Andrey GRISHKEVICH

Politechnika Częstochowska, Instytut Informatyki

doi:10.15199/48.2015.12.26

Rozproszone modele symulacyjne pozwalające oszacować wskaźniki niezawodności strukturalnej systemów elektroenergetycznych

Streszczenie. *Oszacowano wypadkowe wskaźniki niezawodności strukturalnej systemów elektroenergetycznych na podstawie modeli symulacyjnych. Otrzymano implementację algorytmiczną rozproszonego modelu symulacyjnego z wykorzystaniem biblioteki MPI. Podano wyniki testowania modelu symulacyjnego rozproszonego w klasterze.*

Abstract*. Estimates for electric power system failure states contribution to the resulting system reliability indices are obtained on the basis of simulation model. An algorithmic implementation of a distributed simulation model using the MPI library is proposed. The results of testing the distributed simulation model on a cluster are presented.(****Distributed simulation models to estimate structural reliability indices of electric power systems****).*

Słowa kluczowe: symulacja rozproszona, biblioteka MPI, klaster obliczeniowy, niezawodność strukturalna, model Markowa.
Keywords: distributed simulation, library MPI, computing cluster, structural reliability, Markow model.

Wstęp

Obliczanie niezawodności [1] układu zasilania oparte jest na sumowaniu wkładów do wypadkowych wskaźników niezawodności od różnych stanów uszkodzenia systemu. Wkład stanów (dla systemów wysokiej niezawodności są zazwyczaj rozpatrywane jednoczesne uszkodzenia 1 i 2 elementów) jest obliczany na podstawie modeli funkcjonowania 1 i 2 elementów ze względu na niezawodność.

Metody analityczne obliczania niezawodności stosowane są przy użyciu modeli Markowa funkcjonowania elementów. U podstaw tych modeli leży założenie o wykładniczym charakterze rozkładów zmiennych losowych. W rzeczywistych układach rozkład niektórych zmiennych losowych, takich jak czas remontu zapobiegawczego, znacznie różni się od wykładniczego. Istotny wpływ na rozkład zmiennej losowej ma dyscyplina obsługi (sposoby organizowania i prowadzenia remontów), która zmienia się bardzo szybko. Zamiana nieznanego rozkładu zmiennej losowej na rozkład wykładniczy prowadzi do znacznych błędów w wynikach obliczeń.

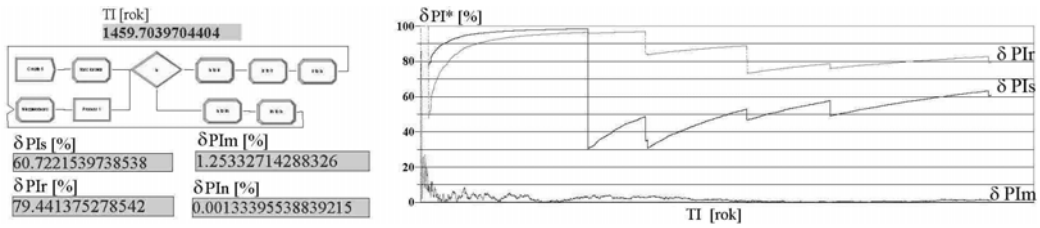
Niepewność danych źródłowych doprowadziła do opracowania metod interwałowego obliczania wypadkowych wskaźników niezawodności [2,3]. Jednak uwzględnienie nieznanego rozkładu zmiennej losowej poprzez przedłużenie przedziału danych wejściowych jest bardzo niewiarygodne.

Wyjątkowo wszechstronne są metody modelowania matematycznego [4,5]. Metody te pozwalają rozważać (symulować) wszelkie rozkłady zmiennych losowych.

