**Завдання до практичної роботи**

1. Розглянути алгоритм Петрі-об’єткного моделювання, реалізований в бібліотеці PetriObjModelPaint (див. github StetsenkoInna). Виконати тестування запропонованого алгоритму на моделі мережі маcового обслуговування. **15 балів.**
2. За текстом завдання 2 практикуму 5 розробити відповідні Петрі-об’єкти та побудувати Петрі-об’єктну модель системи. Отримати результати імітаційного моделювання. Зробити висновки про функціонування моделі. **25 балів.**
3. За текстом завдання 3 практикуму 5 розробити відповідні Петрі-об’єкти та побудувати Петрі-об’єктну модель системи. Отримати результати імітаційного моделювання. Зробити висновки про функціонування моделі. **25 балів.**
4. Побудувати математичні рівняння, що описують побудовану за текстом завдання 1 практикуму 5 Петрі-об’єктну модель. **20 балів.**
5. Сформулювати переваги та недоліки використання технології Петрі-об’єктного моделювання. **5 балів.**

**Хід роботи**

1. Розглянути алгоритм Петрі-об’єткного моделювання, реалізований в бібліотеці PetriObjModelPaint (див. github StetsenkoInna). Виконати тестування запропонованого алгоритму на моделі мережі маcового обслуговування. **15 балів.**

*Результат виконання:*

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

1. Розробити відповідні Петрі-об’єкти та побудувати Петрі-об’єктну модель системи. Отримати результати імітаційного моделювання. Зробити висновки про функціонування моделі. **25 балів.**

*Завдання 2 практикум 5:*

Розробити мережу Петрі для наступної задачі (25 балів): Експериментальна роботизована гнучка виробнича система має два верстати із числовим пультом керування, три роботи, пункт прибуття і склад оброблених деталей. Деталі прибувають на пункт прибуття кожні 40 секунд згідно з експоненціальним законом розподілу, захоплюються одним з вільних роботів і переміщуються ним до першого верстата, після чого робот звільняється. Після завершення обробки на першому верстаті деталь захоплюється одним з роботів і переміщується на другий верстат, а після обробки на другому верстаті – одним з роботів переміщується на склад оброблених деталей. Кожний з верстатів може одночасно обробляти до трьох деталей. Час переміщення робота між пунктом прибуття та першим верстатом, першим і другим верстатом, другим верстатом та пунктом зберігання оброблених деталей складає відповідно 6, 7, і 5 секунд незалежно від того, холостий це хід, чи ні. Роботу потрібний час 8±1 секунд на захоплення або вивільнення деталей. Час обробки на першому верстаті розподілений за нормальним законом із середнім значення 60 секунд і стандартним відхиленням 10 секунд. Середній час обробки на другому верстаті дорівнює 100 секунд і має експоненціальний закон розподілу.

Метою моделювання є визначення найкращого (з точки зору підвищення пропускної здатності гнучкої виробничої системи) способу закріплення роботів до операцій. Можливі варіанти закріплення: 1) по одному роботу на кожний з трьох шляхів переміщення деталей (пункт прибуття – перший верстат, перший верстат – другий верстат, другий верстат, склад); 2) кожний робот може використовуватися на кожному шляху переміщення деталей(при цьому повинен займатися найближчий з роботів).

Изображение выглядит как текст, диаграмма, План, линия

Автоматически созданное описание

*Отримані методи для Петрі-об’єктів:*

*Прибуття:*

Изображение выглядит как текст, Шрифт, линия, число

Автоматически созданное описание

*Переміщення:*

Изображение выглядит как диаграмма, линия, круг, Шрифт

Автоматически созданное описание

public static PetriNet CreateNetTask7.2(double t, double t) throws ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

ArrayList<PetriP> d\_P = new ArrayList<>();

ArrayList<PetriT> d\_T = new ArrayList<>();

ArrayList<ArcIn> d\_In = new ArrayList<>();

