Калініна І.О.
Гожий О.П.

Моделювання складних систем на основі кольорових мереж Петрі

Навчальний посібник

Херсон – 2021 рік

УДК 004.94(075.8)

Рекомендовано до друку
Вченою радою Чорноморського національного університету імені Петра Могили,
протокол № 6 від 30 червня 2021

Рецензенти: ***Літвіненко В.І.,*** завідувач кафедри інформатики і комп'ютерних наук Херсонського національного технічного університету, д.т.н., професор;
Малахов Є.В., завідувач кафедри математичного забезпечення комп'ютерних систем Одеського національного університету ім. І.І. Мечникова, д.т.н., професор

Калініна І.О., **Гожий, О.П.**
Моделювання складних систем на основі кольорових мереж Петрі: Навчальний посібник [Текст] / І.О. Калініна, О.П. Гожий. Херсон, вид-во ФОП Вишемирський В.С., 2021. – 58 с.

Навчальний посібник містить комплекс практичних робіт з поясненними та завданнями для самостійного виконання студентами при вивченні дисциплін «Ймовірнісно-статистичні методи моделювання та прогнозування» і «Методи та візуальні технології імітаційного моделювання». Поряд із теоретичними відомостями наводяться приклади розв'язання задач моделювання складних систем за допомогою програмного середовища CPN Tools.

Посібник призначений для студентів спеціальності 122 – «Комп'ютерні науки» та відповідає вимогам освітньої програми «Інтелектуальні інформаційні системи», а також може бути корисним для студентів спеціальностей 123 – «Комп'ютерна інженерія» та 124 – «Системний аналіз».

УДК 004.94(075.8

ISBN 1111111

ЗМІСТ

Вступ

Моделювання систем є найбільш ефективним способом дослідження складних систем різного призначення, – технічних, економічних, екологічних, соціальних, інформаційних, на різних етапах їх життєвого циклу, проектування, так і в процесі експлуатації. Сучасні технології моделювання не тільки значно полегшили і прискорили процес побудови та дослідження моделей складних систем, але й значно наблизили сприйняття інформації спеціаліста з моделювання систем і спеціаліста, що працює у галузі, яка моделюється.

Можливості моделювання систем постійно розширюються, тому що постійно з'являються найновіші методи та технології моделювання. Процес розробки та створення моделі – це складний та творчий процес, який вимагає від дослідника не тільки розуміння як функціонує об'єкт моделювання та теоретичних знань з різних розділів математики та інших технічних дисциплін, але й творчого підходу до розв'язання різноманітних завдань, Математичні моделі та методи моделювання використовуються при створенні інформаційних систем різного типу: систем прийняття рішень, систем автоматизованого керування, систем штучного інтелекту та ін.

Потреба у розв'язанні задач моделювання систем виникає не тільки у дослідників та науковців, але й у проектувальників та виробників під час проектування та моделювання складних систем. Одним з сучасних методів побудови моделей складних систем є візуальне моделювання на основі кольорових мереж Петрі. Використання мереж Петрі, як інструмента графічного і математичного моделювання складних систем та процесів останнім часом отримало широке розповсюдження. Мережа Петрі (PN) це орієнтований дводольний граф з двома типами вузлів: позиціями (представлені колами), та переходами (представлені прямокутниками). Дуги графа з'єднують позиції і переходи таким чином, що позиції можуть бути пов'язані тільки з переходами і навпаки. Позиції в мережі Петрі можуть містити дискретне число маркерів (*токенів*). Розподіл маркерів над місцями називається маркуванням. Коли К.А.Петрі вперше представив мережі Петрі, вони служили для опису паралельних систем з точки зору причинно-наслідкових зв'язків без обліку часу. Для моделювання складних систем та процесів різної природи використовуються різні модифікації мереж Петрі: стохастичні мережі Петрі (SPN), часові мережі Петрі (TPN), кольорові мережі Петрі (CPN) , нечіткі мережі Петрі та ін.

Використання методів візуального представлення та моделювання, таких як мережі Петрі на етапі розробки складних технічних систем ефективні за декількома причинами. По-перше візуальні представлення забезпечують високорівневу, але точну мову опису та представлення складної системи, яка дозволяє формально описувати та моделювати на різних рівнях абстракції. По-друге дозволяють моделювати складні системи в динаміці.

Навчальний посібник містить матеріал з імітаційного моделювання систем за допомогою кольорових мереж Петрі в середовищі *CPN Tools*. Імітаційне моделювання систем в *CPN Tools*, є найбільш практично застосовуваним підходом для дослідження складних систем, йому і присвячена найбільша частина матеріалу. Розглядаються алгоритми імітаційного моделювання систем, що базуються на представленні процесу функціонування системи засобами систем масового обслуговування та засобами мереж Петрі. Послідовність викладення матеріалу підпорядкована етапам процесу моделювання та побудови моделей в середовищі *CPN Tools*. Матеріал представлено у вигляді шести практичних робіт. Складність виконання робіт зростає поступово.

Автори сподіваються, що даний посібник допоможе студентам, опанувати цю складну, але цікаву технологію. Навчальний посібник представляє інтерес для студентів комп’ютерних спеціальностей, а також буде корисним для аспірантів та фахівців у галузі інформаційних технологій.

1. Практична робота №1. Знайомство із *CPN Tools*

У даній практичній роботі проводиться первинне знайомство з інтерфейсом та можливостями програмного середовища CPN Tools. Розглядається процес побудови моделей, визначення типів даних, об'ява змінних, завдання охоронних виразів для переходів, завдання виразів на дугах. Розглядаються допоміжні графічні елементи, що призначені для моделювання. Описується процес виконання та налагодження моделей. Розглянуті можливості CPN Tools супроводжуються прикладами.

1.1 Середовище розробки моделей складних систем

Як програмне середовище для вивчення мереж Петрі (МП) та побудови моделей в даному навчальному посібнику використовується програма, розроблена в університеті Орхуса (Данія) - *CPN Tools*. Остання версія програми може бути завантажена з сайту *http://cpntools.org/*. На сайті також є документація для користувача та навчальні приклади, що дозволяють швидко освоїти основні принципи побудови моделей в *CPN Tools*.

Відмінною особливістю *CPN Tools* є наявність великої кількості інструментів, що дозволяє аналізувати різні аспекти функціонування моделей на базі МП (безпека і обмеженість позицій, рівень активності переходів, наявність тупикових маркувань). *CPN Tools* використовується в багатьох реальних проектах в галузі телекомунікаційних технологій, систем багатоканального зв'язку, при розробці автоматизованих систем керування різного призначення, при розробці програмно-апаратних пристроїв в інтелектуальних системах, при моделюванні мереж та мережевих пристроїв, а також при верифікації протоколів зв'язку.

У даній системі моделювання для побудови складних моделей використовуються ієрархічні часові кольорові МП (ІЧК МП). ІЧК МП являє собою універсальну алгоритмічну систему та по виразній потужності еквівалентна машині Тьюринга. Таке розширення можливостей системи може бути використано для опису довільного об'єкта.

Система *CPN Tools* підтримує мову *CPN ML*, яка використовується для опису операцій, умов та функцій. Завдяки застосуванню *CPN ML* програма дозволяє спростити процес побудови та аналізу моделей складних систем.

Імітаційне моделювання за допомогою *CPN Tools* є дискретно-подієвим, що передбачає миттєву зміну стану МП в певні моменти часу, які мають назву

кроків. Наявність тимчасових міток дозволяє моделювати процеси з урахуванням часу виконання.

1.2 Початок роботи у програмному середовище CPN Tools

Після запуску *CPN Tools* користувач побачить перед собою вікно програми (рис. 1), яке розділено на дві частини: ліва - *область меню*, права - *робоча область*. В області меню розташовуються вкладка з інструментами для моделювання мережі, вкладка з документацією та вкладка з налаштуваннями програми. Після створення моделі в області меню з'явиться вкладка з елементами моделі. У робочій області розташовуються вікна зі створеними моделями.

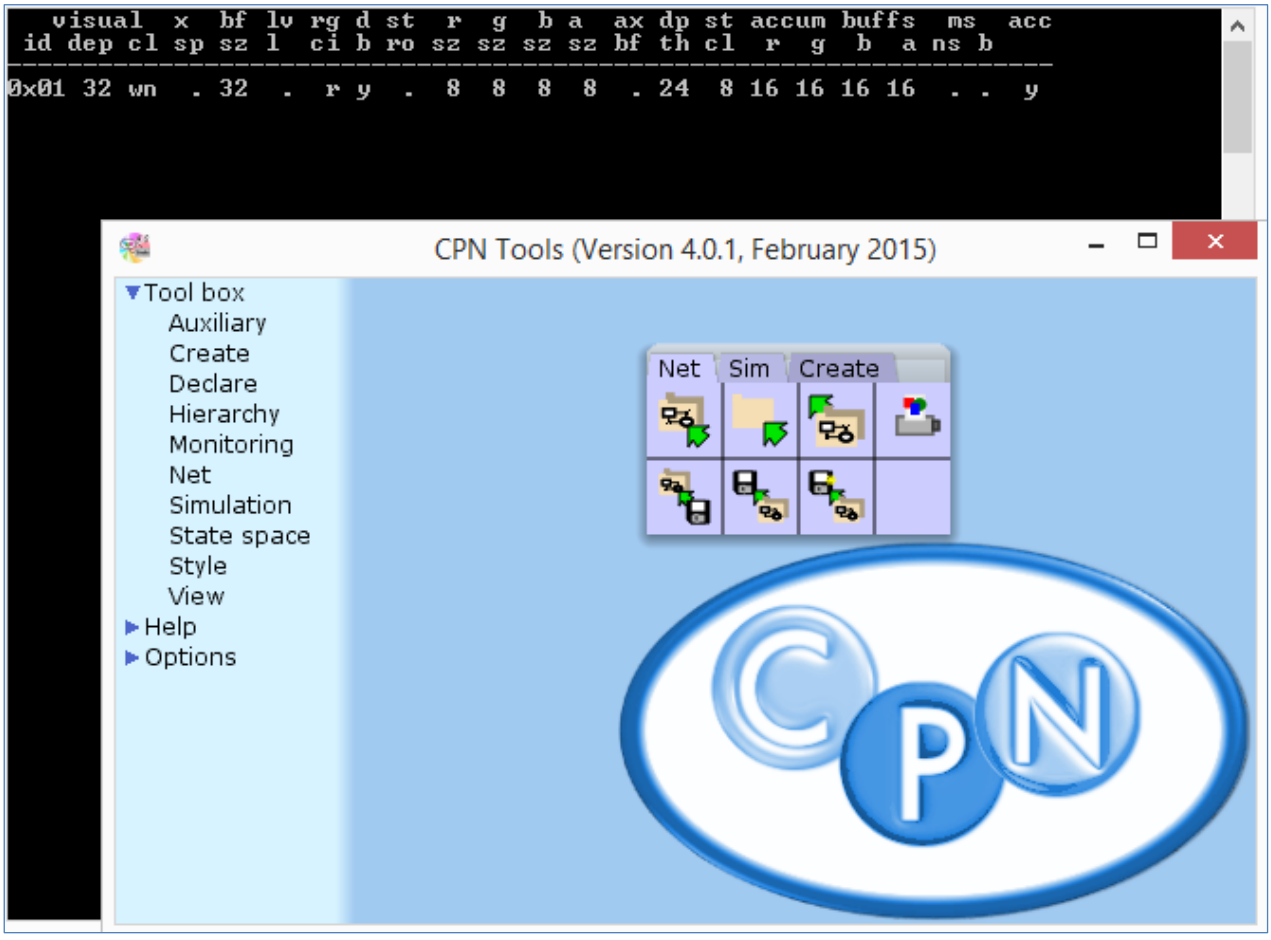


Рис. 1. Інтерфейс CPN Tools після запуску програми

З призначеного для користувача інтерфейсу доступні дві функції:

- завантаження створеної мережі «**Load Net**»;
- створення нової мережі «**New Net**».

