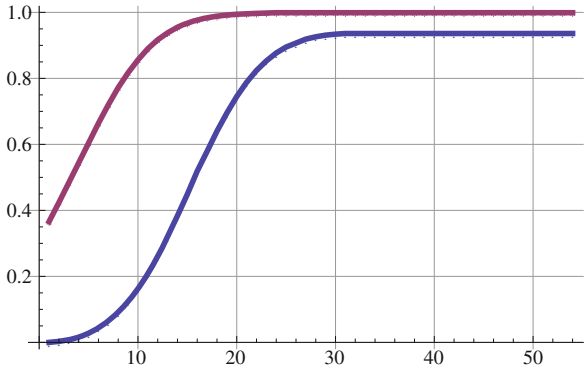

Fig. 2.5 $F_{21}(x)$ for $MF \leq 4.5$ (*lower curve*) and $MF \leq 10$ (*upper curve*)

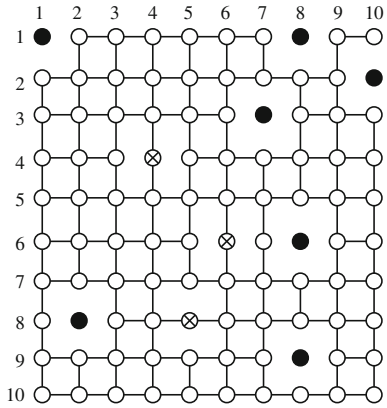

Fig. 2.6 Rectangular grid with 100 nodes and 180 edges. Bold nodes are *down*. Nodes marked by x are in *mid*

2.3 Survival of a Ternary Grid Network

In this section we will consider a 10×10 grid network with nodes having three states:

(i) *down*—all four edges adjacent to this node are erased.
(ii) *mid*—erased are two out of four edges adjacent to the node. Node is called *even* if the sum of its row and column numbers in the grid are even. Otherwise it is called *odd*. At even node in *mid*, the adjacent *horizontal* edges are erased; at odd node—vertical adjacent edges are erased.
(iii) *up*—all edges incident to this node are initially intact. These edges, however may be erased if neighboring nodes are in *down* or in *mid*.

Figure 2.6 shows the rectangular grid with some nodes in *down* and *mid*.

Table 2.3 Fragments of ternary spectrum $F_r(x)$ for $L_1 = 80$

r	$x = 0$	$x = 5$	$x = 10$	$x = 15$	$x = 20$
0	1	1	1	1	1
10	1	1	1	1	1
20	0.9999	0.9999	1	1	1
30	0.9755	0.9910	0.9994	1	1
40	0.7171	0.8218	0.9553	0.9987	1
45	0.4913	0.6403	0.8400	0.9842	1
50	0.2805	0.3798	0.6326	0.9465	1
60	0.0514	0.0839	0.1954	0.6353	1
70	0.0035	0.0074	0.0274	0.2121	1
80	0.0000	0.0001	0.0021	0.0358	1
90	0	0	0	0	0

Definition 2.3.1 Network component.
A subset of nodes V_0 is called a *component* of the network if there is at least one path from each node $v \in V_0$ to each other node in V_0 and no other node in $V \setminus V_0$ can be added to V_0 while preserving this property. #

Network proper functioning, e.g. information delivery, power supply etc can be guaranteed only if its *maximal size* component is large enough [7, 11]. Accordingly, we will define network *UP* state if its maximal component exceeds some critical level $L_{\max}$. We will consider two values for this level: $L_{\max} = L_1 = 80$ and $L_{\max} = L_2 = 50$. Accordingly, we define two versions of network *UP* state:

$$UP1 : \text{maximal component size exceeds } L_{\max} = L_1 = 80$$

and

$$UP2 : \text{maximal component size exceeds } L_{\max} = L_2 = 50.$$

Correspondingly we have network $DOWN1$ state as the complement to $UP1$ and $DOWN2$ as the complement to $UP2$.

Let us consider the ternary spectra for the network. Table 2.3 shows a fragment of the spectrum for $L_1 = 80$, for a selection of r values (column 1) and x values. The ternary spectrum was calculated using Monte Carlo simulation with 100,000 replications for each r value. The total simulation time took 179.5 s of CPU time.

Let us remind that $F_r(x)$ presented in the Table 2.3 is the probability that the network is in *DOWN1* if r of its randomly chosen nodes are *up* and x of its randomly chosen nodes are in *down*, and therefore $(n - r - x)$ components are in *mid*. We see from the table that for $r = 0$ and $r = 10$ we have the situation described in Sect. 1.2 as permutations of type A: when r components are *up* and all other $n - r$ components are in *mid*, the network is in *DOWN1*.