Budowanie modelu symulacyjnego można przeprowadzić w środowiskach symulacyjnych, na przykład, w programie komputerowym Arena [6]. Model funkcjonowania jednego elementu i niektóre wyniki symulacji przedstawiono na rys. 1. Należy zauważyć, że: 1) chociaż graficzny interfejs programu daje znaczne możliwości wizualizacji procesu i metod prezentacji wyników symulacji, to jednak bardzo spowalnia sam proces modelowania; 2) pakiet nie obejmuje środków efektywnego zrównoleglenia symulacji w klasterze. Osiągnięcie zadowalającej dokładności dla modelu dwóch elementów wymaga bardzo dużo czasu. Opis modelu w programu Arena, wykorzystując własny sposób definiowania poleceń, nie daje korzyści w porównywaniu z opisem modelu w języku wyższego rzędu, na przykład w C++. Natomiast program w kodzie źródłowym reprezentuje szerokie spektrum możliwości jego modyfikacji i użycia dodatkowych narzędzi (bibliotek oprogramowania).

Algorytmy symulacyjne są dobrze rozwinięte, bardzo proste i jasne. Jednak uzyskanie zadowalającej dokładności wymaga dużej liczby operacji obliczeniowych i nie jest możliwe, w rozsądnym czasie, w tradycyjnych systemach obliczeniowych z architekturą sekwencyjną.

Nowoczesne oprogramowanie pozwala w standardowy sposób łączyć zasoby informatyczne wielu komputerów (istniejącej sieci), zmieniając je w system z równoległą organizacją obliczeń – klaster obliczeniowy, i zmieniać wielkość klastera w zależności od potrzeb mocy obliczeniowej. Tworzenie prostego klastera nie wymaga zakupu dodatkowego sprzętu, jest to wykonywane łatwo i szybko. Jeśli moc obliczeniowa skonfigurowanego klastera jest niewystarczająca, można użyć mocniejszego klastera lub superkomputera. Obecnie możliwy jest dostęp do tej technologii za pośrednictwem Internetu. Transformacja kodu algorytmu symulacji w języku wysokiego rzędu, aby go uruchomić w klasterze, wymaga połączenia z istniejącym projektem programu specjalnej biblioteki i korzystania z określonych poleceń tej biblioteki. Kod programu jest niezależny od parametrów klastera, w którym dokonuje się obliczeń. W ten sposób możemy osiągnąć wydajność obliczeniową zapewniającą praktyczne zastosowanie technik modelowania.

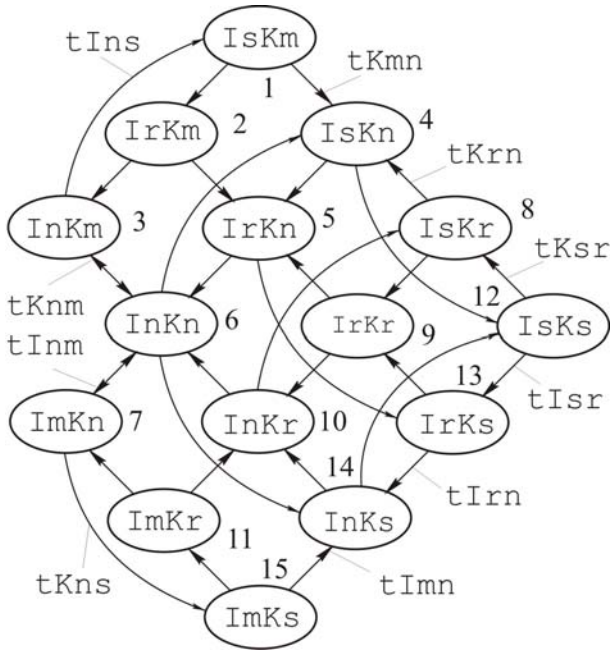


Rys.1. Model funkcjonowania jednego elementu w programie komputerowym Arena

Model symulacyjny funkcjonowania dwóch elementów

Każdy element E systemu elektroenergetycznego (w zastosowaniu do elektroenergetyki to transformator, wyłącznik itd.) może znajdować się w jednym z czterech stanów. Przyjmujemy założenie, że En – stan normalnej pracy elementu, Es – stan między uszkodzeniem elementu i zakończeniem przełączeń operacyjnych, Er – stan awaryjnego remontu elementu, Em – stan remontu zapobiegawczego (zamierzonego odłączenia) elementu; tExy – losowy czas przejścia elementu E ze stanu Ex do stanu Ey. Stan systemu zależy od stanu każdego elementu systemu.

