

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Качество и метрология программного обеспечения»

Тема: «Построение операционной графовой модели программы (ОГМП)
и расчет характеристик эффективности ее выполнения методом
эквивалентных преобразований»

Студент гр. 8304

Мухин А. М.

Преподаватель

Кирияничиков В. А.

Санкт-Петербург

2022

Цель работы.

Построение операционной графовой модели программы (ОГМП) и расчет характеристик эффективности ее выполнения методом эквивалентных преобразований.

Задание.

Для задания из лабораторных работ 1-3 разработать операционную модель управляющего графа программы на основе схемы алгоритма. При выполнении работы рекомендуется для упрощения обработки графа исключить диалог при выполнении операций ввода-вывода данных, а также привести программу к структурированному виду.

Выбрать вариант графа с нагруженными дугами, каждая из которых должна представлять фрагмент программы, соответствующий линейному участку или ветвлению. При расчете вероятностей ветвлений, зависящих от распределения данных, принять равномерное распределение обрабатываемых данных в ограниченном диапазоне (например, $[0,100]$ - для положительных чисел или $[-100,100]$ - для произвольных чисел). В случае ветвлений, вызванных проверкой выхода из цикла, вероятности рассчитываются исходя априорных сведений о числе повторений цикла. Сложные случаи оценки вероятностей ветвлений согласовать с преподавателем.

В качестве параметров, характеризующих потребление ресурсов, использовать времена выполнения команд соответствующих участков программы. С помощью монитора Sampler выполнить оценку времен выполнения каждого линейного участка в графе программы.

Полученную ОГМП, представить в виде графа с нагруженными дугами, у которого в качестве параметров, характеризующих потребление ресурсов на дуге ij , использовать тройку $\{P_{ij}, M_{ij}, D_{ij}\}$, где:

P_{ij} - вероятность выполнения процесса для дуги ij ;

M_{ij} - мат. ожидание потребления ресурса процессом для дуги ij ;

D_{ij} - дисперсия потребления ресурса процессом для дуги ij .

В качестве потребляемого ресурса в данной работе рассматривается время процессора, а оценками мат. ожиданий времен для дуг исходного графа следует принять времена выполнения операторов (команд), соответствующих этим дугам участков программы. Дисперсиям исходных дуг следует присвоить нулевые значения.

Получить описание полученной ОГМП на входном языке пакета CSA III в виде поглощающей марковской цепи (ПМЦ) – (англ.) AMC (absorbing Markov chain) и/или эргодической марковской цепи (ЭМЦ) - EMC (ergodic Markov chain).

С помощью предоставляемого пакетом CSA III меню действий выполнить расчет среднего времени и дисперсии времени выполнения как для всей программы, так и для ее фрагментов, согласованных с преподавателем. Сравнить полученные результаты с результатами измерений, полученными в работе 3.

Ход работы.

1. Текст программы (исходный).

Исходный код программы представлен в приложении А.

2. Профилирование.

Код программы для профилирования, разделенной на функциональные участки, представлен в приложении Б.

3. Граф управления программы.

Граф управления был построен на основе разбиения программы на функциональные участки по коду программы из лабораторной работы №3. Граф представлен на рисунке 1.