For $r \geq 90$, network is in *UP1* for all x values. We see that for $x = 20$ the network is in *DOWN1* with probability 1, as it should be by the definition of *DOWN1*. It is interesting that for $r = 70$, $P(DOWN1)$ changes from 0.0274 to 1 in a narrow

Table 2.4 *DOWN1* probability for grid network

p_2	p_1	p_0	$P(DOWN1)$
0.7	0.2	0.1	**0.0920**
0.4	0.2	0.4	1
0.6	0.3	0.1	**0.2889**
0.5	0.3	0.2	0.9557
0.4	0.3	0.3	0.9999

Table 2.5 Fragments of ternary spectrum $F_r(x)$ for $L_2 = 50$

r	$x = 0$	$x = 5$	$x = 10$	$x = 15$	$x = 20$	$x = 30$	$x = 40$	$x = 50$
0	1	1	1	1	1	1	1	1
10	0.9999	0.9999	1	1	1	1	1	1
20	0.9819	0.9874	0.9956	0.9990	0.9998	1	1	1
30	0.7356	0.8104	0.9049	0.9622	0.9878	0.9996	1	1
40	0.2756	0.3379	0.5020	0.6752	0.8161	0.9719	0.9996	1
45	0.1323	0.1895	0.3279	0.4944	0.8090	0.9792	0.9999	1
50	0.0576	0.0084	0.1679	0.2906	0.4418	0.7848	0.9835	1
60	0.0081	0.0120	0.0293	0.0625	0.1186	0.3688	0.8104	–
70	0.0005	0.0009	0.0026	0.0063	0.0141	0.8734	–	–
80	0.0000	0.0000	0.0004	0.0006	0.0006	–	–	–
85–99	0	0	0	0	0			

interval $x \in [10-20]$, which reminds the phase transition phenomena in percolation. Table 2.4 presents *DOWN1* probabilities for five different combinations of p_2, p_1, p_0.

The results shown in bold were checked by crude Monte Carlo (CMC) simulation with 1,000,000 replications. For example, the CMC estimate is 0.09198, i.e. differs from our result by 0.00004.

Table 2.5 presents the ternary spectrum for $L_{max} = L_1 = 50$. Similarly to the previous case, for $r = 0, ..., 9$, all $F_r(x)$ values are equal 1. The $DOWN2$ probabilities are smaller than the corresponding values $P(DOWN1)$, as expected.

For r in the interval 85–99, we have $F_r(x) = 0$ (permutation of type B) which means that the network is always *UP2*. $F_{70}(x)$ has a slow increase of 0.0005– 0.0141 when x changes from 0 to 20 and a very rapid increase to 0.8734 when x approaches 30. It also reminds a phase transition. Table 2.6 presents a collection of $P(DOWN2)$ probabilities for a selection of p_2, p_1, p_0 values.

It is seen that to provide $P(DOWN2) < 0.1$, the probabilities for a node to be in *up* and *mid* respectively, should be close to (0.7, 0.2). For that case the average number of *up* nodes will be near 70, and *down* nodes—around 10. This is in agreement with data shown for $r = 70$ in Table 2.5.

Table 2.6 Grid network *DOWN2* probability *P(DOWN2)*

p_2	p_1	p_0	$P(DOWN2)$
0.7	0.2	0.1	0.0073
0.6	0.2	0.2	0.1738
0.5	0.2	0.3	0.7236
0.4	0.2	0.4	0.9862
0.6	0.3	0.1	0.0520
0.5	0.3	0.2	0.4598
0.4	0.3	0.3	0.9297

2.4 Component Importance in Hypercube H_4 Network

Let us remind that network *UP* state was defined as the presence of strong connectivity of $T_1 = \{0, 3, 7, 12\}$ and weak connectivity of $T_2 = \{1, 4, 6, 15\}$, see Fig. 2.1. Table 2.7 presents the *up-down* edge importance indices computed by formula (1.5.3). We remind also, that this formula represents the partial derivative of the reliability function with respect to p_2 when edge *mid* probability p_1 remains unchanged (p_2 increases on account of p_0). This definition copies in fact the Birnbaum Importance Measure (BIM) used in binary systems, see [1, 2, 6].

Three variants of $\mathbf{p}$ were considered: $\mathbf{p} = (0.6, 0.3, 0.1)$, $\mathbf{p} = (0.5, 0.3, 0.2)$ and $\mathbf{p} = (0.4, 0.3, 0.3)$, see columns 3, 4, 5 in the table.

Figure 2.7 presents the data of this table in a form convenient for visual analysis of component TIM's.

It is clearly seen that the edges have practically the same ranking for all three versions of $\mathbf{p}$. The most important edge is 11 which connects nodes of sets T_1 and T_2. The second group of edges consists of 14 edges which have one of its ends in the set T_1. These edges are shown by bold in Table 2.7. The importance values for this category are in the range [0.074–0.100]. The third group constitute the remaining edges with importance measures (for $\mathbf{p} = \{0.6, 0.3, 0.1\}$) in the range [0.013–0.034].

If we think about replacing an edge by a more reliable one, the best candidate are edge 11 and edges in the second group. If, for example, we replace edge 11 and 15 having $p_2 = 0.6$ by an edge with $p_2 = 0.7$ (and $p_0 = 0$, respectively), we might expect reliability increase by $0.1 \cdot (0.150 + 0.097) \approx 0.025$, or equivalently, $P(DOWN)$ decrease by the same quantity. As it is seen from Table 2.1, for $p_2 = 0.6, p_0 = 0.3$, $P(DOWN) = 0.1586$. Therefore the above edge replacement will bring $P(DOWN)$ to ≈ 0.1336, which is a quite significant improvement.