Для створення нової мережі (рис. 2) необхідно натиснути правою

кнопкою миші в робочій області та утримуючи кнопку в меню, обрати пункт «**New net**», пересунувши на нього курсор миші. Аналогічним чином організована робота з іншими меню в програмі *CPN Tools*.

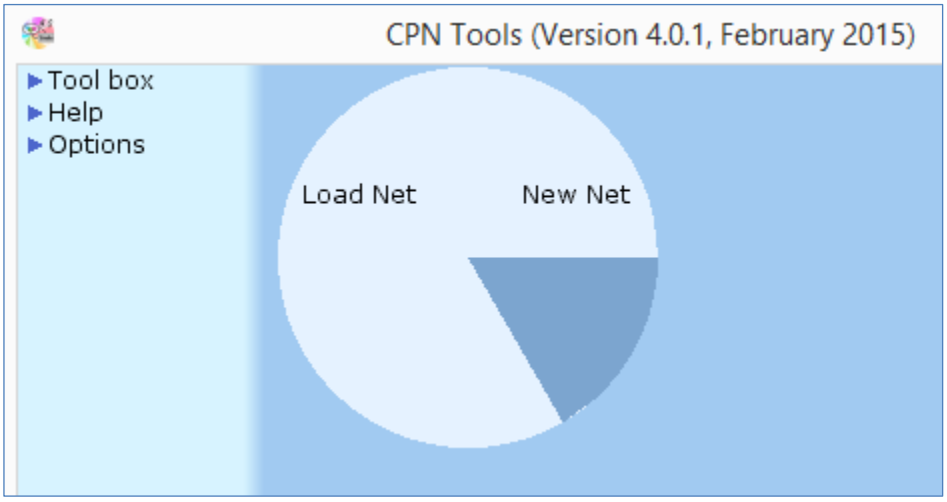


Рис. 2. Створення нової мережі в CPN Tools

Після створення нової мережі вікно програми прийме вигляд, зображений на рис. 3. При цьому в області меню з'явиться новий пункт «**New net.cpn**», розкривши який можна змінити параметри моделі, оголосити типи даних та ввести нові змінні. Крім цього правий клік миші на пункті «**New net.cpn**» відкриє контекстне меню, в якому можна буде зберегти мережу, скасувати / повторити дію, створити нову сторінку або закрити мережу.

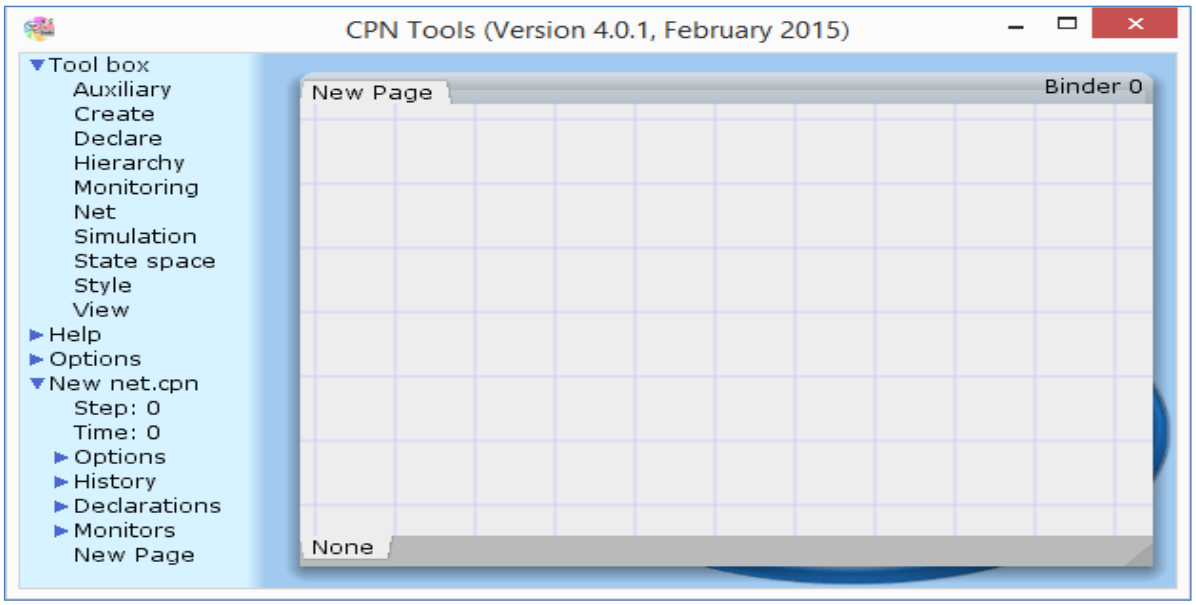


Рис. 3. Зовнішній вигляд програми CPN Tools після створення нової мережі

Перед початком роботи необхідно виконати ще кілька дій. Натисніть на пункті «**Tool box**» робочої області, в списку, що розкрився виберіть пункт «**Create**», клікніть на ньому мишкою та, утримуючи кнопку миші перетягніть на робочу область. Аналогічним чином зробіть з пунктом «**Simulation**». Після виконання даних дій будуть додані панелі для створення мережі Петрі (***Create***) та для управління процесом моделювання (***Sim***), а призначений для користувача інтерфейс програми прийме вигляд, зображений на рис. 4.

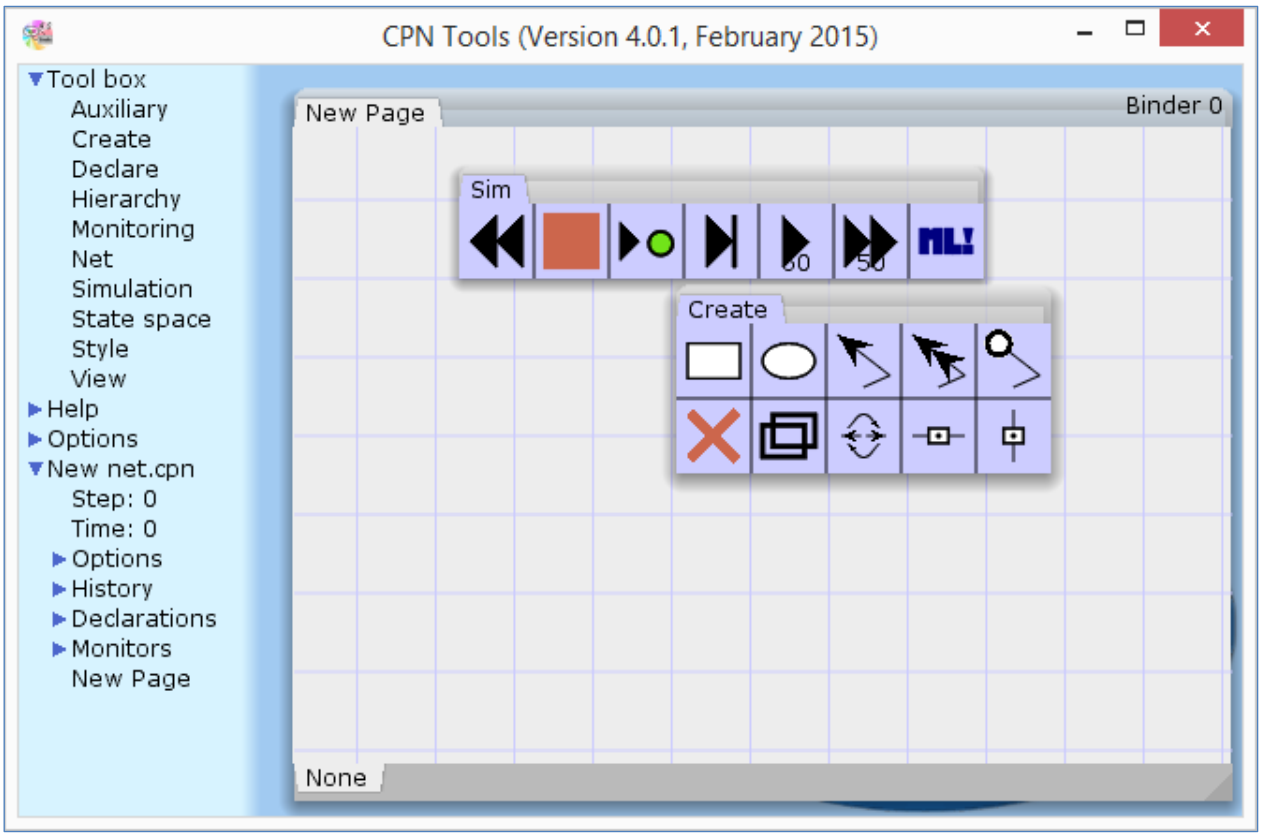


Рис. 4. Зовнішній вигляд програми CPN Tools після додавання нових панелей управління

Інструменти панелі ***Create*** з їх докладним описом представлені в таблиці 1. Інструменти панелі ***Simulation*** з їх докладним описом представлені в таблиці 2.

Для завдання кількості кроків моделювання необхідно натиснути правою кнопкою миші на відповідному інструменті та в контекстному меню вибрати пункт «**Set options**». Для інструменту моделювання з візуалізацією проміжних результатів, аналогічним способом можна задати час між кроками моделювання.

Таблиця 1 - **Інструменти панелі *Create***

Зображення інструменту	Опис інструменту
	*Створення переходу*. Для виконання необхідно натиснути на інструменті, а потім на моделі в тих місцях, де необхідно створити переходи.
	*Створення позиції*. Для виконання необхідно натиснути на інструменті, а потім на моделі в тих місцях, де необхідно створити позиції.
	*Створення дуги*. Для виконання необхідно натиснути на інструменті, а потім на елементах моделі, які потрібно з'єднати. З'єднати можна тільки позицію з переходом. При цьому дуга буде спрямована від першого обраного елемента до другого.
	*Створення вертикальної лінії*. Для виконання необхідно натиснути на інструменті, а потім на моделі. Вертикальна і горизонтальна лінії призначені для організації моделі. Елементи, що знаходяться поблизу лінії, притягуються до неї. За допомогою даного інструменту можна поліпшити сприйняття моделі.
	*Видалення елемента*. Для видалення необхідно натиснути на інструменті, а потім на елементах моделі, які потрібно видалити.
	*Клонування елемента*. Для клонування необхідно обраним інструментом натиснути на елементі, який повинен бути клонований. Після цього курсор набуде вигляду обраного елемента. Потім, натискаючи на моделі, можна кожен раз створювати копію елемента. Клоновані елементи автоматично не перейменовуються. Це необхідно зробити самостійно.
	*Зміна напрямку дуги*. Для виконання необхідно обраним інструментом натиснути на відповідній дузі. Дуга може бути як одностороння, так і двостороння. Двостороння дуга застосовується для економії місця моделі і представляє собою аналог двох дуг, спрямованих в протилежному напрямку і мають однакові вирази.
	*Створення горизонтальної лінії*. Аналогічно створенню вертикальної лінії.

