Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Комп’ютерний практикум №6

Моделювання систем

Тема: Застосування алгоритму стохастичної мережі Петрі для реалізації моделей дискретно-подійних систем

Виконав
студент групи ІП-11:
Панченко С. В.

Перевірила:
Дифучина О. Ю.

Київ 2024

ЗМІСТ

1 МЕТА

Застосувати алгоритм стохастичної мережі Петрі для реалізації моделей дискретно-подійних систем.

2 ЗАВДАННЯ

1. Ознайомитись з бібліотекою класів PetriObjModelPaint моделювання діскретно-подійних сістем на основі стохастичних мереж Петрі та графічним редактором мережі Петрі. 10 балів.
2. З використанням алгоритму імітації стохастичної мережі Петрі класу PetriSim реалізувати модель, розроблену за текстом завдання 1 практикуму 5, та виконати її верифікацію. Зробити висновки про функціонування моделі. 25 балів.
3. З використанням алгоритму імітації стохастичної мережі Петрі класу PetriSim реалізувати модель, розроблену за текстом завдання 4 практикуму 5, та виконати її верифікацію. Зробити висновки про функціонування моделі. 25 балів.
4. Побудувати модель системи, що відтворює обробку потоку запитів головним та допоміжним сервером. Ймовірність звернення до допоміжного сервера 0,3. Часові характеристики обробки запитів задайте самостійно. 20 балів.
5. Побудувати математичні рівняння, що описують побудовану за текстом завдання 4 мережу Петрі. 20 балів.

3 ВИКОНАННЯ

3.1 Завдання 1

Побудуємо простий генератор на рисунку 3.1:

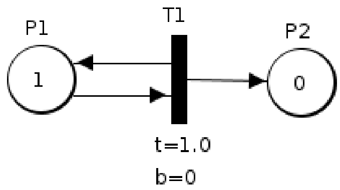

Рисунок 3.1 — Простий генератор

3.2 Завдання 2

Побудуємо мережу Петрі на рисунку 3.2.

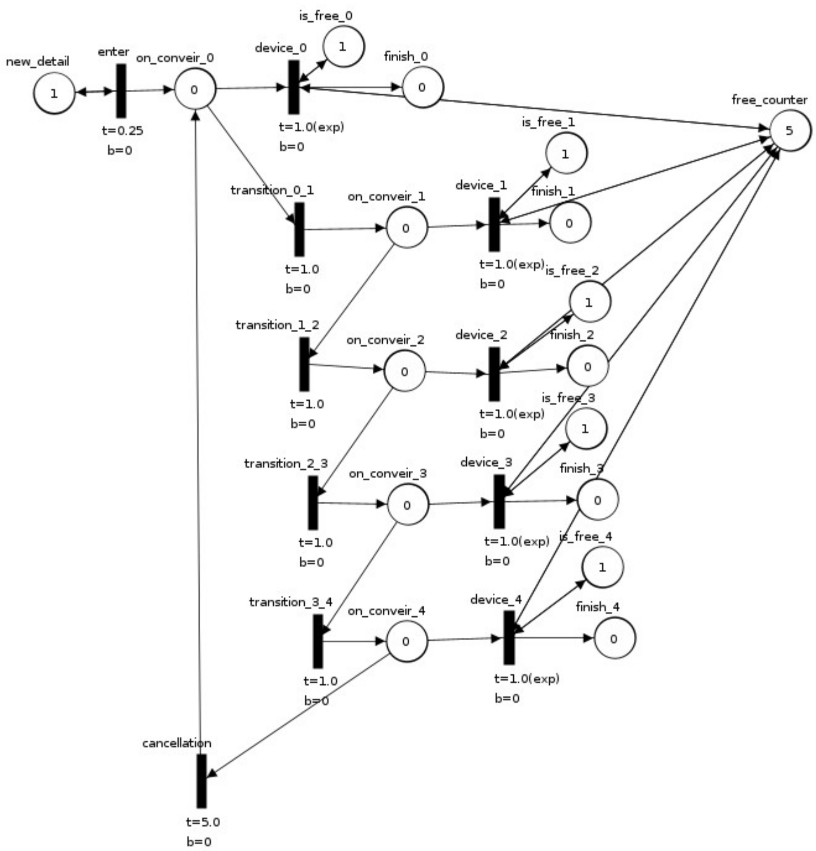

Рисунок 3.2 — Мережа Петрі

Проведемо верифікацію, результати зображені у таблиці 3.1.