Przestrzeń stanów i diagram przejść pomiędzy stanami przy modelowaniu niezawodności funkcjonowania dwóch elementów I, K są przedstawione na rysunku 2 [1]. Dla wygody stany ponumerowano, czyli IsKn=S4. W sumie dla modelu dwóch elementów rozważa się 15 stanów.



Rys.2. Diagram przejść między stanami układu dwóch elementów

Proces modelowania polega na generowaniu sekwencji stanów systemu, począwszy od stanu początkowego InKn. Dla każdego stanu generuje się zmienną losową opisującą wyjście z tego stanu, i o przypadkowy czas pobytu w tym stanie koryguje się łączny czas przebywania w tym stanie TSi_sum1, i=1, 2, ..., 15.

Tak więc, dla stanu S4=IsKn możliwe są przejścia do stanu S5=IrKn z intensywnością 1/tIsr i do stanu S12=IsKs z intensywnością 1/tKns. Fragment kodu programu symulacyjnego procesów przejścia między stanami dla modelu dwóch elementów w języku programowania C++ w stosunku do stanu IsKn przedstawiono poniżej

```
...
SIsKn:
if (tIsr - tIsr >= 0 && tKns - tIsr >= 0)
// tIsr <= tKns; IsKn -> IrKn
{tKns = tKns - tIsr;
TIsKn_sum1 = TIsKn_sum1 + tIsr;
uzyskać nową wartość
zmiennej losowej tIsr;
goto SIrKn;}

if (tIsr - tKns >= 0 && tKns - tKns >= 0)
// tIsr >= tKns; IsKn -> IsKs
{tIsr = tIsr - tKns;
TIsKn_sum1 = TIsKn_sum1 + tKns;
uzyskać nową wartość
zmiennej losowej tIsr;
goto SIsKs;}
...
```

Całkowity czas symulacji

$$TIK = \sum_i TSi \tag{1}$$

jest opisany jako łączny czas pobytu w stanach Si, i=1, 2, ..., 15. Podczas sekwencyjnej organizacji procesu symulacji TSi=TSi_sum1, i=1, 2, ..., 15.

Spełnienie niektórych kryteriów (osiągnięcie przez całkowity czas symulacji wartości TIK0, przekroczenie zadanego czasu pracy procesora, ...) w stanie InKn inicjuje zakończenie procesu symulacji i rozpoczyna obliczanie wskaźników niezawodności. W związku z tym wartość

$$PSi = TSi/TIK \tag{2}$$

jest oszacowaniem prawdopodobieństwa pobytu systemu w stanie Si (wkładu stanu Si w wypadkowe wskaźniki niezawodności), i=1, 2, ..., 15, na podstawie wyników symulacji.

Algorytmiczna implementacja rozproszonego modelu symulacyjnego na podstawie języka programowania wyższego rzędu w klasterze komputerowym

Czas symulacji komputerowej w węźle obliczeniowym z jednym procesorem przy zadowalającej dokładności (wystarczająco dużej liczbie prób) jest zbyt duży. Realizacja takiego modelu w języku C++ z wykorzystaniem biblioteki programowania MPI umożliwia przeprowadzenie symulacji rozproszonych stanów systemu przy użyciu mocy obliczeniowej klastera. MPI jest specyfikacją biblioteki funkcji opartych na modelu wymiany komunikatów dla potrzeb programowania równoległego. Realizacja rozproszonych obliczeń wkładu stanów uszkodzenia systemu w wypadkowe wskaźniki niezawodności polega na zorganizowaniu skutecznej wymiany informacji (pośrednie wyniki obliczeń) między rozproszonymi procesami w trakcie symulacji.

Wybrano realizację DeinoMPI 2.0 [7], która jest bardzo prostym narzędziem do tworzenia klasterów obliczeniowych.