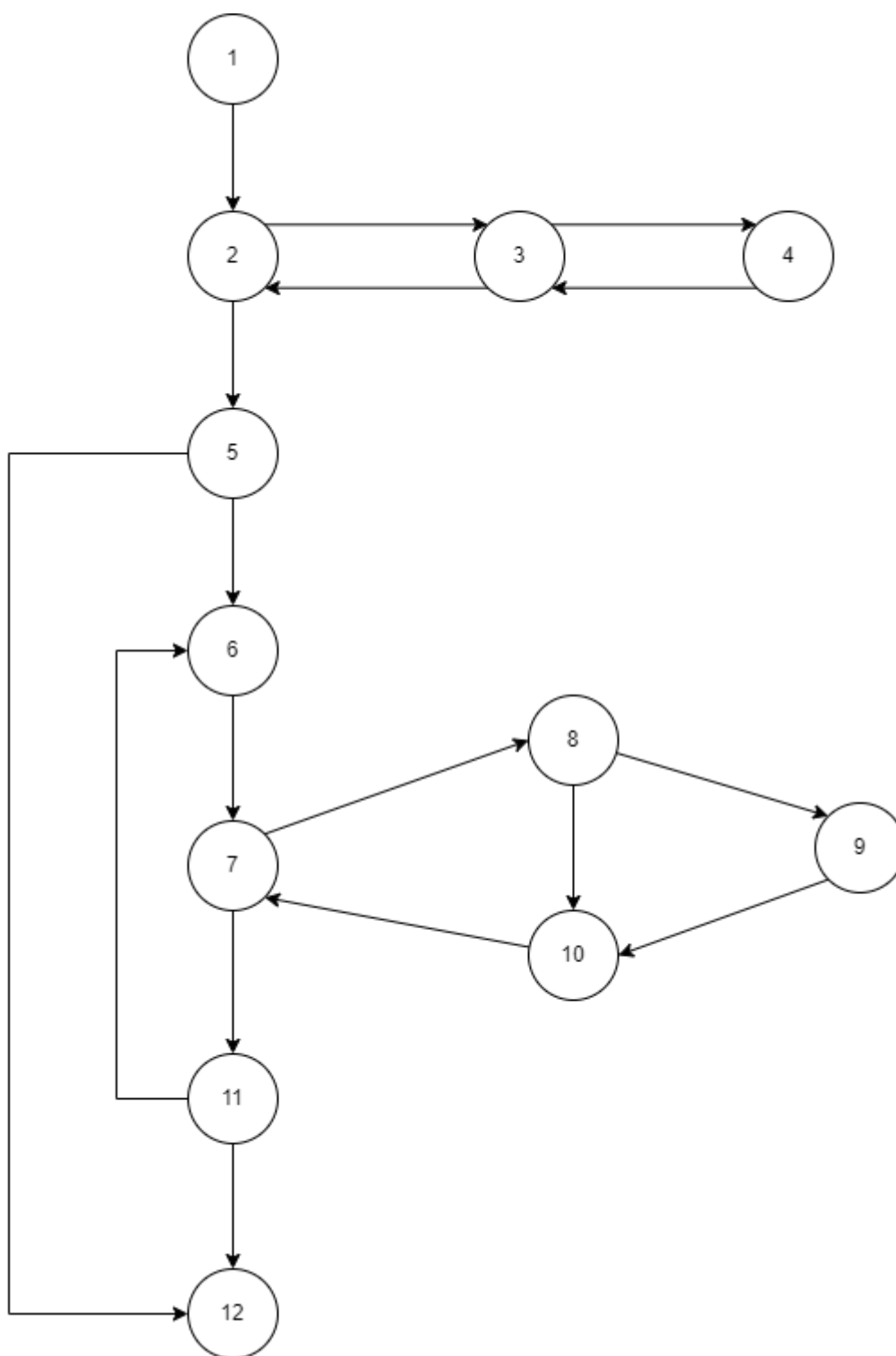


Рисунок 1 - Граф управления

4. Результаты профилирования.

Результаты профилирования из лабораторной работы №3 представлены на рисунке 2.

| исх | прием | общее время | кол-во проходов | среднее время |
|-----|-------|-------------|-----------------|---------------|
| 95 | 54 | 26.678 | 1 | 26.678 |
| 54 | 58 | 12.920 | 1 | 12.920 |
| 58 | 59 | 5.492 | 1 | 5.492 |
| 59 | 60 | 11.291 | 3 | 3.764 |
| 59 | 68 | 6.143 | 1 | 6.143 |
| 60 | 61 | 1.082 | 3 | 0.361 |
| 61 | 62 | 48.202 | 9 | 5.356 |
| 61 | 66 | 21.433 | 3 | 7.144 |
| 62 | 64 | 25.298 | 9 | 2.811 |
| 64 | 61 | 63.935 | 9 | 7.104 |
| 66 | 59 | 5.431 | 3 | 1.810 |
| 68 | 71 | 34.049 | 1 | 34.049 |
| 71 | 77 | 9.672 | 1 | 9.672 |
| 77 | 33 | 14.071 | 1 | 14.071 |
| 33 | 35 | 4.889 | 3 | 1.630 |
| 35 | 36 | 15.964 | 3 | 5.321 |
| 36 | 37 | 71.668 | 9 | 7.963 |
| 36 | 47 | 26.473 | 3 | 8.824 |
| 37 | 39 | 56.011 | 9 | 6.223 |
| 39 | 45 | 10.579 | 3 | 3.526 |
| 39 | 41 | 15.638 | 6 | 2.606 |
| 45 | 36 | 12.152 | 9 | 1.350 |
| 47 | 49 | 69.021 | 3 | 23.007 |
| 49 | 33 | 36.090 | 2 | 18.045 |
| 49 | 81 | 20.337 | 1 | 20.337 |
| 41 | 43 | 50.822 | 6 | 8.470 |
| 43 | 45 | 21.550 | 6 | 3.592 |
| 81 | 83 | 1.874 | 1 | 1.874 |
| 83 | 97 | 20.419 | 1 | 20.419 |

Рисунок 2 – Результаты профилирования

5. Расчет вероятностей и затрат ресурсов для дуг управляющего графа.