Таблиця 3.1 Результати верифікації

N	T_{enter}	T_{device}	T_{trans}	T_{cancel}	N_0	N_1	N_2	N_3	N_5
0	0.25	1.0	1.0	5.0	893	851	799	737	674
1	0.5	2.0	2.0	10.0	400	420	405	382	334
2	0.125	2.0	2.0	1.0	508	511	469	464	480
3	0.5	0.6	0.2	3.0	1143	654	180	21	1
4	0.3	0.6	0.5	2.0	1345	1055	609	253	61

3.3 Завдання 3

На рисунку 3.3 зображено мережу Петрі:

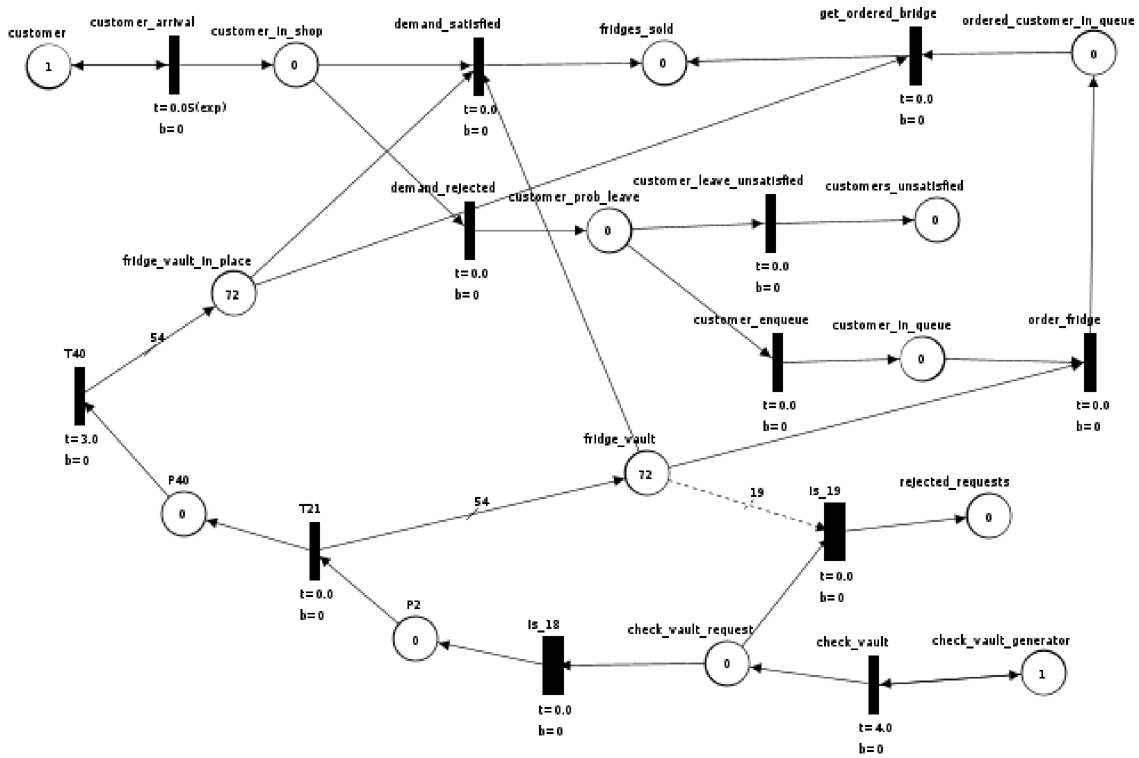

Рисунок 3.3 — Мережа Петрі

У таблиці 3.2 проведемо верифікацію моделі:

Таблиця 3.2 Результати верифікації

N	T_{customer}	$T_{\text{check vault}}$	T_{order}	N_{sold}	$N_{\text{unsatisfied}}$
0	0.2	4.0	3.0	4452	531
1	0.05	4.0	3.0	9190	11098
2	0.7	0.5	3.0	1421	1973
3	0.7	0.02	0.01	1372	0
4	1.0	0.5	0.4	970	0

3.4 Задача 4

На рисунку побудовано мережу Петрі, на рисунках зображені параметри:

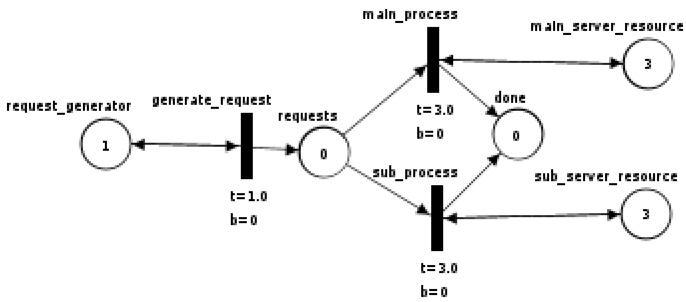

Рисунок 3.4 — Мережа Петрі

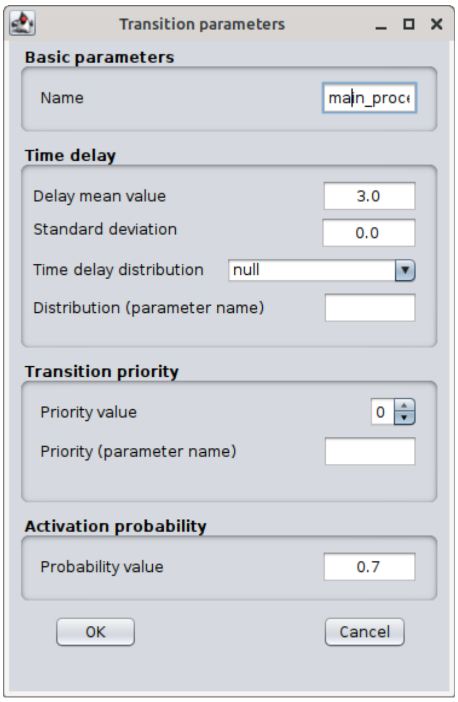

Рисунок 3.5 — Налаштування main_process

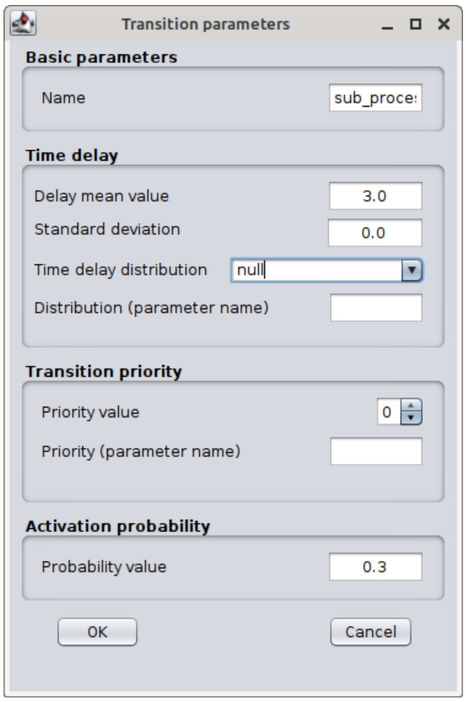

Рисунок 3.6 — Налаштування sub_process

3.5 Задача 5

Нижче зроблено формальний опис стохастичної мережі Петрі:

$N=\{P_N, T_N, A_N, W_N, K_N, I_N, R_N\}$,
$P_N=\{P\}$ — positions,
$T_N=T$ — transitions,
$P_N \cap T_N = \varnothing$,
$A_N \subseteq (P_N \times T_N \cup T_N \times P_N)$ — arcs,
$W: A \cup I \rightarrow N$ — weights,
$K=\{(c_T, b_T) \mid T \in \boldsymbol{T},\ c_T \in N,\ b_T \in [0;1]\}$ — priorities and probabilities,
$R: T \rightarrow \Re_+$ — time delay function,
${}^\bullet T,\ T^\bullet$ — set of input and output positions of a transition T,
${}^\bullet P,\ T^\bullet$ — set of input and output transitions of a position P

$$\text{positions}=\left\{\begin{matrix}\text{request\_generator}\\ \text{requests}\\ \text{done}\\ \text{main\_server\_resource}\\ \text{sub\_server\_resource}\end{matrix}\right\},\quad \text{transitions}=\left\{\begin{matrix}\text{generate\_request}\\ \text{main\_process}\\ \text{sub\_process}\end{matrix}\right\},$$