Table 2.8 presents the *down-mid* component importance values (see Definition 1.5.1-B).

First of all, it is seen that the values in this table are significantly smaller than the corresponding values in the previous table. This is expected, since the *up-down* importance reflects the effect from reinforcing a component in *up* on the account of reducing p_0. It is bigger than the *mid-down* importance calculated for similar reinforcement of *mid* on the account of reducing p_0. Second conclusion is that the

Table 2.7 Edge *up-down* TIM's for H_4 by (1.5.3), $T_1 = (0, 3, 7, 12)$, $T_2 = (1, 4, 6, 15)$

i	Edge	$\mathbf{p} = (0.6, 0.3, 0.1)$	$\mathbf{p} = (0.5, 0.3, 0.2)$	$\mathbf{p} = (0.4, 0.3, 0.3)$
1	(0, 1)	0.084	0.145	0.143
2	(0, 2)	0.076	0.133	0.130
3	(0, 4)	0.091	0.152	0.142
4	(0, 8)	0.083	0.131	0.117
5	(1, 3)	0.092	0.144	0.122
6	(1, 5)	0.025	0.066	0.078
7	(1, 9)	0.032	0.067	0.067
8	(2, 3)	0.089	0.144	0.131
9	(2, 6)	0.028	0.061	0.055
10	(2, 10)	0.018	0.048	0.056
11	(3, 7)	0.150	0.252	0.237
12	(3, 11)	0.074	0.104	0.077
13	(4, 5)	0.024	0.062	0.077
14	(4, 6)	0.033	0.075	0.079
15	(4, 12)	0.097	0.149	0.133
16	(5, 7)	0.075	0.108	0.083
17	(5, 13)	0.025	0.053	0.052
18	(6, 7)	0.088	0.132	0.112
19	(6, 14)	0.035	0.071	0.072
20	(7, 15)	0.082	0.118	0.087
21	(8, 9)	0.029	0.061	0.070
22	(8, 10)	0.034	0.063	0.063
23	(8, 12)	0.091	0.140	0.121
24	(9, 11)	0.019	0.042	0.040
25	(9, 13)	0.025	0.050	0.050
26	(10, 11)	0.013	0.029	0.037
27	(10, 14)	0.022	0.048	0.046
28	(11, 15)	0.024	0.051	0.050
29	(12, 13)	0.082	0.116	0.090
30	(12, 14)	0.076	0.112	0.089
31	(13, 15)	0.027	0.048	0.039
32	(14, 15)	0.030	0.072	0.090

edge ranking remains practically the same for all three sets of *up, mid* and *down* probabilities, see Fig. 2.8.

If we check the edge positions in H_4, we will see that the largest values of *TIMs* have edges adjacent to the nodes of T_2. These edges are most important for providing connectivity of the nodes in this set (Fig. 2.8).

Remark Suppose we are interested in estimating the increase in system reliability achieved by replacing edge 5 having $p_2 = 0.5, p_1 = 0.3, p_0 = 0.2$ by an edge which has $p_2 = 0.8, p_1 = 0, p_0 = 0.2$. This is a replacement of type *mid-up* that leaves p_0

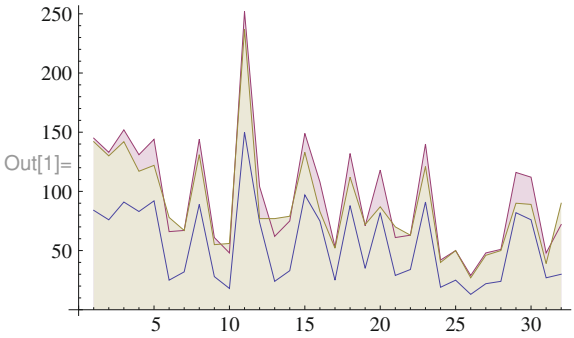

Fig. 2.7 *Up–down* TIM's for edges of H_4 multiplied by a factor 1000. Edge numbers are shown on the x-axis

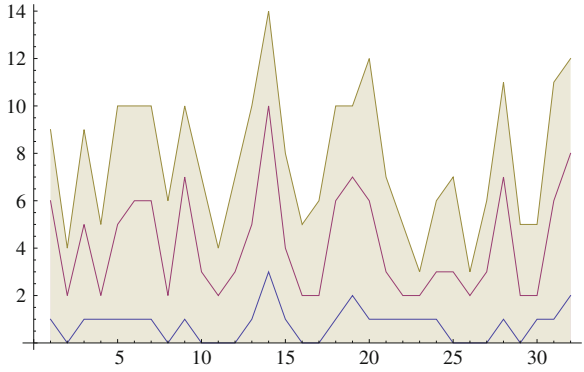

Fig. 2.8 *Mid–down* TIM's for edges of H_4 multiplied by a factor 1000. Edge numbers are on x-axis

unchanged. Following Definition 1.5.1-C, we see that we have to take the difference between *up-down* importance in Table 2.7 and *mid-down* importance in Table 2.8. We obtain for edge 5 and $p_0 = 0.2$ (column 4) the value $0.144 - 0.005 = 0.139$. For the increase of p_2 by $0.8 - 0.5 = 0.3$, we can expect the drop of *DOWN* probability by $\delta \approx 0.139 \cdot 0.3 = 0.0417$.