Результаты расчета представлены в таблице 1.

Таблица 1 – Расчет вероятностей и затрат ресурсов для дуг.

| | Номера строк | Количество проходов | Вероятность |
|---------------------|---------------------|---------------------|-------------|
| $L_{1-2} = 45,09$ | 94-54; 54-58; 58-59 | 1 | 1 |
| $L_{2-3} = 4,125$ | 59-60; 60-61 | 3 | 0,75 |
| $L_{3-2} = 8,954$ | 61-66; 66-59 | 3 | 0,25 |
| $L_{3-4} = 8,167$ | 61-62; 62-64 | 9 | 0,75 |
| $L_{4-3} = 7,104$ | 64-61 | 9 | 1 |
| $L_{2-5} = 40,192$ | 59-68; 68-71 | 1 | 0,25 |
| $L_{5-6} = 23,743$ | 71-77; 77-33 | 1 | 1 |
| $L_{5-12} = 0$ | 71-73; 73-97 | 0 | 0 |
| $L_{6-7} = 6,951$ | 33-35; 35-36 | 3 | 1 |
| $L_{7-8} = 14,186$ | 36-37; 37-39 | 9 | 0,75 |
| $L_{7-11} = 32,161$ | 36-47; 47-49 | 3 | 0,25 |
| $L_{8-9} = 11,076$ | 39-41; 41-43 | 6 | 0,67 |

Таблица 1 (продолжение) – Расчет вероятностей и затрат ресурсов для дуг.

| | Номера строк | Количество проходов | Вероятность |
|---------------------|---------------------|---------------------|-------------|
| $L_{8-10} = 3,526$ | 39-45 | 3 | 0,33 |
| $L_{9-10} = 3,592$ | 43-45 | 6 | 1 |
| $L_{10-7} = 1,35$ | 45-36 | 9 | 1 |
| $L_{11-6} = 18,045$ | 49-33 | 2 | 0,67 |
| $L_{11-12} = 42,63$ | 49-81; 81-83; 83-97 | 1 | 0,33 |

6. Операционная графовая модель программы.

Операционная графовая модель программы представлена на рисунке 3.