Таблиця 2 - **Інструменти панелі *Simulation***

Зображення інструменту	Опис інструменту
	*Переведення мережі в початковий стан*. При цьому всіх переходах встановлюється їх початкова маркування. Для виконання необхідно обраним інструментом натиснути на моделі.
	*Зупинка виконання моделі*. Для виконання необхідно обраним інструментом натиснути на моделі.
	*Виконання активного переходу із завданням конкретного маркування*. Детальніше даний інструмент розглянуто далі.
	*Виконання одного кроку моделювання,* одного переходу. Для виконання необхідно обраним інструментом натиснути на моделі. При цьому відбудеться спрацьовування одного з активних переходів. При натисненні на активний перехід відбудеться його спрацьовування.
	*Виконання заданого числа кроків з певним часовим інтервалом між ними і візуальним зміною маркування мережі.* Для виконання необхідно обраним інструментом натиснути на моделі. Процес моделювання може бути зупинений.
	*Виконання заданого числа кроків, без демонстрації проміжного маркування*. Для виконання необхідно обраним інструментом натиснути на моделі.
	*Перевірка виділеного тексту на правильність з точки зору синтаксису вбудованої мови програмування CPN ML*. Для виконання необхідно обраним інструментом натиснути на тексті, що містить код CPN ML.

1.3 Побудова простої мережі Петрі в CPN Tools

Розглянемо процес створення моделей в середовищі *CPN Tools*, побудуємо просту модель (рис. 5). Для додавання в мережу позицій і переходів необхідно кликнути лівою клавішею миші на відповідному елементі панелі

Create, а потім натиснути на моделі для додавання елементів мережі. Для зміни імені позиції або переходу необхідно натиснути на елементі лівою клавішею миші і ввести ім'я.

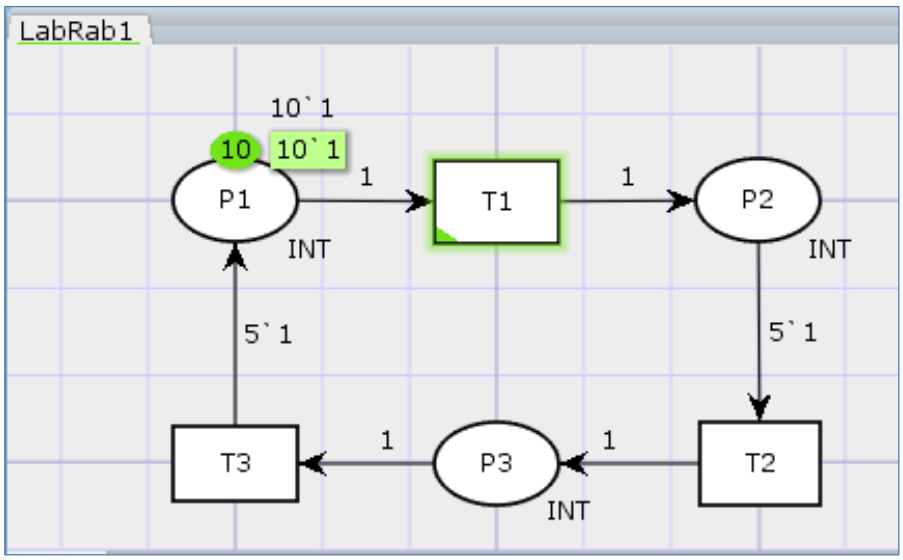


Рис. 5. Приклад простої мережі Петрі в CPN Tools

Для кожної позиції необхідно задати тип та початкове маркування. Завдання імені не обов'язково, проте при визначенні імені необхідно стежити за його унікальністю, не допускається наявність однакових імен позицій і переходів.

Для завдання типу позиції необхідно кликнути лівою клавішею миші на відповідній позиції, натиснути клавішу [Tab] та ввести ім'я типу. Спочатку при створенні моделі доступні чотири типи позицій: *INT, UNIT, BOOL, STRING*. Як приклад введемо тип *INT* (див. рис. 5).

Для завдання початкової маркування необхідно кликнути лівою клавішею миші на відповідній позиції, натиснути клавішу [Tab] два рази та ввести значення початкової маркування. Синтаксис виразу, що визначає початкове маркування, визначається наступним чином: A1++A2++…++An, де елемент Ai визначається, *як num`value*, де *num* – кількість міток, *value* – значення міток, в відповідності з типом позиції.

Зверніть увагу, що кількість міток та їх значення поділяються знаком апостроф (клавіша тильда в англійській розкладці, зліва від одиниці, в лівому верхньому кутку клавіатури). Наприклад, вираз *5`1++10`2* задає початкове маркування позиції, що складається з *15* міток, а саме *5* міток зі значенням *1* та *10* міток зі значенням *2*.

Наступним кроком є визначення виразів на дугах. Вирази на дугах призначені для визначення того, які мітки будуть вилучатись з вхідних

позицій, а які буде додаватися до вихідних позиціях. Вирази на вхідний і вихідний дугах можуть бути різними. Вирази на дугах мають такий же синтаксис, як і вирази для завдання початкової маркування. Наприклад, запис *1`1 ++ 1`2* позначає, що з вхідних позиції, в разі спрацювання переходу буде залучена одна мітка зі значенням 1 і 1 мітка зі значенням 2. Аналогічно для вихідної позиції, при спрацьовуванні переходу в ній буде додана одна мітка зі значенням 1 і 1 мітка зі значенням 2.

Для початкового маркування допускається скорочена форма запису, наприклад запис *1* аналогічно запису *1`1*, а запис *2* аналогічно запису *1`2*. В загальному вигляді *value* аналогічно *1`value*.

1.4 Виконання мережі

Після того як мережа побудована (рис. 5), визначені типи позицій, здійснено початкове маркування та вирази на дугах, можна почати моделювання мережі.

Для початку моделювання необхідно натиснути лівою клавішею мишки на одному з дій вкладки «**Simulation**», а потім, коли курсор набуде вигляду обраної дії, натиснути на моделі. Після цього буде здійснено обрану дію.

Зверніть увагу, що активні переходи підсвічуються зеленим кольором. У разі, якщо не була введена деяка інформація (тип позиції, вираз на дузі), відповідний елемент або декілька елементів будуть підсвічені коричневим кольором. Підсвічування елементів жовтим кольором означає, що в даний момент виконується перевірка правильності введених даних для даного елемента. Така ситуація виникає в момент завантаження моделі, або після зміни типу даних.

Скористаємося інструментом покрокового виконання моделі і простежимо, як змінюється маркування мережі після спрацьовування активних переходів (рис. 6).

1.5 Типи даних та змінні

Модифікуємо розглянуту нами мережу (рис. 5), для цього оголосимо дві змінні *value* і *filter* типу *INT* (рис. 7). Оголошення змінних в даному прикладі буде виглядати наступним чином: *var value,filter:INT;*

Зверніть увагу, що оголошення змінних має слідувати за оголошенням типу даних змінних, в разі змінних *value* і *filter* за типом *INT*. В іншому випадку програма видасть повідомлення про помилку. Після оголошення змінну можна буде використовувати в моделі. Дане правило діє і при оголошенні нових типів міток.

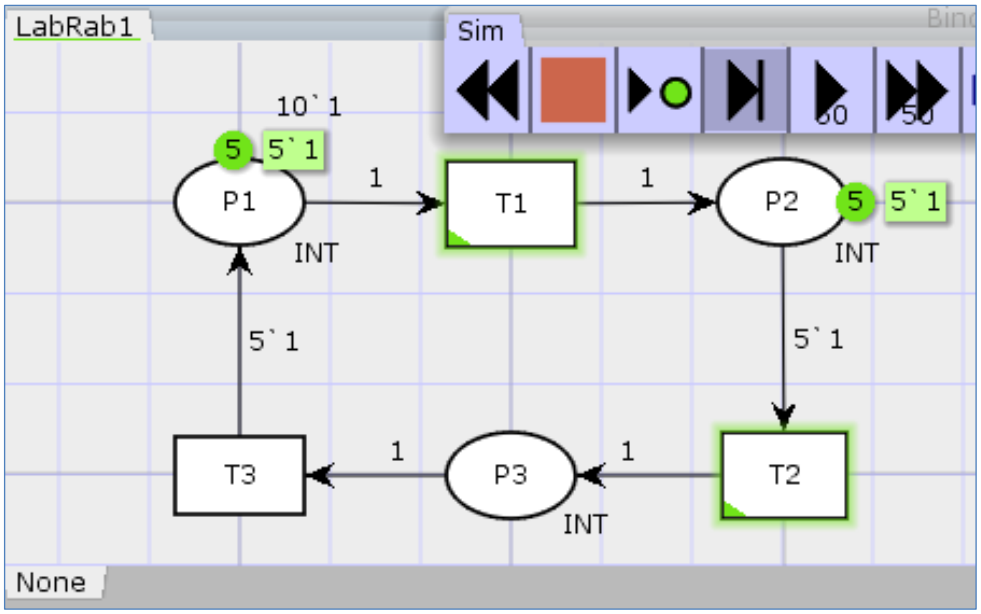


Рис. 6. Приклад виконання простої мережі Петрі в CPN Tools

Після оголошення змінних, додамо нову позицію (*P4*), змінимо початкове маркування позиції *P1*, змінимо вирази на дугах, введемо змінні та поставимо охоронний вираз переходу *T2* (рис. 8).

Зверніть увагу, що в даному прикладі в якості виразів на дугах використовуються змінні (*value* - ім'я змінної), а не конкретні числа. У разі спрацювання переходу *T1* з позиції *P1* буде залучена одна мітка зі значенням 1, 2 або 3. Це значення і буде значенням змінної. Вираз *value* використовується і на вихідний дузі переходу *T1*, це означає, що в позицію *P2* буде поміщена мітка зі значенням *value*, яка була витягнута з позиції *P1*.

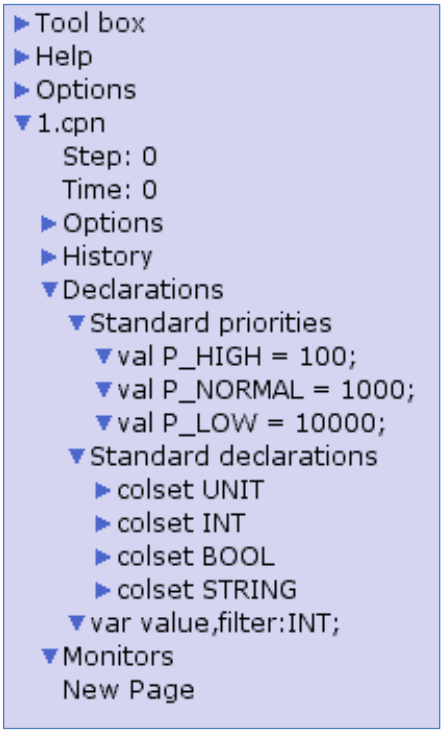


Рис. 7. Об'ява змінних

Якщо ім'я змінної використовується в виразах на вихідних дугах переходу, то така змінна має бути присутня і в вираженні на одній з вхідних дуг. У разі якщо це не буде зроблено, програма повідомить про те, що в вираженні на вихідний дузі використовується невідома змінна.

У разі якщо змінна присутня в виразі на більш ніж однієї вхідний дузі

переходу, програма не повідомить про помилку, однак перехід не буде активований, навіть при наявності міток у вхідних позиціях.