In general, estimation of the effect of component reinforcement on network reliability for ternary network is quite similar to the procedure described in [5], Chapter 2 for binary networks.

Table 2.8 Edge *down–mid* TIM's for H_4 by (1.5.4), $T_1 = (0, 3, 7, 12)$, $T_2 = (1, 4, 6, 15)$

i	Edge	$\mathbf{p} = (0.6, 0.3, 0.1)$	$\mathbf{p} = (0.5, 0.3, 0.2)$	$\mathbf{p} = (0.4, 0.3, 0.3)$
1	(0, 1)	0.001	0.006	0.009
2	(0, 2)	0.000	0.002	0.004
3	(0, 4)	0.001	0.005	0.009
4	(0, 8)	0.001	0.002	0.005
5	(1, 3)	0.001	0.005	0.010
6	(1, 5)	0.001	0.006	0.010
7	(1, 9)	0.001	0.006	0.010
8	(2, 3)	0.000	0.002	0.006
9	(2, 6)	0.001	0.007	0.010
10	(2, 10)	0.000	0.003	0.007
11	(3, 7)	0.000	0.002	0.004
12	(3, 11)	0.000	0.003	0.007
13	(4, 5)	0.001	0.005	0.010
14	(4, 6)	0.003	0.010	0.014
15	(4, 12)	0.001	0.004	0.008
16	(5, 7)	0.000	0.002	0.005
17	(5, 13)	0.000	0.002	0.006
18	(6, 7)	0.001	0.006	0.010
19	(6, 14)	0.002	0.007	0.010
20	(7, 15)	0.001	0.006	0.012
21	(8, 9)	0.001	0.003	0.007
22	(8, 10))	0.001	0.002	0.005
23	(8, 12)	0.001	0.002	0.003
24	(9, 11)	0.001	0.003	0.006
25	(9, 13)	0.000	0.003	0.007
26	(10, 11)	0.000	0.002	0.003
27	(10, 14)	0.000	0.003	0.006
28	(11, 15)	0.001	0.007	0.011
29	(12, 13)	0.000	0.002	0.005
30	(12, 14)	0.001	0.002	0.005
31	(13, 15)	0.001	0.006	0.011
32	(14, 15)	0.002	0.008	0.012

2.5 Evolution Process for Ternary Network

2.5.1 Lomonosov's Evolution Process with Merging

Let us remind that all calculations of network reliability (where the *UP* state was defined as the existence of strong and weak connectivity, or the existence of an $s - t$ flow, or as presence of largest component above critical size) were made under two important conditions: all components subject to failure are *identical* and stochastically *independent*. Below we will describe an ingenious Monte Carlo algorithm of

M.V. Lomonosov for estimating the probability of network connectivity which will allow to treat networks with independent components which have *different* failure probabilities. The first publication of this algorithm is dated 1991 [3], see also its detailed description in [4], Chap. 9.

Originally, Lomonosov's algorithms (LA) was designed for binary networks. We will first describe this version of LA for networks whose components subject to failures are binary edges. Later, in Sect. 2.5.2 we will describe an extension of the LA to ternary components.

Suppose that we have a network $\mathcal{N} = (V, E, T)$, where V is the node set, E is the edge set, and T is a subset of V of "special" nodes called *terminals*. The edges are binary: edge e_i is *down* with probability $q_i = 1 - p_i$ and *up* with probability p_i. Network *UP* state is defined as presence of terminal connectivity of all nodes $v \in T$.

The first idea of LA is to associate with each edge e_i an exponentially distributed random *birth time* τ_i with parameter λ_i having the following property. For an arbitrary chosen time value t_0

$$P(\tau_i \le t_0) = 1 - e^{-\lambda_i t_0} = p_i, \;\; i = 1, 2, ..., n. \tag{2.3}$$

Let us assume that edge e_i is born at time τ_i and initially was in state *down*. At the instant τ_i of its birth it becomes *up* and stays in this state "forever". Then the probability that it will be "alive" at time t_0 equals $P(\tau_i \le t_0) = p_i$. For simplicity we put $t_0 = 1$ and note that from (2.3) follows

$$\lambda_i = -\ln q_i. \tag{2.4}$$

Now comes the following crucial observation: if we take a "snapshot" of the state of all edges at time instant $t_0 = 1$, we will see the network in the state which is stochastically equivalent to the static picture in which edge e_i is *up* or *down* with probability p_i or q_i, respectively. In particular, the snapshot will reveal the network in *UP* with probability identical to that obtained for the static edge "lottery".

The second idea of LA is to consider the edge birth times as a process developing in time. Let us consider an instructive example.