$$\text{arcs}=\left\{\begin{matrix}(\text{request generator},\ \text{generate\_request})\\ (\text{generate\_request},\ \text{request\_generator}),\\ (\text{request\_generator},\ \text{requests}),\\ (\text{requests},\ \text{main\_process}),\\ (\text{requests},\ \text{sub\_process}),\\ (\text{main\_process},\ \text{done}),\\ (\text{sub\_process},\ \text{done}),\\ (\text{main\_process},\ \text{main\_server\_resource}),\\ (\text{main\_server\_resource},\ \text{main\_process}),\\ (\text{sub\_process},\ \text{sub\_server\_resource}),\\ (\text{sub\_server\_resource},\ \text{sub\_process})\end{matrix}\right\},$$

$$\text{weights}=\{1,1,1,1,1,1,1,1,1,1,1\},\quad \text{pri\_prob}=\{(0,1.0),(0,0.7),(0,0.3)\},$$
$$\text{time\_delay\_funcs}=\{\text{unif(1.0)},\text{unif(1.0)},\text{unif(1.0)}\},\quad \text{info\_arcs}=\varnothing,$$

Для кращої зрозумілості замінив математичні символи на повноцінні назви:

S^+ — state_input;

S^- — state_output;

$Z : T \times \Re \rightarrow \{0;1\}$ — предикат, що визначає для кожного переходу умову виконання запуску — trans_predicate(transition: Transition, time_point: TimePoint) -> bool;

$M_P(t)$ — get_position_marker_count(position: Position, time_point: TimePoint) -> MarkerCount;

$E_T(t)$ — get_transition_state(transition: Transition)

$D^-(S(t_n))$ — перетворення стану мережі Петрі, пов'язане з входом маркерів в переходи;

$D^+(S(t_n))$ — перетворення стану мережі Петрі, пов'язане з виходом маркерів з переходів;

$Y : T \times \Re \rightarrow \{0;1\}$ — предикат, що визначає для кожного переходу співпадіння моменту найближчої події з поточним моментом часу — is_transition_out;

$X : T \times \Re \rightarrow \{0;1\}$ — предикат, що визначає для кожного переходу приналежність до множини переходів, вибраних в результаті вирішення конфлікту — is_in_conflict.

Опишемо початковий стан мережі:

$$\text{state\_input}(0.0) = \left\{ \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} \{\infty\} \\ \{\infty\} \\ \{\infty\} \end{pmatrix} \right\},$$

Опишемо, які з переходів можуть увійти:

$$\left\{ \begin{array}{l} \text{request\_generator} > 0 \Rightarrow \text{trans\_predicate(generate, 0.0)=1} \\ \text{requests} < 1, \text{main\_server\_resource} \geq 1 \Rightarrow \text{trans\_predicate(main\_process, 0.0)=0} \\ \text{requests} < 1, \text{sub\_server\_resource} \geq 1 \Rightarrow \text{trans\_predicate(sub\_process, 0.0)=0} \end{array} \right\},$$

Опишемо переходи, які входять:

succesful_transitions = {generate}

Опишемо, чи є конфлікти:

no conflicts

Опишемо, зміну маркерів:

get_position_marker_count(request_generator, 0.0)=1-1=0

get_position_marker_count(requests, 0.0)=0-0=0

get_position_marker_count(main_server_resource, 0.0)=1-0=1

get_position_marker_count(sub_server_resource, 0.0)=1-0=1

get_position_marker_count(done, 0.0)=0-0=0

Опишемо зміну стану переходів:

get_transition_state(generate)={0 + 1}

get_transition_state(main_process)= ∞

get_transition_state(sub_process)= ∞

Опишемо вихідний стан:

$$\mathrm{state\_output}(0.0)=\left\{\begin{pmatrix}0\\0\\1\\1\\0\end{pmatrix},\begin{pmatrix}\{1\}\\\{\infty\}\\\{\infty\}\end{pmatrix}\right\}$$