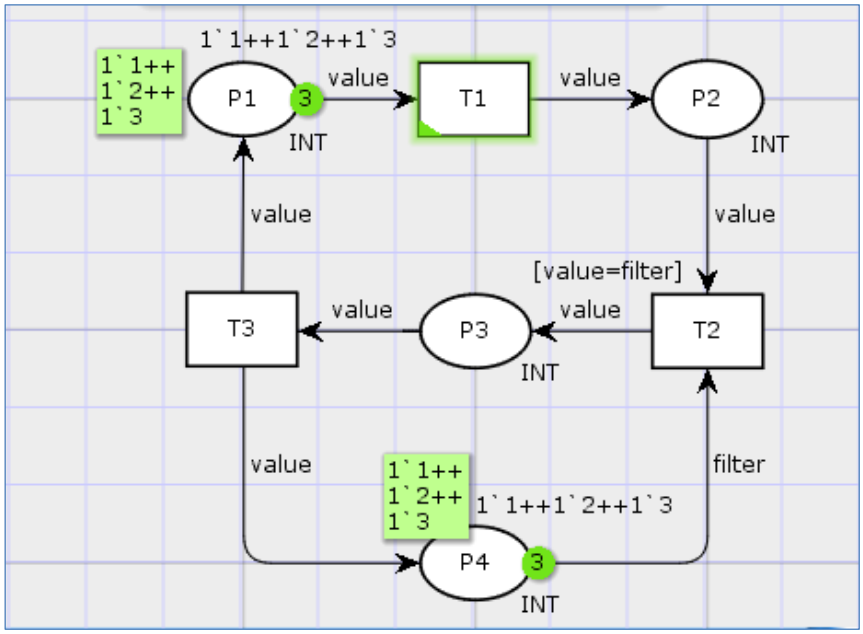


Рис. 8. Приклад мережі в CPN Tools з використанням змінних і охоронних виразів

Для оголошення змінних необхідно виконати таку послідовність дій. У пункті меню вашої моделі (**New net.cpn**, якщо вона ще не була збережена) натиснути лівою клавішею мишки на пункті **Declarations**. У списку, ви побачите два елементи (блоку):

- **Standard priorities**, містить визначення стандартних пріоритетів, *val P_HIGH = 100; val P_NORMAL = 1000; val P_LOW = 10000*;
- **Standard declarations**, містить визначення стандартних типів (кольорів) *colset UNIT = unit; colset INT = int; colset BOOL = bool; colset STRING = string*.

Додаткові змінні та типи даних користувачу краще створювати в окремих блоках. Це дозволить поліпшити сприйняття моделі. Для створення нового блоку правою кнопкою мишки оберіть елемент **Declarations**, а в меню, що з’явилося, оберіть пункт «**New Block**». Введіть ім'я блоку, а потім, натиснувши на ньому лівою клавішею мишки, перетягніть вниз списку. Це необхідно, тому що оголошення змінних має йти після оголошення типів.

Синтаксис оголошення змінних виглядає наступним чином:

var id1, id2, ..., idn : cs_name ,

где *id1 ... idn* – ідентифікатори змінної, *cs_name* – ім'я типу (кольору), що було

оголошеного раніше.

Ідентифікатор змінної може складатися з букв, цифр, знаку підкреслення і апострофа ('). Починатися ідентифікатор повинен з букви.

1.6 Охоронні вирази

Охоронний вираз - це вираз, який визначає умову спрацьовування переходу. Іншими словами, перехід може спрацювати тільки в тому випадку, якщо в його вхідних позиціях знайдуться мітки з такими значеннями, що охоронний вираз прийме значення істини. В охоронному виразі можуть використовуватися змінні з виразів на вхідних дугах переходу.

Для завдання охоронного виразу для переходу необхідно натиснути на ньому мишкою, натиснути клавішу [Tab] (при цьому в лівому верхньому кутку переходу з'явиться текстове поле, доступне для зміни) і ввести охоронний вираз. Синтаксис охоронного виразу має наступний вигляд:

Bool-exp1 Operation Bool-exp2 Operation … Operation Bool-expn,

де *Bool-exp* – логічний вираз виду *змінна оператор змінна*, оператор: дорівнює (=), більше (>), менше (<), не дорівнює (<>); *Operation* – логічна зв’язка , та – *andalso,* або – *orelse*.

В даному прикладі введемо охоронний вираз *filter = value* для переходу *T2*, це означає, що перехід *T2* може спрацювати в тому випадку, якщо в позиціях *P2* і *P4* знайдуться мітки з однаковими значеннями (див. рис. 8). При спрацьовуванні переходу ці мітки вилучаються з відповідних вхідних позицій.

1.7 Налагодження моделей

При побудові великих моделей важливим етапом є налагодження і перевірка правильності функціонування. *CPN Tools* має кілька інструментів для налагодження моделей. По-перше, може бути використаний інструмент покрокового виконання на вкладки «**Simulation**». При покроковому моделюванні відбувається спрацьовування одного з активних переходів та зміна маркування.

Іншим інструментом налагодження моделей є виконання активних переходів із заданим маркуванням. Для цього необхідно вибрати відповідну дію на вкладці «**Simulation**», натиснути на один з активних переходів і обрати доступні значення для змінних на вхідних дугах переходу, як показано на рисунку 9. У даному прикладі активним переходом є перехід *T2* з двома вхідними змінними *i, j*. Для змінної *i* доступні значення 1 і 2, а для змінної *j* 3 і 4.

Після вибору значень вхідних змінних вони будуть показані в прямокутнику під переходом. При повторному натисканні на перехід він буде активований.

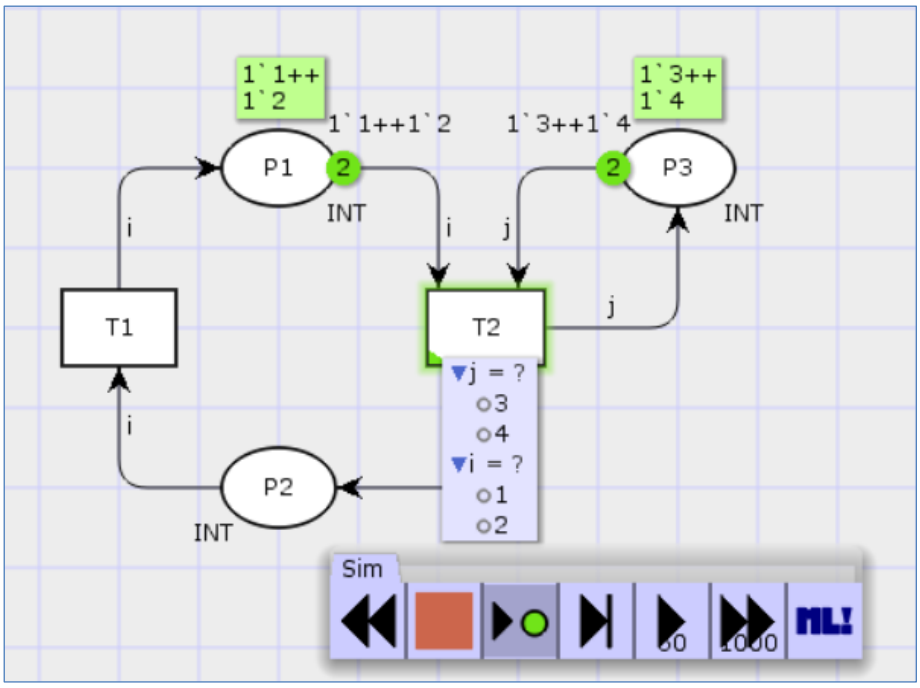


Рис. 9. Виконання переходу із заданим маркуванням

Для налагодження моделей також може бути використано виконання декількох кроків моделювання з відображенням проміжних маркувань. Такий режим виконання може бути зручний при тестуванні великої кількості кроків.

Завдання для практичної роботи №1

1. Побудуйте модель (рис. 7), встановіть маркування позицій *P1, P4* відповідно до варіанта завдання (таблиця 3). Наприклад, варіанту (*n1, n2, n3*) (*v1, v2, v3*) буде відповідати маркування *n1`v1 ++ n2`v2 ++ n3`v3*.

2. Побудувати мережу, наведену на рисунку 10. Мережа повинна здійснити генерацію *N* міток (всього) типу *INT* зі значеннями *v1, v2, v3* відповідно. Після генерації мітки повинні бути відсортовані відповідно до значень в позиціях *P4, P5, P6*. Варіанти завдання наведені в таблиці 4, у вигляді *N, v1, v2, v3*.

3. Модифікувати мережу із завдання 2 таким чином, щоб можна було генерувати мітки з заздалегідь відомим відсотковим розподілом за значеннями.

Зміст звіту по практичній роботі №1

Мета роботи, завдання відповідно до варіанту.

Зміст звіту за завданням 1

1. Скріншот вікна програми *CPN Tools* з побудованою моделлю, на скріншоті крім самої моделі має бути оголошення змінних.
2. Маркування всіх позицій після кожного кроку моделювання у вигляді таблиці з трьома графами 1 - номер кроку і ім'я переходу, що спрацював, 2 - назва позиції, 3 - маркування позиції. У таблицю внести дані по п'яти крокам моделювання.

Таблиця 3 - **Маркування позицій *P1, P4* відповідно до номеру варіанта**

Варіант	Маркування	Варіант	Маркування
1	(2,4,5) (2,7,6)	11	(6,9,2) (5,6,3)
2	(5,2,4) (4,5,2)	12	(8,8,4) (5,7,2)
3	(7,3,1) (5,2,6)	13	(4,1,8) (6,8,3)
4	(3,8,1) (2,1,4)	14	(9,3,5) (9,4,2)
5	(3,2,7) (9,1,5)	15	(5,2,1) (8,2,4)
6	(7,1,4) (5,4,5)	16	(8,1,2) (6,3,7)
7	(2,5,8) (9,5,8)	17	(2,6,7) (9,4,7)
8	(3,6,5) (6,2,4)	18	(4,6,8) (7,2,5)
9	(5,9,2) (3,5,1)	19	(5,8,3) (4,5,2)
10	(7,2,5) (6,3,5)	20	(9,1,4) (6,2,7)

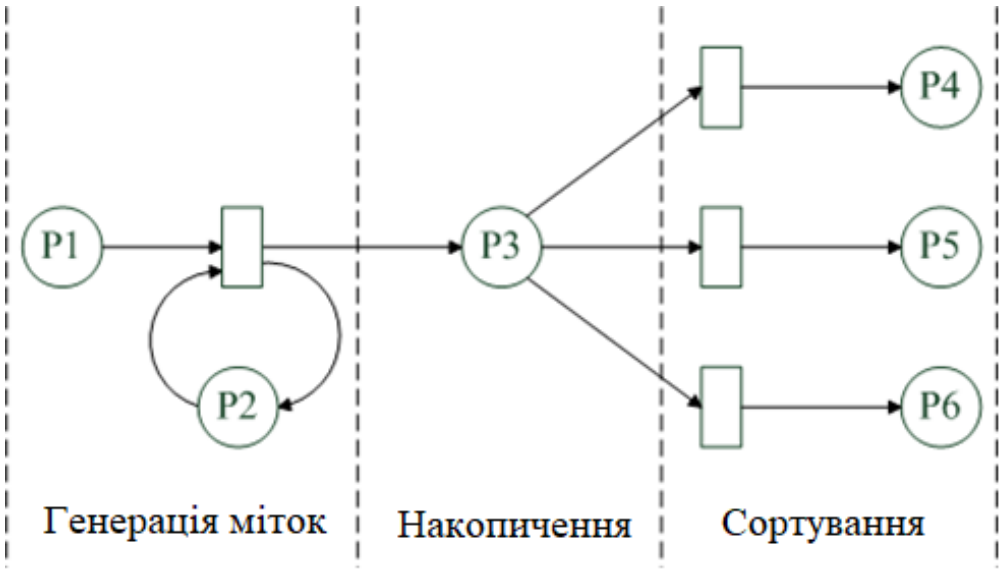


Рис. 10. Схема мережі для завдання № 2

Таблиця 4 - **Кількість і значення міток відповідно до номеру варіанта**

Варіант	Завдання	Варіант	Завдання
1	1500, 3, 4, 1	11	1400, 8, 4, 3
2	1250, 5, 7, 2	12	1550, 1, 2, 3
3	1300, 6, 2, 8	13	1700, 5, 8, 2
4	1550, 8, 6, 9	14	1650, 9, 8, 7
5	1700, 2, 4, 5	15	1200, 5, 3, 1
6	1450, 5, 6, 7	16	1800, 7, 5, 2
7	1400, 6, 9, 8	17	1750, 8, 2, 4
8	1350, 2, 5, 6	18	1850, 9, 1, 7
9	1650, 6, 4, 7	19	1500, 6, 5, 2
10	1600, 7, 5, 8	20	1500, 1, 7, 9

Зміст звіту за завданням 2 та 3

1. Скріншот вікна програми *CPN Tools* з побудованою моделлю, на скріншоті крім самої моделі має бути оголошення змінних.
2. Модель необхідно виконати не менше п'яти разів. За результатами кожного процесу моделювання необхідно привести маркування позицій *P4, P5, P6*, переконатися, що мітки відсортовані правильно. Для *P4, P5, P6* привести кількість міток в кожній позиції та їх сумарну кількість. Переконатися в тому, що сумарна кількість міток дорівнює кількості *N* завдання. Розрахувати відсотковий розподіл міток по позиціях *P4, P5, P6*.
3. Порахувати загальну кількість міток в позиціях *P4, P5, P6* по всіх експериментах, розрахувати відсоткове співвідношення кількості міток.