Example 2.5.1 Network with 5 nodes and 7 edges.
Figure 2.9 shows a network with 5 nodes and 7 edges. Network is *UP* if all its nodes are connected to each other. The birth process starts at $t = 0$ from state denoted as σ_0: no edges are born. Suppose that edge e_5 is born first. Note that the time ξ_1 of its birth $\xi_1 = \min\{\tau_1, \tau_2, ..., \tau_7\}$ and it is exponentially distributed with parameter $\Lambda_1 = \sum_{i=1}^{7} \lambda_i$:

$$\xi_1 \sim Exp(\Lambda_1).$$

The probability that the first birth will be of edge 5 equals λ_5/Λ_1.

The next edge born will be one of the remaining edges. Suppose that the next birth takes place at time instant $\xi_1 + \xi_2$. Due to the properties of exponential independent random variables, see e.g. [12], ξ_1 and ξ_2 are independent, and

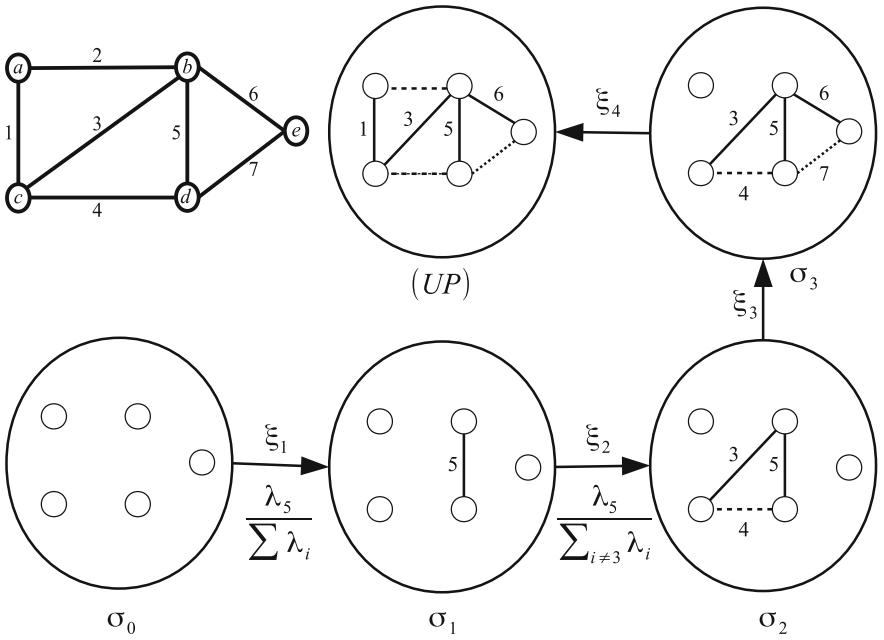

Fig. 2.9 Evolution trajectory

$$\xi_2 \sim Exp(\Lambda_2 = \sum_{i \neq 5} \lambda_i).$$

The probability that the second birth will be of edge 3 equals λ_3/Λ_2. The network enters now the state shown on Fig. 2.9 as σ_2. Continuing this process we will observe the third birth at the instant $\xi_1 + \xi_2 + \xi_3$ where

$$\xi_3 \sim Exp(\Lambda_3 = \sum_{i \neq 3,5} \lambda_i),$$

and ξ_3 is independent of ξ_1 and ξ_2. The third birth will be of edge 6 with probability λ_6/Λ_3. After the births of edges 5, 3 and 6, the network will be in state σ_3—see Fig. 2.9. Suppose that the next birth will be of edge 1. Suppose it happens at time instant $\xi_1 + \xi_2 + \xi_3 + \xi_4$. Obviously, all ξ_i-s are independent,

$$\xi_4 \sim Exp(\Lambda_4 = \sum_{i \neq 3,5,6} \lambda_i),$$

and fourth birth will be of edge 1 with probability λ_1/Λ_4. The birth of edge 1 signifies network entrance into the *UP* state. It is an absorbing state, and for all further births the system remains in *UP*.

The sequence of states $\omega = \{\sigma_0 \Rightarrow \sigma_1 \Rightarrow \sigma_2 \Rightarrow \sigma_3 \Rightarrow UP\}$ leading from the initial state σ_0 to the *UP* state is called *trajectory*. The probability that the trajectory will be as in our example is

$$\frac{\lambda_5}{\Lambda_1} \cdot \frac{\lambda_3}{\Lambda_2} \cdot \frac{\lambda_6}{\Lambda_3} \cdot \frac{\lambda_1}{\Lambda_4}.\#$$

Now let consider how the evolution process is used for computing network reliability. Denote the set of all trajectories ω by Ω. By conditioning over particular trajectory ω, we can represent the network *UP* probability in the following form:

$$P(\text{Network is } UP) = \sum_{\omega \in \Omega} P(\omega) P(\xi_1 + \xi_2 + \cdots + \xi_r \le t_0 = 1|\omega). \tag{2.5}$$

The probability $P(\xi_1 + \xi_2 + \cdots + \xi_r \le t_0|\omega)$ can be computed in a closed form using hypo-exponential distribution, see [10, 12] and [4], Appendix B.