Na podstawie programu komputerowego opracowanego z wykorzystaniem biblioteki MPI, uruchamia się pewna liczba (numprocs) równoległych procesów w węzłach obliczeniowych sieci (klastera).

Każdy proces wylicza całkowity czas TSi_sum1 przebywania w stanie Si, i=1, 2, ... , 15, na podstawie 50 000 000 trafień do stanu InKn. Zgodnie z poleceniem

```
MPI_Reduce(&TSi_sum1, &TSi_sum2, 1,
MPI_DOUBLE,MPI_SUM, 0, MPI_COMM_WORLD);
```

biblioteki MPI wyniki wysyłane są do procesu 0. Proces 0 oblicza sumaryczny czas TSi_sum2 pobytu w stanie Si dla wszystkich numprocs procesów. Następnie proces 0 modyfikuje czas pobytu w stanie Si

```
TSi = TSi + TSi_sum2;
```

uzyskany na podstawie metody modelowania. Wynik ten reprezentuje jeden elementarny krok symulacji.

Wspomniana powyżej sekwencja działań (podstawowe kroki symulacji) powtarza się. Oszacowania prawdopodobieństwa stanów na podstawie wyników testów liczone są zgodnie z wyrażeniami (1), (2) i przechowywane. Taka organizacja procesu obliczeń pozwala kontrolować ten proces i jednocześnie zmniejszyć liczbę zarówno "wolnych" operacji przesyłania przez sieć jak i wizualizacji wyników.

Modelowanie pożądanego rozkładu zmiennej losowej w języku programowania C++ może być oparte na równomiernym rozkładzie, który opisuje się funkcją standardową `rand()`. Początkowy parametr rozkładu równomiernego ustala się za pomocą polecenia

```
srand((unsigned)time(NULL)*(myid + 1));
```

gdzie myid – numer procesu.

W celu adekwatnej symulacji zmiennych losowych na długich przedziałach czasu zaleca się korzystać z narzędzi (generator liczb pseudolosowych) biblioteki BOOST C++ [8].

Testowanie modelu symulacyjnego

Testowanie proponowanego modelu symulacyjnego przeprowadzono przy założeniu wykładniczości rozkładu przejścia między stanami systemu jak na rysunku 2. Pozwoliło to porównać wyniki symulacji z dokładnym rozwiązaniem dla stałych prawdopodobieństw stanów.

Testowe dane numeryczne dotyczące wskaźników niezawodności elementów `I, K` systemu elektroenergetycznego są wzięte z [1]: `TIns0`=1/0,01 [rok] – średni czas przejścia elementu `I` ze stanu `In` do stanu `Is`; `TInm0`=1/2,2 [rok]; `TIsr0`=2/8760 [rok]; `TIrn0`=11,39/8760 [rok]; `TImn0`=7,96/8760 [rok]; `TKns0`=1/0,04 [rok]; `TKnm0`=1 [rok]; `TKsr0`=2/8760 [rok]; `TKrn0`=2,19/8760 [rok]; `TKmn0`=7/8760 [rok]. Prezentowane wartości liczbowe były traktowane jako parametry rozkładu wykładniczego odpowiedniej zmiennej losowej.

Ustalone prawdopodobieństwa stanów dla modelu funkcjonowania dwóch elementów przy wykładniczych rozkładach intensywności przejść między stanami spełniają układ równań liniowych Markowa 15 rzędu. W celu wyeliminowania wpływu błędów numerycznych metod otrzymano rozwiązanie analityczne wspomnianego układu równań liniowych.

Rozwiązania analityczne otrzymano za pomocą programu komputerowego MatSym [9] przeznaczonego do analitycznego rozwiązywania rzadkich układów równań liniowych o szczególnej postaci (odpowiadających rzeczywistym obwodom elektrycznym). Rozwiązanie układu składa się z 55000 elementów i wskutek jego złożoności nie zostało tu przytoczone. Dla badanych źródłowych danych testowych liczbowe rozwiązania przedstawione są w tabeli 1, kolumna 2.