В кінці звіту по практичній роботі повинні бути сформульовані *висновки*, що відображають зміст виконаної роботи та отримані результати.

Контрольні питання. Блок №1

1. Що таке охоронний вираз переходу?
2. Яким чином здійснюється завдання охоронного виразу переходу?
3. Що означає вираз на дузі?
4. Яким чином здійснюється оголошення змінних?
5. Як використовують змінні у виразах на вхідних і вихідних дугах переходу?
6. Які типи даних (кольору) існують в *CPN Tools* за замовчуванням?

7. Що таке пріоритет спрацьовування переходу, яким чином він змінює процедуру спрацьовування?

2. Практична робота №2. Вивчення можливостей *CPN Tools*, вбудовані функції, масиви

У даній практичній роботі розглядаються можливості CPN Tools по роботи зі списками, приклад оголошення типу списку, приклад моделі з додаванням і видаленням елементів з черги. Наводиться перелік вбудованих функцій CPN ML для роботи зі списками з їх коротким описом. Розглядається приклад роботи зі складними структурами даних, доступ до полів структури і вирази на вихідних дугах для генерації міток типу record.

2.1 Робота з списками

При розробці моделі реального об'єкту, пристрою або інформаційної системи може знадобитися можливість збереження запитів користувача або пакетів даних в черзі, поки вони будуть чекати звільнення обробного пристрою. Для організації черг в *CPN Tools* передбачений спеціальний тип даних:

colset LIST_NAME = list EL_CS_NAME [with INT-EXP1..INT-EXP2],

де *LIST_NAME* – ім'я нового типу даних (списку елементів заданого типу), *EL_CS_NAME* – тип елементів, з яких буде складатися список, *INT-EXP1* – цілочисельний вираз, що задає мінімальну кількість елементів у списку, *INT-EXP2* – цілочисельний вираз, що задає максимальну кількість елементів в списку. Частина, що укладена в квадратні дужки, не є обов'язковою. Тип списку, що складається з логічних змінних, може бути оголошений таким чином:

colset BoolList = list BOOL.

Щоб задати вміст списку, необхідно перерахувати елементи списку в квадратних дужках [*element_1, element_2, ..., element_n*], для завдання порожнього списку можна записати квадратні дужки без елементів []. *CPN Tools* підтримує ряд функцій для роботи зі списками. У таблиці 5 наведені деякі з цих функцій.

Таблиця 5 - **Функції *CPN Tools* для роботи зі списками**

Функція зі списком параметрів	Опис функції
nil	Завдання порожнього списку, аналогічно запису []. Даний запис може бути використано для завдання початкових маркувань позицій відповідного типу.
e::l	Видалення першого (головного) елемента списку *l* і запис його в змінну *e.*
hd l	Отримання першого елемента списку *l.*
tl l	Отримання списку *l* після видалення головного елемента.
length l	Отримання довжини списку *l.*
rev l	Запис списку *l* в зворотному порядку.
map f l	Застосування функції *f* до елементів списку *l*. Результатом буде список, заповнений повернутими значеннями функцій *f.*
List.nth(l,n)	Отримання елемента списку *l* з індексом *n*, при цьому *0 <= n < length l.*
List.take(l,n)	Отримання списку з *n* перших елементів списку *l.*
List.drop(l,n)	Отримання списку, після відкидання *n* перших елементів списку *l.*
List.null l	Перевірка списку на наявність елементів. Повертає *true*, якщо контактів немає, в іншому випадку повертає *false.*
Додаткові функції для роботи зі списками	
l1^^l2	Зчеплення списку *l1* і *l2*. результуючий список містить послідовно елементи *l1* и *l2.*
mem l x	Перевірка наявності елемента *x* в списку *l*. Повертає *true*, якщо елемент *x* присутній в списку *l*, в іншому випадку повертає *false.*
remdupl l	Видалення зі списку *l* елементів, що повторюються.
rm x l	Видалення першого елемента *x* зі списку *l.*
rmall x l	Видалення всіх елементів *x* зі списку *l.*
intersect l1 l2	Отримання списку, що містить елементи, що входять до перетинання списку *l1* та *l2.*
ins l x	Вставка елемента *x* в кінець списку *l.*
ins_new l x	Вставка елемента *x* в кінець списку *l* за умови, що список *l* не містить *x*
sort lt_fun l	Сортування списку *l* за допомогою функції *lt_fun,* яка використовується для визначення найменшого з двох елементів. Для всіх типів може бути використана стандартна функція *cs.lt*, де *cs* це ім'я типу, наприклад *INT.lt.*

Приклад організації черги наведено на рисунку 11. У даному прикладі введений новий тип для подання списку цілих чисел *INT_LIST* та оголошені дві змінні.

colset INT_LIST = list INT;

var i:INT;

var il:INT_LIST;

P1 – позиція, яка містить початковий набір міток *1`1++1`2++1`3++1`4*;

P2 – позиція, що представляє чергу, початковим маркуванням даної позиції є порожній список []; *T2* - перехід, який здійснює додавання елемента *i* в кінець черги *il*, *il^^[i]*; *T1* – перехід, який здійснює вилучення з черги головного елемента *i::il*.

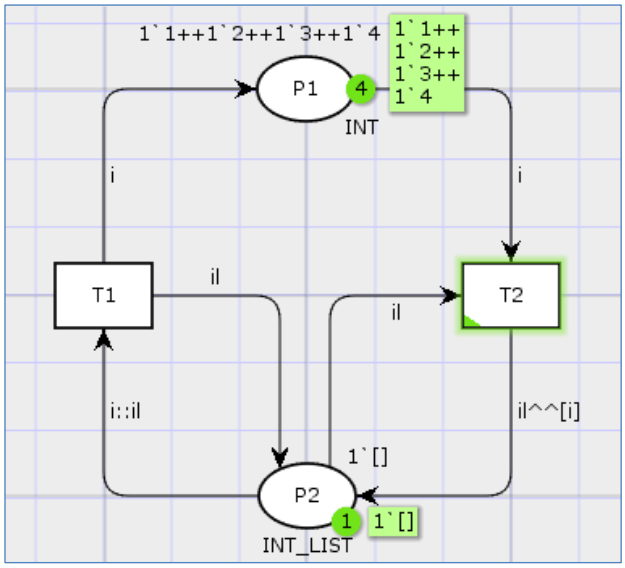


Рис. 11. Реалізація черги в CPN Tools

На рисунку 12 представлений приклад використання функцій для роботи зі списками. Функції для роботи зі списками задані на вихідних дугах переходів *IsNull*, *Reverse* і *Sort*. На вхідних дугах даних переходів задана змінна *il* типу *INT_LIST*. *IsNull* – перевіряє відсутність елементів в списку *il*, *Reverse* – інвертує елементи в списку *il*, *Sort* – сортує елементи списку *il* по зростанню.

При роботі зі списками необхідно звертати увагу на значення, що повертає функція. Воно повинно відповідати типу позиції, для якої дуга з виразом є вхідної.

.

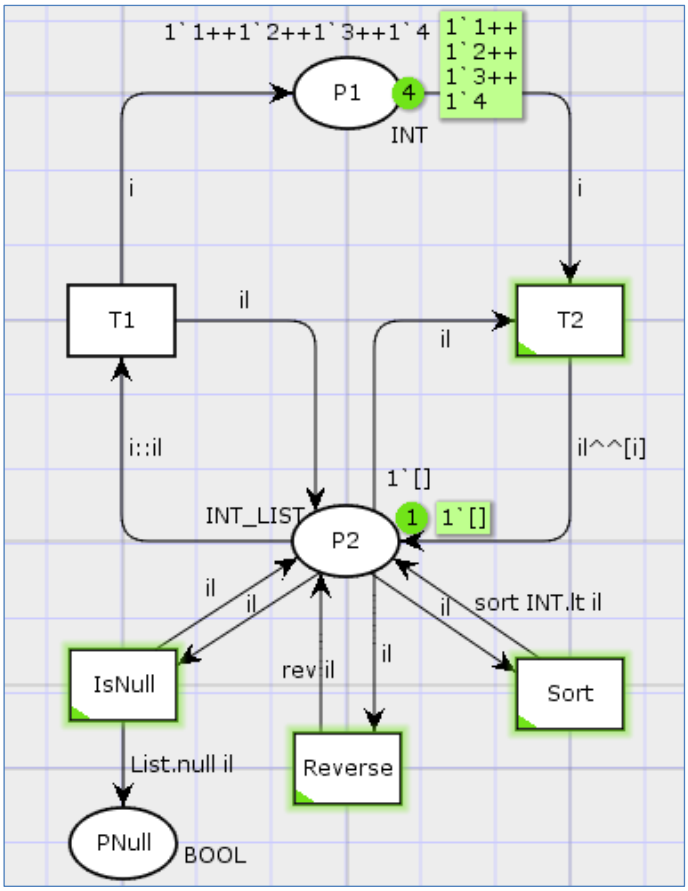


Рис. 12. Застосування функцій для роботи зі списками.

2.2 Тип даних record

При побудові моделі з обробкою запитів може виникнути необхідність подання запиту не однією міткою типу *INT*, а деякою більш складною структурою, що містить множину полів різних типів. Для подання таких структур в *CPN Tools* передбачений спеціальний тип даних *record*. Оголошення такого типу виглядає наступним чином:

*colset CS_NAME = record ID1:CS1 * ID2:CS2 * ... * IDN:CSN,*

де *CS_NAME* – ім'я нового типу, *ID1, ID2, ..., IDN* – імена полів структури, *CS1, CS2, ..., CSN* – типи відповідних полів структури. Таким чином оголошення структури, що містить три поля типу *INT*, буде виглядати наступним чином:

*colset INT_REC=record i1:INT*i2:INT*i3:INT.*

Для завдання міток типу *record* може бути використана наступна форма запису:

{ID1=VALUE1, ID2=VALUE2, ..., IDN=VALUEN},

де *ID1, ID2, ... , IDN* – імена полів структури, *VALUE1, VALUE2, ... , VALUEN* – значення відповідних полів структури.

Для отримання доступу до окремих полів структури необхідно використовувати такий запис: *#idi rec*, де *idi* – ім'я поля структури, *rec* – ім'я змінної типу *record*.