Let $\Lambda_1 > \Lambda_2 > \Lambda_3 > \cdots > \Lambda_r$, and $\xi_i \sim Exp(\Lambda_i)$ be independent.
Then

$$P(\sum_{i=1}^{r} \xi_i \le t) = 1 - \sum_{i=1}^{r} e^{-\Lambda_i t} \prod_{j \ne i} \frac{\Lambda_j}{\Lambda_j - \Lambda_i}. \tag{2.6}$$

Since the quantity of interest in (2.5) is expressed as mathematical expectation, it can be estimated without bias as the sample average of computed probabilities $P(\xi_1 + \xi_2 + \cdots + \xi_r \le t_0 = 1|\omega_j)$ over a sample of trajectories $\{\omega_1, \omega_2, ..., \omega_M\}$:

$$\widehat{P}(UP) = \frac{\sum_{j=1}^{M} P(\xi_{i_1} + \xi_{i_2} + \cdots + \xi_{r_j} \le t_0 = 1)|\omega_j)}{M}. \tag{2.7}$$

In the case when the components of the network are edges, the above described evolution method can be complemented by so-called *closure* or *merging* operation. Let us explain what means closure by Example 2.5.1. Consider the state σ_2 on Fig. 2.9. Edges 3 and 5 are already born. These edges connect three nodes: b, c and d. On each stage of the birth process the edges born and their nodes create components, i.e. connected sets of nodes. So, nodes c, b and d belong to one component. Note that edge 4 which connects nodes c and d of this component can be merged with the already born edges without changing the component which already has been born! More formally, we can conclude that edge $4 = (c, d)$ is not *relevant* and can be excluded from the further evolution process. After adding this edge, the remaining edges are only 1, 2, 6 and 7. Suppose that the next edge born will be 6, see state σ_3. Now the component has nodes b, c, d, e, and edge $7 = (d, e)$ joins the nodes belonging to the already existing component. So, now edge 7 can be added to the

already born edges. After "merging" edge 7, we are left with only two non born edges, 1 and 2. The birth of any of them signifies the entrance into the *UP* state.

So, after each birth of an edge, we look for those edges, whose nodes belong to the already existing component. These edges are joined to this component and excluded from further considerations as irrelevant. Thus, after carrying out the closure operation, the trajectory leading from σ_0 to *UP* state represents a "thick" set, a "bundle" of trajectories. As the result, the dimension of the set of all trajectories Ω becomes smaller, and the estimate (2.7)—less variable.

Let us note that the estimate $P(DOWN) = 1 - P(UP)$ by (2.7) is free of so-called rare event phenomenon (unlimited increase of variance when $P(DOWN) \to 0$). In addition, the relative error of the estimator of $P(DOWN) = 1 - P(UP)$ is uniformly bounded with respect to the λ_i values. For additional details see [4], Chap. 9.

2.5.2 Extension of LA to Multistate Components

Originally, Lomonosov's algorithm described in [3] dealt with systems whose components subject to failure had only two states: *up* and *down*, or *exist-erased*. Now suppose that we have a network whose edges can be in three states: *up, down* and *mid*. In practice, edge $e = (a, b)$ in *mid* may correspond to a connection between a and b with reduced capacity or reduced speed of information exchange.

Suppose that we have a network whose components subject to failures are edges $e_1, e_2, ..., e_n$, and these edges are independent but *not* identical. Edge e_i has probabilities $p_2(e_i), p_1(e_i)$ and $p_0(e_i)$ to be in state *up, mid* or *down*, respectively. Our goal is to find network static *DOWN* probability.

Consider a two-stage edge birth process: on the first stage the edge is born without specifying its state, *mid* or *up*. Afterwards, on the second stage, the born edge is assigned its "type"—*mid* or *up*. Let the probability that edge is born be

$$P(e \text{ born}) = p_b(e) = p_2(e) + p_1(e). \tag{2.8}$$

On the second stage, after the edge e is born, we carry out a *random lottery*, in which we assign state *up* to e with probability

$$P(\text{edge } e \text{ is } up | e \text{ is born}) = \frac{p_2(e)}{p_1(e) + p_2(e)} \tag{2.9}$$

and state *mid* with probability

$$P(\text{edge } e \text{ is } mid | e \text{ is born}) = \frac{p_1(e)}{p_1(e) + p_2(e)}. \tag{2.10}$$

Now let us associate with edge e its *birth time* $\tau_e \sim Exp(\lambda_e)$, where λ_e is chosen in such a way that the probability that at t_0 edge e is born equals $p_b(e)$:

$$P(\tau_e \le t_0) = 1 - e^{-\lambda_e t_0} = p_b(e). \tag{2.11}$$

A "snapshot" at t_0 on all edges (born or not) will reveal a picture which is stochastically equivalent to static independent "lotteries" which determine the edge states according to the above probabilities $\mathbf{p}(e) = (p_2(e), p_1(e), p_0(e))$. Now we can implement the evolution process of LA and simulate, similarly to the above described procedure, the ω trajectories.

Here, however, arises a complication. The principal property of the evolution process for a binary system was the fact that each trajectory starting from the initial state with no edges born led eventually to system *UP* state. This was guaranteed by the property of monotone systems: if all components are born, then the system should be *UP*. But now, we are in a situation where component can be born as *mid* or as "*up*". And we have no guarantee that even a full trajectory with all edges born brings the system into *UP*. Suppose, for example, that network *UP* state means the following: (i) all nodes are connected with each other by *mid* or *up* edges; (ii) nodes a, b, c must be "strongly" connected, i.e. connected only by edges in *up*. Suppose it happens that a full trajectory has only *mid* edges, i.e all births produced *mid* state. Then the trajectory can not end in the *UP* state, and it ends, in fact, in *DOWN* state.