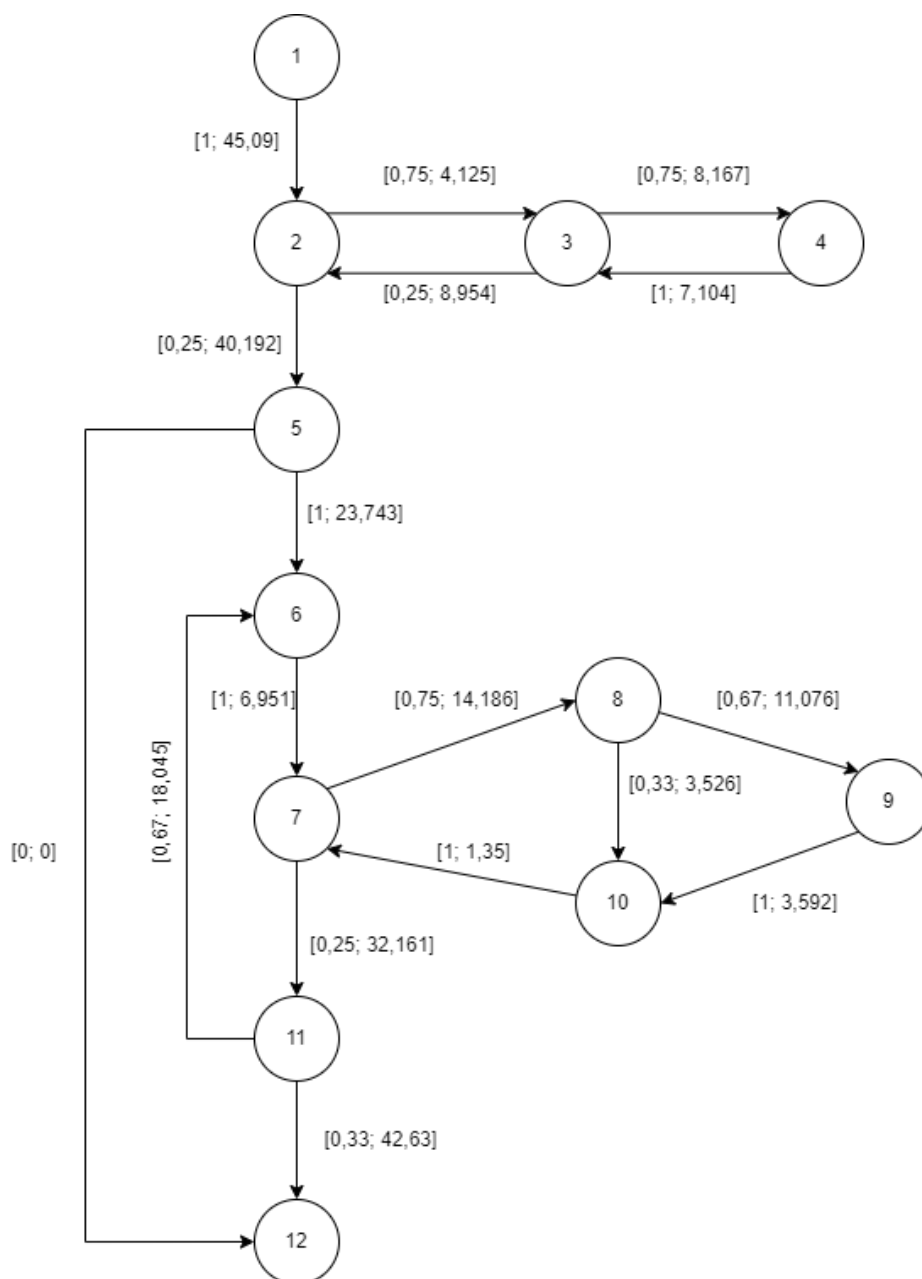


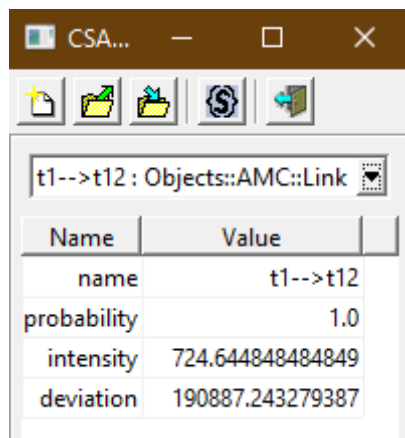
Рисунок 3 - Операционная модель

7. Описание модели model.xml.

Описание модели представлено в приложении В.

8. Результаты.

Результаты работы программы представлены на рисунке 4.



The screenshot shows a window titled 'CSA...' with a toolbar containing icons for file operations and a search icon. Below the toolbar is a text field containing 't1-->t12 : Objects::AMC::Link'. Below this is a table with the following data:

| Name | Value |
|-------------|------------------|
| name | t1-->t12 |
| probability | 1.0 |
| intensity | 724.644848484849 |
| deviation | 190887.243279387 |

Рисунок 4 - Результат работы программы

Согласно расчётам программы, среднее время выполнения составляет 724,645 мкс. В пункте 4 данного отчёта приведен результат профилирования программы с использованием SAMPLER_v2, где суммарное время выполнения составило 719,184 мкс. В итоге, разница между результатами составляет менее 1 %.

Выводы.

В ходе выполнения лабораторной работы была построена операционная графовая модель заданной программы, нагрузочные параметры которой были оценены с помощью профилировщика SAMPLER_v2 и методом эквивалентных преобразований с помощью пакета CSA III были вычислены математическое ожидание и дисперсия времени выполнения для всей программы. Результаты сравнения этих характеристик с полученными в ходе выполнения лабораторной работы №3 согласуются.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <stdio.h>
#include <stdlib.h>