Опишемо, які з переходів виходять в даний момет часу:

is_transition_out(generate, 1) = 1

is_transition_out(main_process, 1) = 0

is_transtion_out(sub_process, 1) = 0

Опишемо зміну маркерів позицій:

get_postion_markers_count(request_generator, 1) = 0 + 1 = 0 get_postion_markers_count(requests, 1) = 0 + 1 = 1

get_postion_markers_count(main_server_resource, 1) = 1 + 0 = 1

get_postion_markers_count(sub_server_resource, 1) = 1 + 0 = 1

get_postion_markers_count(request_generator, 1) = 0 + 0 = 0

Опишемо зміну стану переходів:

get_transition_state(generate)= ∞

get_transition_state(main_process)= ∞

get_transition_state(sub_process)= ∞

Опишемо вихідний стан:

$$\text{state\_output}(0.0) = \left\{ \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} \{\infty\} \\ \{\infty\} \\ \{\infty\} \end{pmatrix} \right\}$$

Опишемо, які переходи мають увійти:

$$\left\{ \begin{matrix} \text{request\_generator} > 1 \Rightarrow \text{trans\_predicate(generate, 1)=1} \\ \text{requests} \geq 1 \, \text{main\_server\_resource} \geq 1 \Rightarrow \text{trans\_predicate(main\_process, 1)=1} \\ \text{requests} \geq 1, \text{sub\_server\_resource} \geq 1 \Rightarrow \text{trans\_predicate(sub\_process, 1)=1} \end{matrix} \right\},$$

Опишемо, які переходи входять:

succesful_transitions = {generate, main_process, sub_process}

Маємо, конфлікт:

is_in_conflict(main_process) = 1

is_in_conflict(sub_process) = 1

Припустимо, що спрацьовав main_process:

Опишемо зміну маркерів позицій:

get_postion_markers_count(request_generator, 1) = 1 - 1 = 0

get_postion_markers_count(requests, 1) = 1 - 1 = 0

get_postion_markers_count(main_server_resource, 1) = 1 - 1 = 0

get_postion_markers_count(sub_server_resource, 1) = 1 - 0 = 1

get_postion_markers_count(request_generator, 1) = 0 - 0 = 0

Опишемо зміну стану переходів:

get_transition_state(generate)={1 + 1}

get_transition_state(main_process)={1 + 1}

get_transition_state(sub_process)= ∞

Опишемо вихідний стан:

$$\text{state\_output}(0.0)=\left\{\begin{pmatrix}0\\0\\0\\1\\0\end{pmatrix},\begin{pmatrix}\{2\}\\\{2\}\\\{\infty\}\end{pmatrix}\right\}$$

Опишемо, які з переходів виходять в даний момет часу:

is_transition_out(generate, 2) = 1

is_transition_out(main_process, 2) = 0

is_transtion_out(sub_process, 2) = 0

Опишемо зміну маркерів позицій:

get_postion_markers_count(request_generator, 2) = 0 + 0 = 0

get_postion_markers_count(requests, 2) = 0 + 0 = 0

get_postion_markers_count(main_server_resource, 2) = 0 + 1 = 1

get_postion_markers_count(sub_server_resource, 2) = 1 + 0 = 1

get_postion_markers_count(request_generator, 2) = 0 + 0 = 0

Опишемо зміну стану переходів:

get_transition_state(generate)= ∞

get_transition_state(main_process)= ∞

get_transition_state(sub_process)= ∞

$$\text{state\_output}(0.0)=\left\{\begin{pmatrix}0\\0\\1\\1\\0\end{pmatrix},\begin{pmatrix}\{\infty\}\\\{\infty\}\\\{\infty\}\end{pmatrix}\right\}$$

ВИСНОВОК

Під час комп'ютерного практикуму я ознайомився з бібліотекою PetriObjModelPaint для моделювання дискретно-подійних систем на основі стохастичних мереж Петрі та вивчив графічний редактор цієї бібліотеки. Це дало змогу краще зрозуміти принципи роботи з мережами Петрі для моделювання реальних систем.

У другому завданні я реалізував модель конвеєрної системи за допомогою алгоритму імітації PetriSim, провів її верифікацію, що підтвердило правильність роботи, та зробив висновки про її динаміку.

У третьому завданні створив модель системи управління запасами холодильників, провів верифікацію та аналіз ефективності роботи системи.

Четверте завдання полягало у моделюванні системи обробки запитів двома серверами з імовірністю звернення до допоміжного сервера 0,3. Я задав часові характеристики, побудував мережу Петрі та провів її верифікацію.

У п’ятому завданні розробив математичні рівняння для аналізу переходів і позицій мережі Петрі з завдання 4, що дозволяють описати поведінку системи та передбачити її реакцію на вхідні дані.