We will adopt the following zero-rule for the evolution process for which trajectories may end in *DOWN* state:

If the trajectory $\omega^* = \{\sigma_0 \Rightarrow \sigma_1 \Rightarrow \cdots \sigma_r \equiv DOWN\}$ ends in *DOWN* , put

$$P(\xi_1 + \cdots + \xi_r \le t_0 | \omega^*) := 0. \tag{2.12}$$

This rule can be justified by the following reasoning. Supply our original network with a set of "superedges" (or "supercomponents"), such that two properties will be guaranteed:

(i) these edges have no *mid* state, and if they are born they always are in *up*;
(ii) If *all* superedges are born, network is always *UP*.

For example, network is *UP* if nodes s and t are connected only by edges in *up* and all other nodes are connected to each other. Then it will be enough to have one superedge $e_0 = (s, t)$. Another example is a flow network which is in *UP* if $s - t$ flow exceeds γ. Then add to the network $s - t$ superedge and assign to it capacity that exceeds the maximal flow L.

Now modify the original problem by adding the superedges (supercomponents) to the network and giving them negligible small *up* probability ε.

Let us see what will be the consequences of that. Suppose $\varepsilon = 10^{-10} = p_2(e_0)$ is the probability that superedge e_0 is born. (Remind that the superedge after its birth is in state *up*.) Then the probability that it will be born before $t_0 = 1$ equals

$$P(\tau_{e_0} \le 1) = 10^{-10} = 1 - e^{-\lambda}.$$

Therefore $\lambda \simeq 10^{-10} = \varepsilon$. This means that τ_{e_0} has a huge mean value $1/\varepsilon$. Therefore the probabilities of type $P(\xi_1 + \cdots + \xi_r + \cdots + \tau_{e_0} \le 1 | \omega)$ will be very close to

zero. On the other hand, the presence of very small ε in the expressions of type $\lambda_j/(\lambda_1+\cdots+\lambda_k+\varepsilon)$ will have no influence on the evolution process and the choice of the next born edge. Therefore, with probabilities arbitrary close to 1, the modified birth process will be developing exactly as if the superedges do not exist. After all "regular" edges will be born and only superedges are left, the evolution trajectories will continue by including the superedges. This will guarantee that all trajectories end in *UP* state. But the contribution to the numerator of (2.7) coming from these trajectories will be negligible. All this justifies the above "zero-rule" (2.12).

Now we are ready to formulate in general terms the Monte Carlo algorithm for estimating reliability of a ternary network with statistically independent and *non-identical* components.

Modified LA

(I) Using the $p_2(e), p_1(e), p_0(e)$, assign to each component e its birth rate λ_e and the type *mid* or *up* in case component e is born, according to (2.8)–(2.10).

(II) Generate the evolution trajectory by choosing on each step a component e from the set of nonborn components with probability λ_e/Λ_e, where Λ_e is the sum of λ's for all nonborn components. If component e is born, remember its birth time $\tau_e \sim Exp(\lambda_e)$. If closure operation is used, apply it, and merge the closing components to the existing ones and cross them out from the list of non born components.

After each birth, check if the trajectory has reached the *UP* state. If the *UP* state is being reached, STOP, and go to (III); otherwise continue the birth process until all non born components are exhausted. If *UP* state is not being reached remember the corresponding trajectory and mark it as "zero" type.

(III) For each trajectory ω of non "zero" type compute $P(\xi_1+\xi_2+\cdots+\xi_r \le 1|\omega)$ by (2.5)–(2.6); put $P(\xi_1+\xi_2+\cdots+\xi_r \le 1|\omega)=0$ for zero type trajectory.

(IV) Put $P(DOWN) := P(DOWN) + P(\xi_1+\xi_2+\cdots+\xi_r \le 1|\omega)$

(V) Repeat Steps (II–IV) M times. Put $P(DOWN) := P(DOWN)/M$.

2.5.3 Flow Network

Let us consider the dodecahedron flow network studied earlier in Sect. 2.2. In Sect. 2.2 it was assumed that all edges have identical probabilities to be in *up, mid* or *down.* Now we introduce different probabilities for the edges to fail, according to the following rule. Nodes are numbered $j = 1, 2, ..., 20$. Edge $e = (i, j)$ is declared even if $i + j$ is even, and odd—otherwise. Assume that even edges have probabilities $p_2(a), p_1(a), p_0(a)$, and odd—$p_2(b), p_1(b), p_0(b)$. Edge capacities remain the same: 0, 3, and 6 for *down*, *mid* and *up* edge, respectively. Similar to the example in Sect. 2.2. there are two threshold values for the $s-t$ flow $L_1 = 10$ and $L_2 = 4.5$.