2.3 Використання типу даних record при моделюванні

На рисунку 13 показаний приклад роботи зі структурами в *CPN Tools*. В даному прикладі здійснюється генерація міток типу *record*, додавання міток в чергу, витяг міток з черги та доступ до полів структури. Для цього прикладу були оголошені два типи даних *INT_REC* – структура з трьох цілих чисел *і INT_REC_LIST* – список з елементів типу *INT_REC*.

*colset INT_REC = record i1:INT*i2:INT*i3:INT;*
colset INT_REC_LIST = list INT_REC;
var irl:INT_REC_LIST;
var v1,v2,v3:INT;
var ir1:INT_REC;

Позиції *P1, P2, P3* містять мітки цілого типу, які будуть використані для генерації міток типу *INT_REC*. Перехід *Gen* здійснює генерацію міток типу *INT_REC* на основі вхідних змінних *v1, v2, v3*. Для цього використовується

вираз *1` {i1 = v1, i2 = v2, i3 = v3}* на вихідний дузі переходу *Gen*. *P4* – позиція, що здійснює накопичення міток типу *INT_REC*. *Put* – перехід, який здійснює додавання міток в кінець черги. *Queue* – позиція, що моделює чергу з елементів *INT_REC*. *Get* – перехід, який здійснює: вилучення головного елемента *ir1* зі списку *irl*, додавання однієї мітки в позицію *Num*, додавання по одній мітці в позиції *O1, O2, O3* зі значеннями полів *i1, i2, i3* структури *ir1* відповідно. *O1, O2, O3* – позиції, які здійснюють накопичення значень полів *i1, i2, i3* відповідно, за результатами моделювання можна буде оцінити, який із значень кожне поле приймало найчастіше. *Num* – позиція, що обмежує максимальну кількість одночасно існуючих міток в моделі, що були згенеровані.

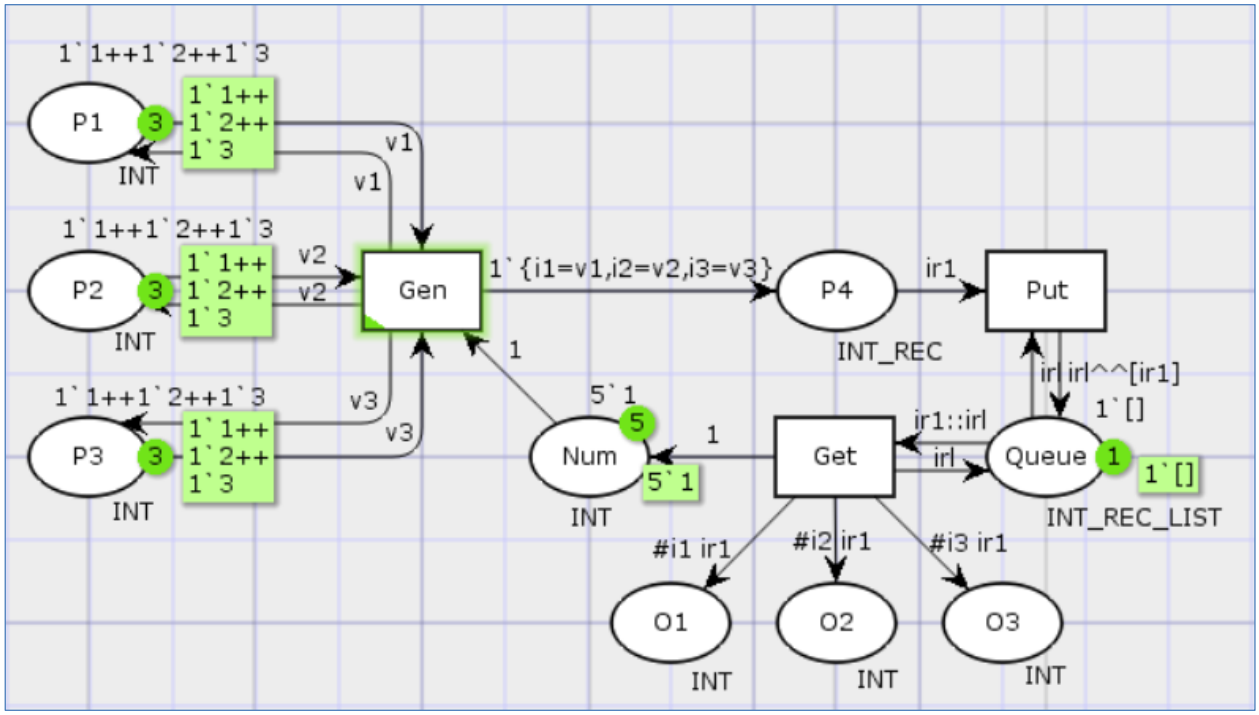


Рис. 13. Робота зі структурами

Тип *record* може використовуватися для подання інформації про запити користувача (наприклад ім'я, номер запиту, пріоритет, код запиту і ін.), змісту полів пакету даних (наприклад адреса джерела та адресу призначення, розмір пакета, поле даних, пріоритет і ін.). Або для подання інших складних типів даних в моделях.

Завдання для практичної роботи №2

1. Взяти за основу приклад з чергою (рисунок 11), модифікувати його таким чином, щоб додавання і витяг міток відповідало структурі даних *стек*. У звіті представити скріншот моделі та коментарі щодо внесених змін.

2. Розробити модель роботи відділу обслуговування клієнтів в банку.

Відділ обслуговує фізичних та юридичних осіб, в банк можуть звертатися клієнти з різними запитами: відкриття рахунку, закриття рахунку, взяття кредиту, внесення грошей на рахунок, зняття грошей з рахунку. У трьох останніх випадках клієнт здійснює операцію з деякою сумою грошей. Таким чином, може бути сформована структура даних для представлення клієнта, що складається з трьох полів цілого типу:

1 – тип клієнта,

2 – тип операції,

3 – сума грошей для операції (якщо операція це передбачає, 0 в іншому випадку).

У відділі передбачено дві черги для фізичних та юридичних осіб. Клієнти з кожної черги обслуговуються окремим оператором. Кожен оператор підраховує кількість обслугованих клієнтів, кількість запитів різного типу та суму грошей по всіх клієнтах, що обслуговані.

Один з можливих запитів в даному відділі не обслуговується, відповідні клієнти направляються в інший відділ. Це може бути змодельоване простим відкиданням міток. Передбачити обмеження кількості клієнтів, одночасно очікують запиту *QMax*. Передбачувана структура моделі представлена на рисунку 14.

Змоделювати роботу відділу з обслуговування 1000 клієнтів. У звіт повинна бути наступна статистика: кількість кроків моделювання, кількість обслужених клієнтів по типу, кількість запитів кожного типу по кожному типу клієнтів, кількість клієнтів яким було відмовлено в обслуговуванні, сума грошей по кожному типу клієнтів.

У звіті представити скріншот моделі оголошених типів даних і змінних, використаних в моделі.

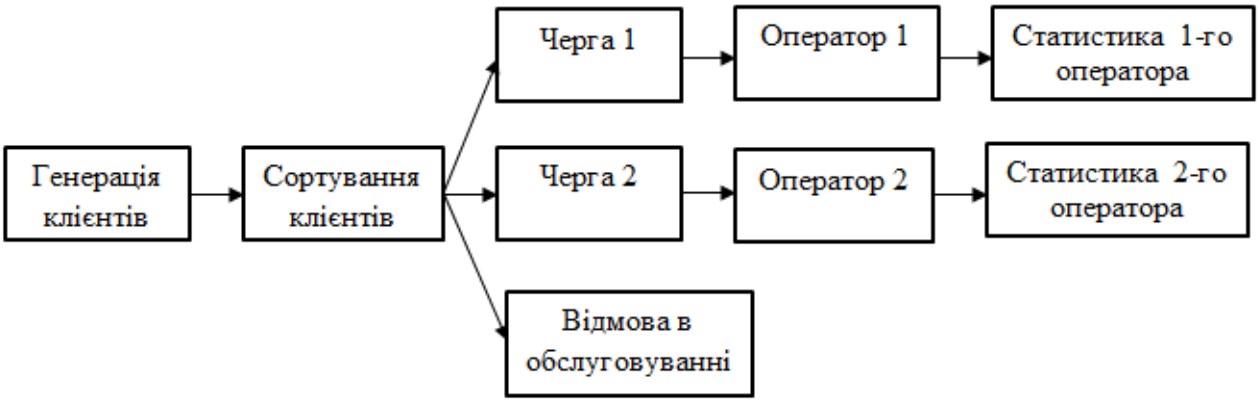


Рис. 14. Структура моделі

Зміст звіту по практичній роботі №2

Звіт повинен містити номер варіанта і завдання, звіт за кожним завданням, висновок, що відображає зміст і результати виконаної роботи, відповіді на контрольні питання.

Контрольні питання. Блок №2

1. Як виглядає визначення списку елементів деякого типу?
2. Які функції передбачені для додавання та вилучення елементів зі списку?
3. Як виглядає визначення типу структури, що містить кілька полів деякого типу?
4. Як здійснюється доступ до полів структури?
5. Яким чином здійснюється генерація міток типу *record*?

3. Практична робота №3. Час в *CPN Tools*

У даній практичній роботі розглядаються можливості програми CPN Tools з моделювання процесів з урахуванням часу. Розглядаються способи завдання часових значень при моделюванні, а також функції для отримання поточного модельного часу. Розглядається зв'язок модельного та реального часу, наводяться приклади моделей з використанням часових міток. Вивчаються функції генерації випадкових величин з різними законами розподілу та застосування цих функцій при моделюванні затримок виконання. Описується спосіб і розглядається приклад зберігання статистики про результати моделювання в позиціях мережі.

3.1 Час в CPN Tools

При побудові моделей реальних пристроїв потрібен розгляд динаміки процесу і визначення часових характеристик роботи моделі. Реальні процеси розвиваються в часі, тому і математичний апарат, призначений для їх моделювання, повинен мати можливість подання подій у часі. Тому моделі, що описує тільки внутрішню структуру та логіку, часто буває недостатньо.

Для моделювання процесів у часі, мережі Петрі розширюються введенням часових міток в маркери. Часова мітка показує, з якого моменту часу маркер буде доступний в позиції. При визначенні активності переходів в часової мережі Петрі враховуються тільки ті маркери, у яких значення часу менше або дорівнює поточному часу моделі.

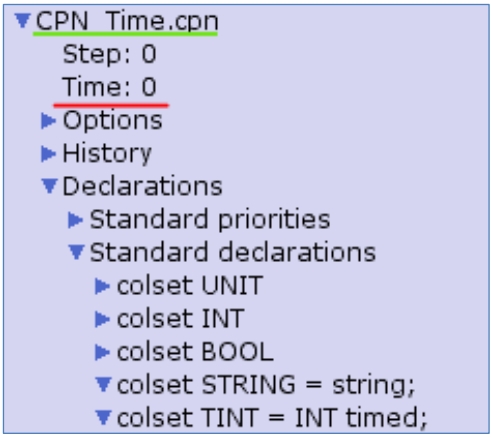


Рис. 15. Поточний модельний час в інтерфейсі CPN Tools

Час в моделях *CPN Tools* є безрозмірною величиною і представляється у вигляді невід'ємного цілого числа. Поточне значення модельного часу розташовується у вкладці моделі (рис. 15), поруч із значенням кроку моделювання. Якщо моделювання ведеться без використання часових міток, то значення поточного модельного часу не змінюватиметься при виконанні моделі.

У разі, якщо на поточному кроці моделювання немає активних переходів з часом, меншим або рівним поточному, модельне час збільшується до тих пір, поки в мережі не з'явиться активний перехід.