Tabela 1. Prawdopodobieństwa `PSi` stanów `Si` modelu dwuelementowego

`Si`	Rozwiązania analityczne	Modelowanie `TIK` =5,43*10^{11} [rok]	δ %
1	2	3	4
`IsKm`	1,4149609777187515e-09	1,4141744512117363e-09	0,06
`IrKm`	3,0672876224447003e-09	3,0667900805793985e-09	0,02
`InKm`	7,9682516773818828e-04	7,9682670606513349e-04	0,00
`IsKn`	2,277060424673709e-06	2,277011693507628e-06	0,00
`IrKn`	1,2972850007795631e-05	1,2972532861437888e-05	0,00
`InKn`	0,99717539055489546	0,99717539184612969	0,00
`ImKn`	1,9934130756254858e-03	1,9934105581828543e-03	0,00
`IsKr`	2,2778585976710348e-11	2,2927079997908464e-11	0,65
`IrKr`	1,2972334741148401e-10	1,3026858646755552e-10	0,42
`InKr`	9,9791941325101828e-06	9,9793866946375404e-06	0,00
`ImKr`	1,2493900917220783e-08	1,2497453355914473e-08	0,03
`IsKs`	2,0797399351670186e-11	2,0840403658753217e-11	0,21
`IrKs`	1,1846868756106299e-10	1,1861967840341811e-10	0,13
`InKs`	9,1102801331409688e-06	9,1101365696134461e-06	0,00
`ImKs`	1,4549126260272936e-08	1,4550730254242144e-08	0,01

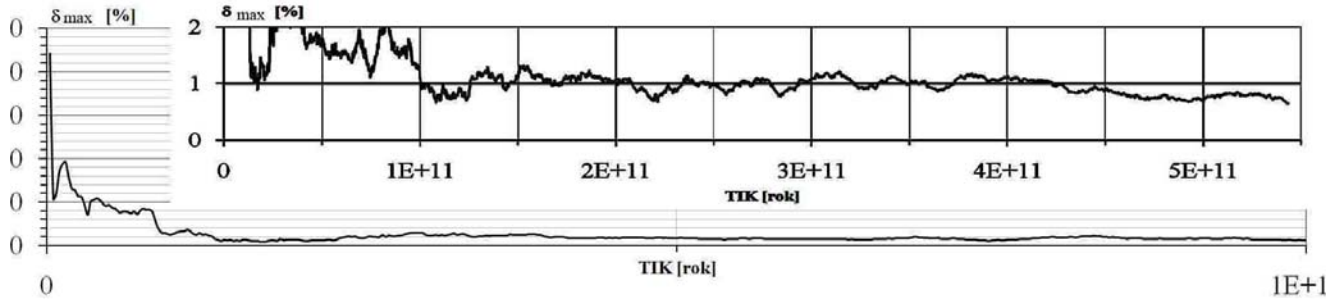
Rozkład wykładniczy z parametrem `a` symulowany jest za pomocą rozkładu równomiernego (funkcja `rand`) poprzez funkcję

```
double EXPO(double a) {
int x;
b: x = rand();
if (x==0) goto b;
return
-a*log( (double)x / (double)RAND_MAX );}
```

Ustalone prawdopodobieństwa stanów uzyskane metodą symulacji zostały przedstawione w tabeli 1, kolumna 3. W kolumnie 4 podano błąd względny δ. Rys. 2 jest wykresem maksymalnego błędu względnego δ_{max} na końcu przedziału symulacji.

Obliczenia przeprowadzono dla 8 procesów (`numprocs=8`) startowych w 2 węzłach obliczeniowych (Intel (R) Core (TM) CPU @ 1.70GHz i i5-3317U @ 2.40GHz i3-3110M, węzeł obejmuje dwa rzeczywiste i dwa wirtualne rdzenie), które zapewniają prawie 4-krotne przyspieszenie obliczeń.

Jeden krok modelowania (50 milionów trafień w stan `InKn`) wykonuje się w ciągu 18,3 [s] na klasterze o powyższej konfiguracji. Symulacja czasu `TIK`=5,43*10^{11} lat wymagała 14:35:44 godzin pracy klastera obliczeniowego.