ArrayList<ArcOut> d\_Out = new ArrayList<>();

d\_P.add(new PetriP("Пункт прибуття",0));

d\_P.add(new PetriP("Деталь захоплена",0));

d\_P.add(new PetriP("Деталь переміщена",0));

d\_P.add(new PetriP("Робот звільнений",0));

d\_P.add(new PetriP("Робот у доступності",0));

d\_P.add(new PetriP("Переміщення деталі",0));

d\_T.add(new PetriT("Захоплення деталі роботом",8.0));

d\_T.get(0).setDistribution("norm", d\_T.get(0).getTimeServ());

d\_T.get(0).setParamDeviation(0.0);

d\_T.add(new PetriT("Переміщення деталі до верстату",t));

d\_T.add(new PetriT("Звільнення робота",8.0));

d\_T.get(2).setDistribution("norm", d\_T.get(2).getTimeServ());

d\_T.get(2).setParamDeviation(0.0);

d\_T.add(new PetriT("Переміщення роботу до пункту прибуття",t));

d\_In.add(new ArcIn(d\_P.get(0),d\_T.get(0),1));

d\_In.add(new ArcIn(d\_P.get(1),d\_T.get(1),1));

d\_In.add(new ArcIn(d\_P.get(2),d\_T.get(2),1));

d\_In.add(new ArcIn(d\_P.get(3),d\_T.get(3),1));

d\_In.add(new ArcIn(d\_P.get(4),d\_T.get(0),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(1),1));

d\_Out.add(new ArcOut(d\_T.get(1),d\_P.get(2),1));

d\_Out.add(new ArcOut(d\_T.get(2),d\_P.get(3),1));

d\_Out.add(new ArcOut(d\_T.get(3),d\_P.get(4),1));

d\_Out.add(new ArcOut(d\_T.get(2),d\_P.get(5),1));

PetriNet d\_Net = new PetriNet("Task7.2",d\_P,d\_T,d\_In,d\_Out);

PetriP.initNext();

PetriT.initNext();

ArcIn.initNext();

ArcOut.initNext();

return d\_Net;

}

*Обробка машиною:*

Изображение выглядит как снимок экрана, диаграмма, круг, дизайн

Автоматически созданное описание

public static PetriNet CreateNetMachine() throws ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

ArrayList<PetriP> d\_P = new ArrayList<>();

ArrayList<PetriT> d\_T = new ArrayList<>();

ArrayList<ArcIn> d\_In = new ArrayList<>();

ArrayList<ArcOut> d\_Out = new ArrayList<>();

d\_P.add(new PetriP("Переміщені деталі",0));

d\_P.add(new PetriP("Оброблені деталі",0));

d\_P.add(new PetriP("Верстат",3));

d\_T.add(new PetriT("Обробка деталі",60.0));

d\_T.get(0).setDistribution("norm", d\_T.get(0).getTimeServ());

d\_T.get(0).setParamDeviation(0.0);

d\_In.add(new ArcIn(d\_P.get(0),d\_T.get(0),1));

d\_In.add(new ArcIn(d\_P.get(2),d\_T.get(0),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(1),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(2),1));

PetriNet d\_Net = new PetriNet("Untitled",d\_P,d\_T,d\_In,d\_Out);

PetriP.initNext();

PetriT.initNext();

ArcIn.initNext();

ArcOut.initNext();

return d\_Net;

}

Результати симуляції:

Изображение выглядит как текст, чек, Шрифт, белый

Автоматически созданное описание

Виводимо прибуток.

1. За текстом завдання 3 практикуму 5 розробити відповідні Петрі-об’єкти та побудувати Петрі-об’єктну модель системи. Отримати результати імітаційного моделювання. Зробити висновки про функціонування моделі. **25 балів.**

*Завдання 3 практикум 5:*

Розробити мережу Петрі для наступної задачі (25 балів): На маршруті приміського сполучення працюють два мікроавтобуси (А і В), кожний з яких має n місць. Мікроавтобус А користується більшою популярністю, ніж автобус В, оскільки водій мікроавтобуса А їздить акуратніше і швидше. Тому пасажир, який підійшов до зупинки, сідає в мікроавтобус В тільки у випадку, коли автобуса А немає. Мікроавтобус відправляється на маршрут, якщо всі місця в ньому зайняті. Пасажири підходять до зупинки через 0,5±0,2 хвилин і , якщо немає мікроавтобусів, утворюють чергу. Якщо черга більша, ніж 30 осіб, то пасажир не стає у чергу і йде до іншого маршруту. Припускається, що всі пасажири їдуть до кінця маршруту. На проходження маршруту мікроавтобус А витрачає 20±5 хвилин, а мікроавтобус В – 30±5 хвилин. Після того, як пасажири звільнили автобус (протягом часу 5±1 хвилин), він їде у зворотному напрямку тим же чином. Плата за проїзд складає 20 гривень. Авто підприємство стільки ж втрачає (недоотримує), якщо пасажир, прийшовши на зупинку, не стає у чергу і обирає інший маршрут.

Метою моделювання є визначення таких характеристик:

- час очікування пасажира у черзі;

- кількість місць n (не більше 25), при якому час очікування в черзі пасажира буде мінімальним;

- виручку автопідприємства за день від маршруту, якщо мікроавтобуси працюють 10 годин на добу.