3.2 Зв'язок модельного і реального часу

Так як в *CPN Tools* час представляється у вигляді цілого числа, а в реальності воно безперервно, необхідно встановити взаємозв'язок між значеннями модельного часу та їх реальними еквівалентами. Для цього може бути використано два підходи:

1. Деякому інтервалу реального часу (наприклад, *1* секунда) обирається кількість тактів модельного часу (наприклад, *10*). Відповідно до цього вибору встановлюються затримки для всіх дій в мережі. Якщо деяка дія триває *25* секунд, то в моделі вона триватиме *25 * 10 = 250* тактів модельного часу. Аналогічно, якщо відомо кількість кроків модельного часу, що зайняв певний процес, можна обчислити його тривалість в секундах. Наприклад, *600* тактів модельного часу дорівнює *600/10 = 60* секунд.

2. Вибирається найменш тривалий процес, який необхідно моделювати, наприклад мінімальний час обслуговування найпростішого запиту користувача до інформаційної системи, дорівнює *1* мс. Цьому часу будуть відповідати *10* кроків модельного часу. Відповідно до цього вибором встановлюються інші затримки в моделі.

При виборі модельного часу слід обирати відповідність з запасом. Якщо розробляється модель з розрахунком *1* такт дорівнює *1* секунді, то неможливо змоделювати інтервал часу менше *1* секунди або трохи більше, наприклад *1.1*

секунда. Так як в житті тривалість кожної події може дещо відрізнятися в різних експериментах, то і в моделі слід це передбачити. Для цього можна побудувати ту ж модель з розрахунком *10* тактів на одну секунду реального часу.

3.3 Використання часових міток в моделях

Перед тим як вводити часові затримки в модель мережі Петрі в *CPN Tools*, необхідно визначити нові типи даних для часових міток. Формат часових міток виглядає наступним чином:

colset NEW_TYPE = TYPE timed,

де *colset* – ключове слово для визначення типу, *timed* - ключове слово, що показує, що маркери даного типу мають часові мітки, *NEW_TYPE* – назва нового типу, *TYPE* – назва типу, який був визначений раніше, без додавання часової мітки.

Як приклад розглянемо тип, який визначає множину цілих чисел з часовими мітками. Даний тип може бути заданий наступним чином:

colset TINT = INT timed.

Значення часу в мітках може бути змінено за допомогою завдання виразів на дугах. Для модифікації часової мітки застосовується наступний формат вираження на дузі:

expr @ + k,

де *expr* – вираз на дузі без урахування часу, *@ + k* – запис означає установку часу в мітці, як поточне + *k*. Приклад завдання часових міток наведено на рисунку 16.

Часові мітки можуть бути задані при спрацьовуванні переходу. Для цього необхідно модифікувати перехід в такий спосіб: виділити перехід, натиснути клавішу *[Tab]* двічі та ввести значення виду *@ + k.* Приклад, мережі Петрі із завданням часових міток при спрацьовуванні переходу наведено на рисунку 17.

Часова мітка буде додана до всіх маркерів, які будуть спрямовані в позиція *P2*, при спрацьовуванні переходу *T1*. Однак на роботу самого переходу *T1* така модифікація не вплине.

Для завдання затримок спрацьовування переходу модифікуємо мережу Петрі на рисунку 17. Для цього додамо одну позицію та поставимо вирази на дугах (рисунок 18).

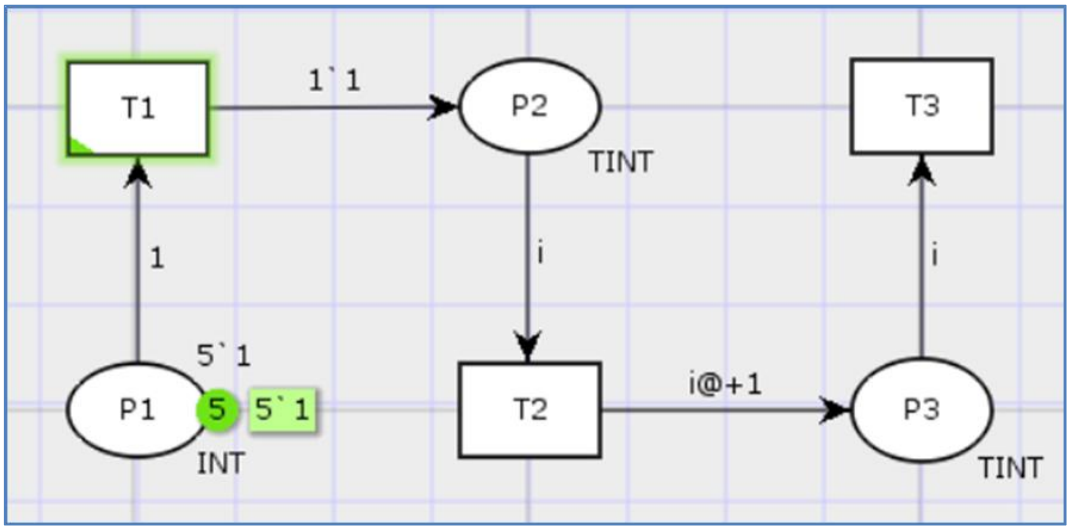


Рис. 16. Приклад мережі з часовими позиціями та модифікацією часових міток в виразах на дугах

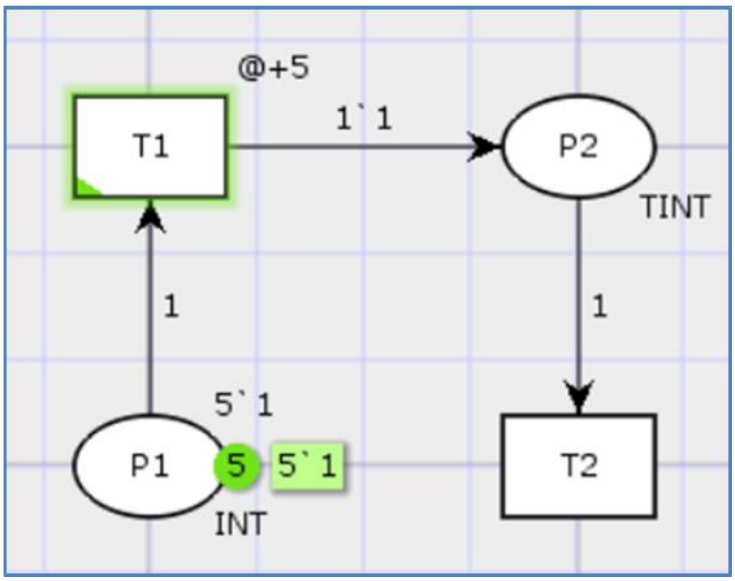


Рис. 17. Завдання часових міток при спрацьовуванні переходів

Після модифікації мережі (див. рис. 18) перехід *T1* буде спрацьовувати кожні *5* тактів модельного часу. Так як одна з його вхідних позицій містить маркери з часовими мітками. Мітки стають доступними через *5* тактів модельного часу після спрацьовування переходу *T1*. Використовуючи такий підхід можливо задавати час роботи деяких ділянок моделі, наприклад час обробки пакетів даних або час обслуговування клієнтів.

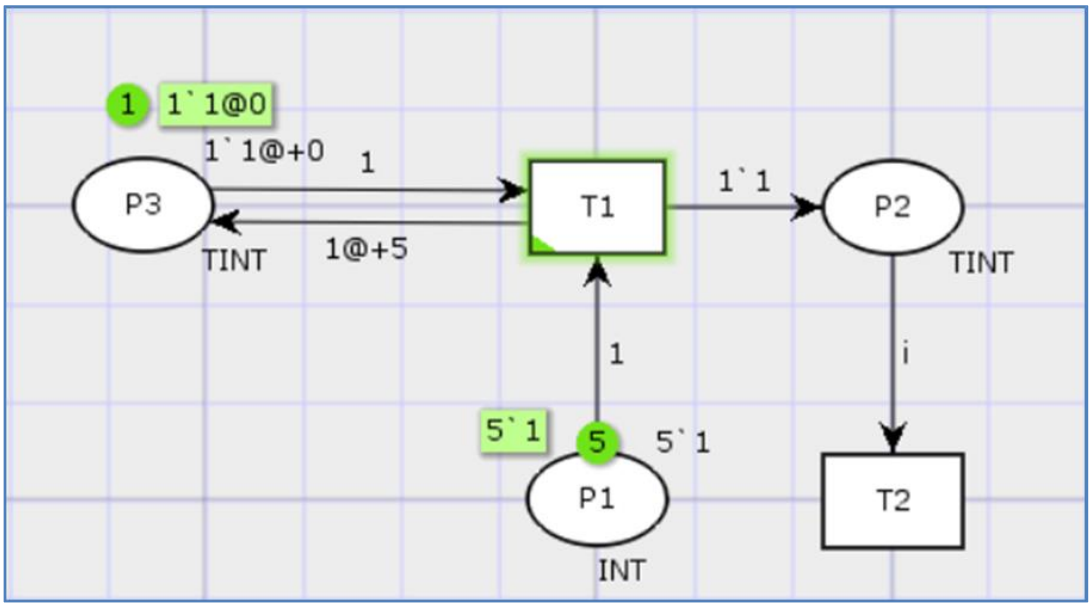
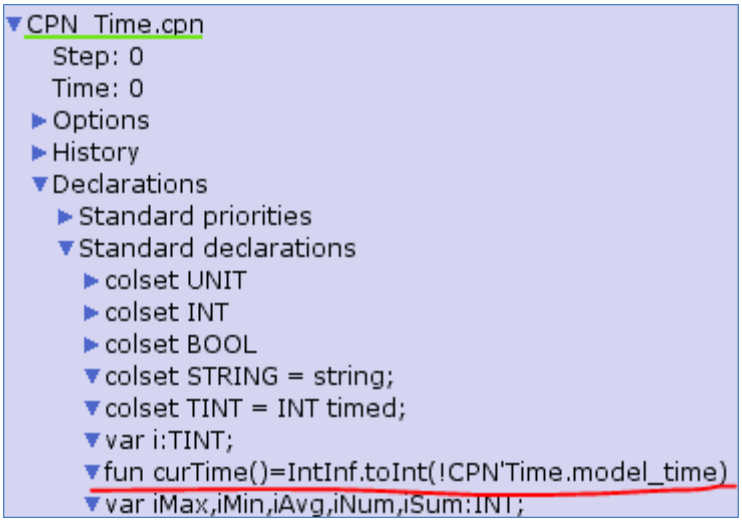


Рис. 18. Завдання затримки спрацьовування переходу

3.4 Здобуття поточного модельного часу



При побудові моделей часто виникає задача здобуття поточного значення модельного часу. Для цього необхідно скористатися наступною функцією:

fun curTime()=IntInf.toInt(!CPN'Time.model_time)

Рис. 19. Об'ява та реалізація функції curTime() в CPN Tools

Текст даної функції вводиться в блок разом з оголошеннями типів та змінних (рисунок 19).

Приклад використання функції *curTime ()* для генерації маркерів з поточним значенням часу представлений на рисунку 20. Зверніть увагу на вираз на вихідній дузі переходу *T1*, в даному прикладі значення часу стає значенням маркера, а не часовою мітки. При обробці таких міток це значення може бути використано для розрахунку часу руху мітки по мережі.

Для роботи з часом в *CPN Tools* є ще кілька функцій. Зокрема альтернативним способом отримання поточного часу може бути виклик функції *time ()*, проте результатом даної функції є число в спеціальному

форматі *intinf*, призначеному для зберігання великих чисел. Для перетворення цього значення в тип *INT* може бути використана функція *IntInf.toInt (time ())*, або функція *ModelTime.toString (time ())* – для перетворення в тип *STRING*.