Rys.2. Maksymalny błąd względny stanów na końcu przedziału symulacji

Zwiększenie zakresu modelowania powoduje, że błąd względny ma tendencję do zmniejszania się. Największe błędy względne na końcu przedziału symulacji obserwuje się w najmniej prawdopodobnych stanach (IsKr, IsKs, IrKs, IrKr), a dla stanu IsKr błąd jest maksymalny i wynosi 0,65%. Z wykresu na rys. 2 widać, że przy interwale modelowania większym niż 10^{11} ($4{,}5*10^{11}$) lat maksymalny błąd względny jest mniejszy niż 2% (1%). Taka dokładność jest osiągana odpowiednio po 2,5 i 12 godzinach pracy klastera obliczeniowego.

Wnioski

Proponowana jest metoda obliczania niezawodności strukturalnej systemów złożonych, w której na podstawie modeli symulacyjnych funkcjonowania elementów systemu oblicza się wkład stanów uszkodzenia w wypadkowe wskaźniki niezawodności. Taka metoda pozwala uwzględniać inne rozkłady zmiennych losowych niż rozkład wykładniczy, w szczególności, uwzględnić różne sposoby remontu awaryjnego i zapobiegawczego elementów systemu. Wymieniona właściwość pozwala zaadaptować proponowaną metodę obliczania niezawodności do szczególnych cech danego systemu.

W pracy przedstawiono wyniki testów rozproszonych modeli symulacyjnych, które pozwalają nam oszacować wymagane zasoby obliczeniowe (parametry klastera obliczeniowego) niezbędne do przeprowadzania symulacji niezawodności systemu elektroenergetycznego z wystarczającą dokładnością w rozsądnym czasie.

Klastery komputerowe mają wystarczającą moc obliczeniową, zapewniającą dopuszczalną dokładność symulacji w rozsądnym czasie. Szeroka dostępność i niski koszt klasterów umożliwiają praktyczne wykorzystanie metod rozproszonej symulacji do obliczania niezawodności strukturalnej złożonych systemów energetycznych.

LITERATURA

[1] Grishkevich A.A., Hudym V.I., Kruczinin A.M., Sawicki A., Zagadnienia energetyczne wybranych współczesnych urządzeń i systemów elektrostalowniczych, Wydawnictwo Politechniki Częstochowskiej, Częstochowa (2010).

[2] Grishkevich A., Burmutaev A., Modelowanie statystyczne oszacowań interwałowych wskaźników niezawodności strukturalnej układów elektrycznych, Przegląd Elektrotechniczny (Electrical Review), R88(2012), nr. 8, 77-79.

[3] Grishkevich A., Grishkevich M., Interwałowe oszacowania wskaźników niezawodności strukturalnej systemów elektroenergetycznych, Przegląd Elektrotechniczny (Electrical Review), R90(2014), nr. 6, 249-252.

[4] Ge H., Asgarpoor S., Parallel Monte Carlo simulation for reliability and cost evaluation of equipment and systems, Electric Power Systems Research, 81 (2011), 347–356.

[5] Prauzner, T. Effectiveness of magnetic detectors in alarm systems, Przegląd Elektrotechniczny (Electrical Review), R90(2014), nr 12, 269-272.

[6] Arena Simulation Software. (www.arenasimulation.com)

[7] DeinoMPI. (http://mpi.deino.net/)

[8] Библиотека BOOST C++: генераторы псевдослучайных чисел. (http://www.solarix.ru/for_developers/cpp/boost/random/boost-random-generators.shtml)

[9] Filaretov V.V., Shein D.V. Symbolic circuit-matrix processor and evaluator (MatSym). (http://astrometric.sai.msu.ru/~symbol/index.html)

***Autor**: prof. dr hab. inż. Andrey Grishkevich, Politechnika Częstochowska, Instytut Informatyki, al. Armii Krajowej 17, 42-200 Częstochowa, E-mail:* a.grischkevich@el.pcz.czest.pl, *grishkev_ramb@rambler.ru*