*Прибуття:*

Изображение выглядит как диаграмма, линия, текст, круг

Автоматически созданное описание

Изображение выглядит как диаграмма, линия, круг, текст

Автоматически созданное описание

Изображение выглядит как диаграмма, линия, круг, снимок экрана

Автоматически созданное описание

public static PetriNet IncomeStation\_7\_3() throws ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

ArrayList<PetriP> d\_P = new ArrayList<>();

ArrayList<PetriT> d\_T = new ArrayList<>();

ArrayList<ArcIn> d\_In = new ArrayList<>();

ArrayList<ArcOut> d\_Out = new ArrayList<>();

d\_P.add(new PetriP("",1));

d\_P.add(new PetriP("",0));

d\_P.add(new PetriP("Черга",0));

d\_P.add(new PetriP("Втрачено",0));

d\_P.add(new PetriP("К-сть що були в цій черзі",0));

d\_T.add(new PetriT("Надходження",0.5));

d\_T.get(0).setDistribution("norm", d\_T.get(0).getTimeServ());

d\_T.get(0).setParamDeviation(0.1);

d\_T.add(new PetriT("Стати в чергу",0));

d\_T.add(new PetriT("Вихід",0));

d\_T.get(2).setPriority(1);

d\_In.add(new ArcIn(d\_P.get(0),d\_T.get(0),1));

d\_In.add(new ArcIn(d\_P.get(1),d\_T.get(1),1));

d\_In.add(new ArcIn(d\_P.get(1),d\_T.get(2),1));

d\_In.add(new ArcIn(d\_P.get(2),d\_T.get(2),30));

d\_In.get(3).setInf(true);

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(0),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(1),1));

d\_Out.add(new ArcOut(d\_T.get(1),d\_P.get(2),1));

d\_Out.add(new ArcOut(d\_T.get(2),d\_P.get(3),1));

PetriNet d\_Net = new PetriNet("IncomeStation\_5\_3",d\_P,d\_T,d\_In,d\_Out);

PetriP.initNext();

PetriT.initNext();

ArcIn.initNext();

ArcOut.initNext();

return d\_Net;

}

public static PetriNet Ride\_7\_3(int b, int n, int p, double Trtime) throws ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

ArrayList<PetriP> d\_P = new ArrayList<>();

ArrayList<PetriT> d\_T = new ArrayList<>();

ArrayList<ArcIn> d\_In = new ArrayList<>();

ArrayList<ArcOut> d\_Out = new ArrayList<>();

d\_P.add(new PetriP("Черга",0));

d\_P.add(new PetriP("",0));

d\_P.add(new PetriP("Автобус на станції",0));

d\_P.add(new PetriP("Автобус на ст. Б",0));

d\_P.add(new PetriP("Виконаний переїзд",0));

d\_T.add(new PetriT("Посадка або висадка",5.0));

d\_T.get(0).setDistribution("norm", d\_T.get(0).getTimeServ());

d\_T.get(0).setParamDeviation(1.0);

d\_T.add(new PetriT("Поїздка",Trtime));

d\_T.get(1).setDistribution("norm", d\_T.get(1).getTimeServ());

d\_T.get(1).setParamDeviation(5.0);

d\_In.add(new ArcIn(d\_P.get(0),d\_T.get(0),n));

d\_In.add(new ArcIn(d\_P.get(2),d\_T.get(1),1));

d\_In.add(new ArcIn(d\_P.get(1),d\_T.get(1),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(1),1));

d\_Out.add(new ArcOut(d\_T.get(1),d\_P.get(3),1));

d\_Out.add(new ArcOut(d\_T.get(1),d\_P.get(4),1));

PetriNet d\_Net = new PetriNet("Ride\_5\_3",d\_P,d\_T,d\_In,d\_Out);

PetriP.initNext();

PetriT.initNext();

ArcIn.initNext();

ArcOut.initNext();

return d\_Net;

}

public static PetriNet Money\_7\_3() throws ExceptionInvalidNetStructure, ExceptionInvalidTimeDelay {

ArrayList<PetriP> d\_P = new ArrayList<>();

ArrayList<PetriT> d\_T = new ArrayList<>();

ArrayList<ArcIn> d\_In = new ArrayList<>();

ArrayList<ArcOut> d\_Out = new ArrayList<>();

d\_P.add(new PetriP("Виконаний переїзд",0));

d\_P.add(new PetriP("Прибуток",0));

d\_T.add(new PetriT("",0.0));

d\_In.add(new ArcIn(d\_P.get(0),d\_T.get(0),1));

d\_Out.add(new ArcOut(d\_T.get(0),d\_P.get(1),20));