#define RMAX 3
#define CMAX 3

void get_data(double a[RMAX][CMAX], double y[CMAX]) {
    a[0][0] = 1;
    a[0][1] = -43;
    a[0][2] = 19;
    y[0] = 81;
    a[1][0] = 145;
    a[1][1] = -134;
    a[1][2] = 99;
    y[1] = 12;
    a[2][0] = 325;
    a[2][1] = 991;
    a[2][2] = -199;
    y[2] = 213;
}

double deter(double a[RMAX][CMAX]) {
    return(a[0][0] * (a[1][1] * a[2][2] - a[2][1] * a[1][2])
        - a[0][1] * (a[1][0] * a[2][2] - a[2][0] * a[1][2])
        + a[0][2] * (a[1][0] * a[2][1] - a[2][0] * a[1][1]));
}

void setup(double a[RMAX][CMAX], double b[RMAX][CMAX], double
y[CMAX], double coef[CMAX], int j, double det) {
    int i;
    for (i = 0; i < RMAX; i++) {
        b[i][j] = y[i];
        if (j > 0) {
            b[i][j - 1] = a[i][j - 1];
        }
    }
    coef[j] = deter(b) / det;
}

void solve(double a[RMAX][CMAX], double y[CMAX], double
coef[CMAX]) {
    double b[RMAX][CMAX];
    int i, j;
    double det;

    for (i = 0; i < RMAX; i++) {
        for (j = 0; j < CMAX; j++) {
```



```

        b[i][j] = a[i][j];
    }
}

det = deter(b);
if (det == 0) {
    return;
}
else {
    setup(a, b, y, coef, 0, det);
    setup(a, b, y, coef, 1, det);
    setup(a, b, y, coef, 2, det);
}
}

int main() {
    double a[RMAX][CMAX];
    double y[CMAX];
    double coef[CMAX];

    get_data(a, y);
    solve(a, y, coef);

    return 0;
}

```

ПРИЛОЖЕНИЕ Б.

КОД ПРОГРАММЫ С РАЗДЕЛЕНИЕМ НА ФУ

```
1. #include <stdio.h>
2. #include <stdlib.h>
3. #include "sampler.h"
4.
5. #define RMAX 3
6. #define CMAX 3
7.
8.
9. void get_data(double a[RMAX][CMAX], double y[CMAX]) {
10.     a[0][0] = 1;
11.     a[0][1] = -43;
12.     a[0][2] = 19;
13.     y[0] = 81;
14.     a[1][0] = 145;
15.     a[1][1] = -134;
16.     a[1][2] = 99;
17.     y[1] = 12;
18.     a[2][0] = 325;
19.     a[2][1] = 991;
20.     a[2][2] = -199;
21.     y[2] = 213;
22. }
23.
24.
25. double deter(double a[RMAX][CMAX]) {
26.     return(a[0][0] * (a[1][1] * a[2][2] - a[2][1] * a[1][2])
27.         - a[0][1] * (a[1][0] * a[2][2] - a[2][0] * a[1][2])
28.         + a[0][2] * (a[1][0] * a[2][1] - a[2][0] * a[1][1]));
29. }
30.
31.
32. void setup(double a[RMAX][CMAX], double b[RMAX][CMAX], double
y[CMAX], double coef[CMAX], int j, double det) {
33.     SAMPLE;
34.     int i;
35.     SAMPLE;
36.     for (i = 0; SAMPLE, i < RMAX; i++) {
37.         SAMPLE;
38.         b[i][j] = y[i];
39.         SAMPLE;
40.         if (j > 0) {
41.             SAMPLE;
42.             b[i][j - 1] = a[i][j - 1];
43.             SAMPLE;
44.         }
45.         SAMPLE;
46.     }
47.     SAMPLE;
48.     coef[j] = deter(b) / det;
```

```

49.     SAMPLE;
50. }
51.
52.
53. void solve(double a[RMAX][CMAX], double y[CMAX], double
coef[CMAX]) {
54.     SAMPLE;
55.     double b[RMAX][CMAX];
56.     int i, j;
57.     double det;
58.     SAMPLE;
59.     for (i = 0; SAMPLE, i < RMAX; i++) {
60.         SAMPLE;
61.         for (j = 0; SAMPLE, j < CMAX; j++) {
62.             SAMPLE;
63.             b[i][j] = a[i][j];
64.             SAMPLE;
65.         }
66.         SAMPLE;
67.     }
68.     SAMPLE;
69.
70.     det = deter(b);
71.     SAMPLE;
72.     if (det == 0) {
73.         SAMPLE;
74.         return;
75.     }
76.     else {
77.         SAMPLE;
78.         setup(a, b, y, coef, 0, det);
79.         setup(a, b, y, coef, 1, det);
80.         setup(a, b, y, coef, 2, det);
81.         SAMPLE;
82.     }
83.     SAMPLE;
84. }
85.
86.
87. int main(int argc, char **argv)
88. {
89.     sampler_init(&argc, argv);
90.     double a[RMAX][CMAX];
91.     double y[CMAX];
92.     double coef[CMAX];
93.
94.     get_data(a, y);
95.     SAMPLE;
96.     solve(a, y, coef);
97.     SAMPLE;
98.
99.     return 0;
100. }

```

ПРИЛОЖЕНИЕ В.

ОПИСАНИЕ МОДЕЛИ .XML