To find network *DOWN* probability, we applied the modified LA. The closure operation is not applicable here since the *UP* criterion is not formulated in terms of node connectivity. The number of simulated trajectories was M =1,000,000. All

Table 2.9 $P(DOWN)$ for flow network with two types of edges

i	$p_2(a); p_2(b)$	$p_1(a); p_1(b)$	$p_0(a); p_0(b)$	L	$P(DOWN)$	RE	CPU sec
1	0.6; 0.4	0.3; 0.4	0.1; 0.2	10	**0.757**	0.1 %	123
2	0.6; 0.4	0.3; 0.4	0.1; 0.2	4.5	**0.119**	0.2 %	103

simulation results obtained by the evolution method were checked by CMC with 10^6 replications, and numerically the results coincide with the results obtained by the LA with high accuracy (Table 2.9).

Below is a short summary of simulation results using LA:

Line $i = 1$ presents the *DOWN* probability for $L_1 = 10$. The first figure for p_2, p_1, p_2 is for even edges, the second— for odd edges. Line 2 shows similar results for reduced maximal flow $L_2 = 4.5$. Network *DOWN* changes considerably with the reduction of the maximal flow. The RE's are rather small and the CPU time is quite satisfactory. Let us note that the zero-type trajectories were observed in about 8 % of replications for $L_1 = 10$ and were not observed for $L_1 = 4.5$.

2.5.4 Grid Network

Here we consider a 10×10 grid network shown on Fig. 2.5. The network has 100 nodes. Node failure (*down* state) means erasing all edges adjacent to this node. If node v is in *mid* state, and this node is even (row number + column number is even), then the horizontal edges adjacent to v are erased. If node v is in *mid* and its row number + column number is odd, then the vertical edges adjacent to v are erased.

Node state probabilities are

$$p_{up}(a) = 0.6, p_{mid}(a) = 0.3, p_{down}(a) = 0.1 - \text{ if node } a \text{ is even};$$

and node state probabilities are

$$p_{up}(b) = 0.65, p_{mid}(b) = 0.15, p_{down}(b) = 0.2 - \text{ if node } b \text{ is odd.}$$

The *DOWN* state for the network was defined in two ways:

(i) The maximal component has less than $L_1 = 81$ node;
(ii) The maximal component has less than $L_2 = 51$ node.

The LA can be easily applied to the case when the components subject to failure are the nodes, see [9]. Formally, a closure operation can be applied for the nodes too, but its algorithmic implementation is quite complicated. We applied the modified LA without closure.

Table 2.10 $P(DOWN)$ for grid network with two types of nodes

i	$p_2(a); p_2(b)$	$p_1(a); p_1(b)$	$p_0(a); p_0(b)$	L	$P(DOWN)$	RE	CPU sec
1	0.6; 0.65	0.3; 0.15	0.1; 0.2	80	**0.5349**	0.05 %	256
2	0.6; 0.65	0.3; 0.15	0.1; 0.2	50	**0.0710**	0.2 %	197

The simulation results using the modified LA are presented in Table 2.10.

The number of trajectories in the evolution algorithm was M =1,000,000. Let us note that the % of evolution trajectories ending in *DOWN* state was equal 0.1 % for L_1 and 1 % for L_2.

References

1. Barlow RE, Proschan F (1975) Statistical theory of reliability and life testing. Holt Rinehart and Winston Inc, New York
2. Birnbaum ZW (1969) On the importance of different components in multicomponent system. In: Krishnaiah PR (ed) Multivarite analysis-II. Academic Press, New York. pp. 581–592
3. Elperin T, Gertsbakh IB, Lomonsov MV (1991) Estimation of network reliability using graph evolution models. IEEE Trans Reliab 40(5):572–581
4. Gertsbakh I, Shpungin Y (2009) Models of network reliability: analysis combinatorics and Monte Carlo. CRC Press, Boca Raton
5. Gertsbakh I, Shpungin Y (2011) Network reliability and resilience. Springer briefs in electrical and computer engineering. Springer, Heidelberg
6. Gertsbakh I, Shpungin Y (2012) Combinatorial approach to computing importance indices of coherent systems. Probab Eng Inf Sci 26:117–128
7. Gertsbakh I, Shpungin Y (2012) Stochastic models of network survivability. Qual Technol Quant Manag 9(1):45–58
8. Gertsbakh I, Rubinstein R, Shpungin Y, Vaisman R (2014) Permutational methods for performance analysis of stochastic flow networks. Probab Eng Inf Sci 28:21–38
9. Gertsbakh I, Shpungin Y, Vaisman R (2014) Network reliability Monte Carlo with nodes subject to failure. Int J Performability Eng 10(2):161–170
10. Kroese D, Taimre T, Botev IZ (2011) Handbook of Monte Carlo methods. Wiley, New York
11. Lewis TG (2009) Network science: theory and applications. Wiley, Hoboken
12. Ross S (2007) Introduction to probability models, 9th edn. Academic Press, New York

http://www.springer.com/978-3-319-06439-0

Ternary Networks
Reliability and Monte Carlo
Gertsbakh, I.; Shpungin, Y.; Vaisman, R.
2014, XIV, 62 p. 18 illus., 7 illus. in color., Softcover
ISBN: 978-3-319-06439-0