PetriNet d\_Net = new PetriNet("Ride\_5\_3",d\_P,d\_T,d\_In,d\_Out);

PetriP.initNext();

PetriT.initNext();

ArcIn.initNext();

ArcOut.initNext();

return d\_Net;

}

package LibTest;

//import PetriObj.PetriObjModel;

import LibNet.NetLibrary;

import PetriObj.ExceptionInvalidNetStructure;

import PetriObj.ExceptionInvalidTimeDelay;

import PetriObj.PetriObjModel;

import PetriObj.PetriSim;

import java.util.ArrayList;

public class Lab7\_3 {

public static void main(String[] args) throws ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure {

// цей фрагмент для запуску імітації моделі з заданною мережею Петрі на інтервалі часу timeModeling

PetriObjModel model = getModel();

model.setIsProtokol(false);

double timeModeling = 1000;

model.go(timeModeling);

for (int j = 0; j < 8; j++) {

model.getListObj().get(j).printMark();

}

}

public static PetriObjModel getModel() throws ExceptionInvalidTimeDelay, ExceptionInvalidNetStructure{

ArrayList<PetriSim> list = new ArrayList<>();

list.add(new PetriSim(NetLibrary.IncomeStation\_5\_3())); // Generator on Left

list.add(new PetriSim(NetLibrary.Ride\_5\_3(1,20,2,20))); //Bus A on Left

list.add(new PetriSim(NetLibrary.Ride\_5\_3(1,20,1,30))); //Bus A on Right

list.add(new PetriSim(NetLibrary.IncomeStation\_5\_3())); // Generator on Right

list.add(new PetriSim(NetLibrary.Ride\_5\_3(0,20,2,20))); //Bus B on Left

list.add(new PetriSim(NetLibrary.Ride\_5\_3(0,20,1,30))); //Bus B on Right

list.add(new PetriSim(NetLibrary.Money\_5\_3())); // Lost Profit

list.add(new PetriSim(NetLibrary.Money\_5\_3())); // Income

//

list.get(1).getNet().getListP()[0] = list.get(0).getNet().getListP()[2];

list.get(2).getNet().getListP()[0] = list.get(0).getNet().getListP()[2];

list.get(0).getNet().getListP()[3] = list.get(6).getNet().getListP()[0];

list.get(4).getNet().getListP()[0] = list.get(3).getNet().getListP()[2];

list.get(5).getNet().getListP()[0] = list.get(3).getNet().getListP()[2];

list.get(3).getNet().getListP()[3] = list.get(6).getNet().getListP()[0];

list.get(4).getNet().getListP()[3] = list.get(1).getNet().getListP()[2];

list.get(1).getNet().getListP()[3] = list.get(4).getNet().getListP()[2];

list.get(5).getNet().getListP()[3] = list.get(2).getNet().getListP()[2];

list.get(2).getNet().getListP()[3] = list.get(5).getNet().getListP()[2];

list.get(1).getNet().getListP()[4] = list.get(7).getNet().getListP()[0];

list.get(2).getNet().getListP()[4] = list.get(7).getNet().getListP()[0];

list.get(4).getNet().getListP()[4] = list.get(7).getNet().getListP()[0];

list.get(5).getNet().getListP()[4] = list.get(7).getNet().getListP()[0];

PetriObjModel model = new PetriObjModel(list);

return model;

}

}

Результат роботи:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Побудувати математичні рівняння, що описують побудовану за текстом завдання 1 практикуму 5 Петрі-об’єктну модель.

Изображение выглядит как зарисовка, диаграмма, линия, рисунок

Автоматически созданное описание

Петрі-об’єктна модель описується стохастичною мережею Петрі, що є об’єднанням Петрі об’єктів, з яких вона складається.

, де

Дана Петрі-об’єктна модель складається з таких об’єктів: «генератор деталей», «пристрій»(5 шт.), «результат обробки».

//Позиції

//Переходи

//Дуги

//Кратність дуг

//Пріорітет/ймовірність

//Час

Розглянемо кожен об’єкт окремо.

*Генератор деталей*

*Пристрій*

*Результат обробки*

**Завдання 5**

|  |  |
| --- | --- |
| Переваги | Недоліки |
| Узагальнення при конструюванні | Складно конфігурувати |
| Поділ мережі на частини | Обмеження гнучкості |
| Можливість повторного використання |  |
| Швидкодія |  |