```
<model type = "Objects::AMC::Model" name = "model">
  <node type = "Objects::AMC::Top" name = "t1"></node>
  <node type = "Objects::AMC::Top" name = "t2"></node>
  <node type = "Objects::AMC::Top" name = "t3"></node>
  <node type = "Objects::AMC::Top" name = "t4"></node>
  <node type = "Objects::AMC::Top" name = "t5"></node>
  <node type = "Objects::AMC::Top" name = "t6"></node>
  <node type = "Objects::AMC::Top" name = "t7"></node>
  <node type = "Objects::AMC::Top" name = "t8"></node>
  <node type = "Objects::AMC::Top" name = "t9"></node>
  <node type = "Objects::AMC::Top" name = "t10"></node>
  <node type = "Objects::AMC::Top" name = "t11"></node>
  <node type = "Objects::AMC::Top" name = "t12"></node>
  <link type = "Objects::AMC::Link" name = "t1-->t2" probability
= "1.0" intensity = "45.09" deviation = "0.0" source = "t1" dest =
"t2"></link>
  <link type = "Objects::AMC::Link" name = "t2-->t3" probability
= "0.75" intensity = "4.125" deviation = "0.0" source = "t2" dest =
"t3"></link>
  <link type = "Objects::AMC::Link" name = "t3-->t2" probability
= "0.25" intensity = "8.954" deviation = "0.0" source = "t3" dest =
"t2"></link>
  <link type = "Objects::AMC::Link" name = "t3-->t4" probability
= "0.75" intensity = "8.167" deviation = "0.0" source = "t3" dest =
"t4"></link>
  <link type = "Objects::AMC::Link" name = "t4-->t3" probability
= "1.0" intensity = "7.104" deviation = "0.0" source = "t4" dest =
"t3"></link>
  <link type = "Objects::AMC::Link" name = "t2-->t5" probability
= "0.25" intensity = "40.192" deviation = "0.0" source = "t2" dest =
"t5"></link>
  <link type = "Objects::AMC::Link" name = "t5-->t6" probability
= "1.0" intensity = "23.743" deviation = "0.0" source = "t5" dest =
"t6"></link>
  <link type = "Objects::AMC::Link" name = "t5-->t12"
probability = "0.0" intensity = "0.0" deviation = "0.0" source = "t5"
dest = "t12"></link>
  <link type = "Objects::AMC::Link" name = "t6-->t7" probability
= "1.0" intensity = "6.951" deviation = "0.0" source = "t6" dest =
"t7"></link>
  <link type = "Objects::AMC::Link" name = "t7-->t8" probability
= "0.75" intensity = "14.186" deviation = "0.0" source = "t7" dest =
"t8"></link>
  <link type = "Objects::AMC::Link" name = "t7-->t11"
probability = "0.25" intensity = "32.161" deviation = "0.0" source =
"t7" dest = "t11"></link>
```

```

    <link type = "Objects::AMC::Link" name = "t8-->t9" probability
= "0.67" intensity = "11.076" deviation = "0.0" source = "t8" dest =
"t9"></link>
    <link type = "Objects::AMC::Link" name = "t8-->t10"
probability = "0.33" intensity = "3.526" deviation = "0.0" source = "t8"
dest = "t10"></link>
    <link type = "Objects::AMC::Link" name = "t9-->t10"
probability = "1.0" intensity = "3.592" deviation = "0.0" source = "t9"
dest = "t10"></link>
    <link type = "Objects::AMC::Link" name = "t10-->t7"
probability = "1.0" intensity = "1.35" deviation = "0.0" source = "t10"
dest = "t7"></link>
    <link type = "Objects::AMC::Link" name = "t11-->t6"
probability = "0.67" intensity = "18.045" deviation = "0.0" source =
"t11" dest = "t6"></link>
    <link type = "Objects::AMC::Link" name = "t11-->t12"
probability = "0.33" intensity = "42.63" deviation = "0.0" source =
"t11" dest = "t12"></link>
</model>

```