Для отримання значення поточного кроку моделювання передбачена функція *step ()*. Вона також повертає значення типу *intinf*. Приклад використання функцій *time ()* і *step ()* представлений на рисунку 21.

T1 – перехід, який здійснює генерацію міток зі значенням поточного часу і кроку моделювання в форматах *INT* і *STRING*. *Delay* – позиція, призначена для зберігання маркерів з часовими мітками. *IStep, SStep, ITime, STime* – позиції, призначені для зберігання міток зі значеннями кроку і часу моделювання в форматах *INT* та *STRING* відповідно.

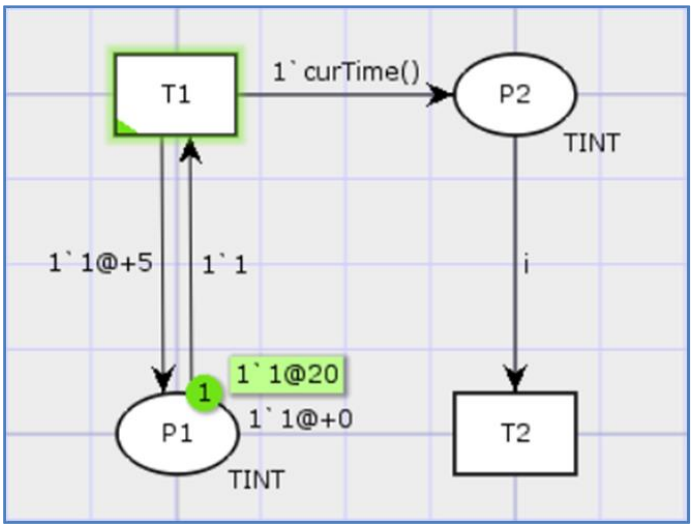


Рис. 20. Використання функції curTime()

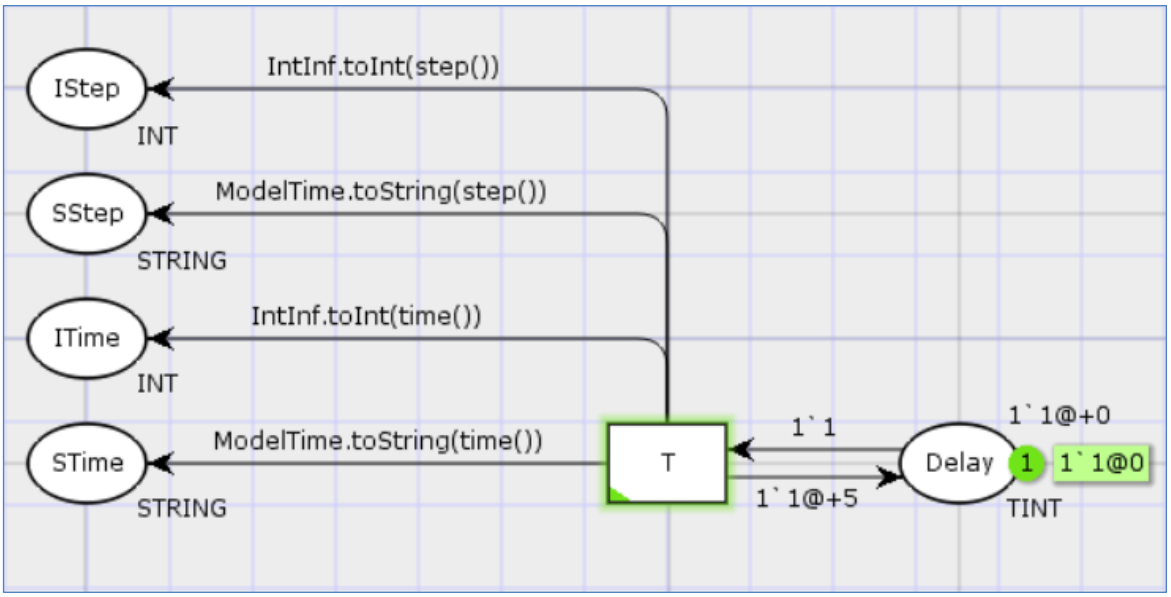


Рис. 21. Приклад використання функцій time() і step()

3.5 Генерація випадкових величин

При побудові моделі реальної системи часові інтервали не завжди можуть бути представлені у вигляді певного числа, замість цього в моделях час майже завжди задається в вигляді випадкової величини, що розподілена по деякому закону. Для завдання випадкових величин можуть бути використані наступні функції:

uniform (real min, real max),

де *max* і *min* – максимальне та мінімальне значення випадкової величини.

normal (real M, real D),

де *M* і *D* – математичне сподівання та дисперсія випадкової величини. Математичне сподівання характеризує середнє значення випадкової величини, а дисперсія можливе відхилення конкретного значення від математичного очікування.

В допомозі до програми *CPN Tools* можна знайти ще декілька функцій для завдання випадкових величин.

При завданні часових затримок можуть бути використані тільки цілі числа, а функції *uniform* і *normal* повертають дійсні. Для перетворення результатів роботи функції до прийнятного виду скористаємося функціями *floor (real v)* та *ceil (real v).* Функція *floor* округлює число до найближчого цілого в меншу сторону, а функція *ceil* – в більшу.

Приклад використання функцій генерації випадкових чисел і функції *ceil* представлений на рисунку 22.

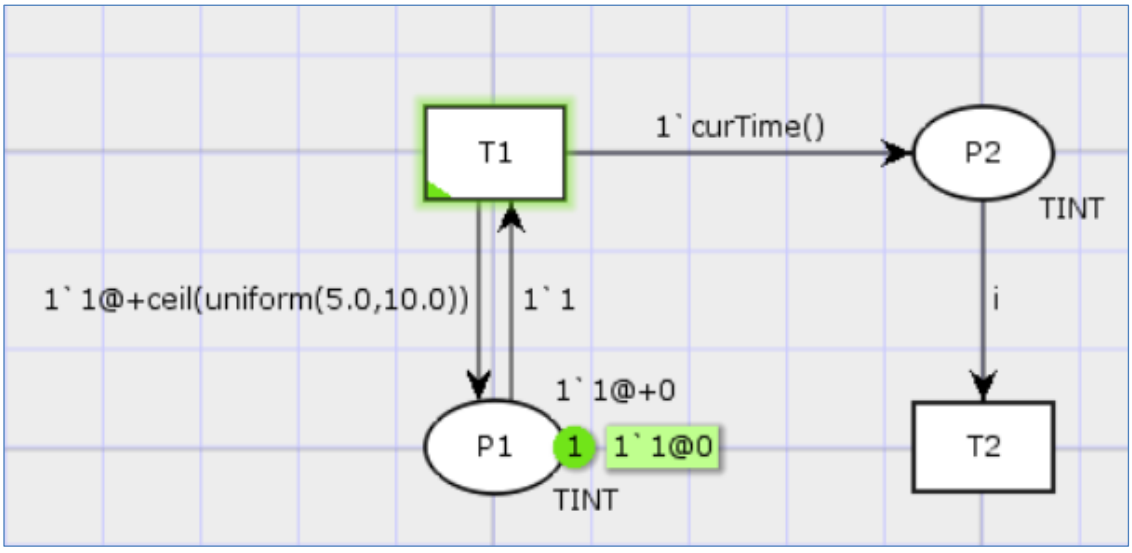


Рис. 22. Приклад використання функції генерації випадкових чисел

Крім розглянутих функцій *uniform* і *normal CPN ML* містить ще кілька функцій для моделювання випадкових величин з іншими законами розподілу. Перелік функцій з їх докладним описом і діапазонами допустимих значень для вхідних параметрів представлений в таблиці 6.

Таблиця 6 - **Функції *CPN ML* для генерації випадкових величин**

Функція зі списком параметрів	Опис функції
bernoulli(p:real) : int	*Розподіл Бернуллі.* Діапазон допустимих значень для параметра *p*: *0.0<=p<=1.0.* Тип значення, що повертається ціле число, зазначений в оголошенні функції після двокрапки. У цій та інших функціях, в разі виходу значень параметра за межі допустимих границь буде згенеровано виняток та моделювання припиниться.
binomial(n:int, p:real) : int	*Біноміальний розподіл.* Діапазон значень параметрів: *n>=1* та *0.0<=p<=1.0*
chisq(n:int) : real	*Розподіл ксі-квадрат.* Діапазон значень параметра: *n>=1.*
discrete (a:int, b:int) : int	*Дискретний рівномірний розподіл.* Параметр *a* менше або дорівнює параметру *b*, *a<=b.*
erlang (n:int, r:real) : real	Гамма розподіл з цілим параметром *n* (*розподіл Ерланга*). Діапазон значень параметрів: *n>=1* та *r>0.0*
exponential(r:real) : real	*Експоненціальне розподіл.* Діапазон значень параметра *r>0.0*
normal(n:real, v:real) : real	*Нормальний розподіл.* Діапазон значень параметра *v>=0.0*
poisson(m:real) : int	*Розподіл Пуассона.* Діапазон значень параметра *m>0.0*
student (n:int) : real	*Розподіл Стьюдента.* Діапазон значень параметра *n>=1.*
uniform(a:real, b:real) : real	*Рівномірний розподіл.* Параметр *a* менше або дорівнює параметру *b*, *a<=b.*
rayleigh(s:real) : real	*Розподіл Рейлі.* Діапазон значень параметра *s>=0.0*
gamma(l:real, k:real) : real	*Гамма розподіл.* Діапазон значень параметрів *l,k>=0.0*
beta(a:real, b:real) : real	*Бета розподіл.* Діапазон значень параметрів *a,b>=0.0*

Більшість функцій приймають в якості параметрів вирази типу *real*. Для роботи цих функцій необхідно або явно вказати значення параметра із зазначенням частини числа після десяткового роздільника, наприклад 1.0 або 11.34, або перетворити число до типу *real* наступним способом: *real int_value*, де: *int_value* - це вираз типу *INT*.

Для завдання значень маркерами на основі значення, що повертається типу *real* і для передачі цілочисельних параметрів в функції необхідно округлити результат за допомогою функцій *floor (real v)* або *ceil (real v).* Функція *floor (real v)* округлює дійсне число до найближчого цілого, меншого або рівного *v*. Функція *ceil (real v*) округлює дійсне число до найближчого цілого, більшого або рівного *v*. Використання в якості параметра функції виразу з типом, відмінним від очікуваного, призведе до помилки в моделі та до неможливості її виконання.

3.6 Статистика

Для накопичення статистики про результати роботи моделей можуть бути використані різні методи. У даній роботі буде розглянуто метод *накопичення інформації в позиціях*. На рисунку 23 наведено приклад мережі, в якій здійснюється генерація міток з випадковими значеннями і підрахунок деяких характеристик множини згенерованих випадкових чисел. *Max* – позиція для зберігання максимального значення випадкового числа, *Min* – позиція для зберігання мінімального значення випадкового числа, *Avg* – позиція для зберігання середнього значення випадкового числа, *Num* – допоміжна позиція для зберігання значення кількості випадкових чисел, що були згенеровані. *Summ* – допоміжна позиція для зберігання суми згенерованих випадкових чисел. У даній моделі використовуються наступні змінні *var i, iMax, iMin, iAvg, iNum, iSum: INT*; Накопичення статистики є важливим елементом процесу моделювання.

Деякі характеристики моделі можуть бути обчислені в реальному часу, в процесі моделювання, для інших можуть знадобитися додаткові розрахунки в спеціалізованих математичних додатках.

У другому випадку результати моделювання зручно зберегти в файл для подальшого аналізу. Інструменти, що дозволяють зробити це, будуть розглянуті в практичних роботах 4 і 5.

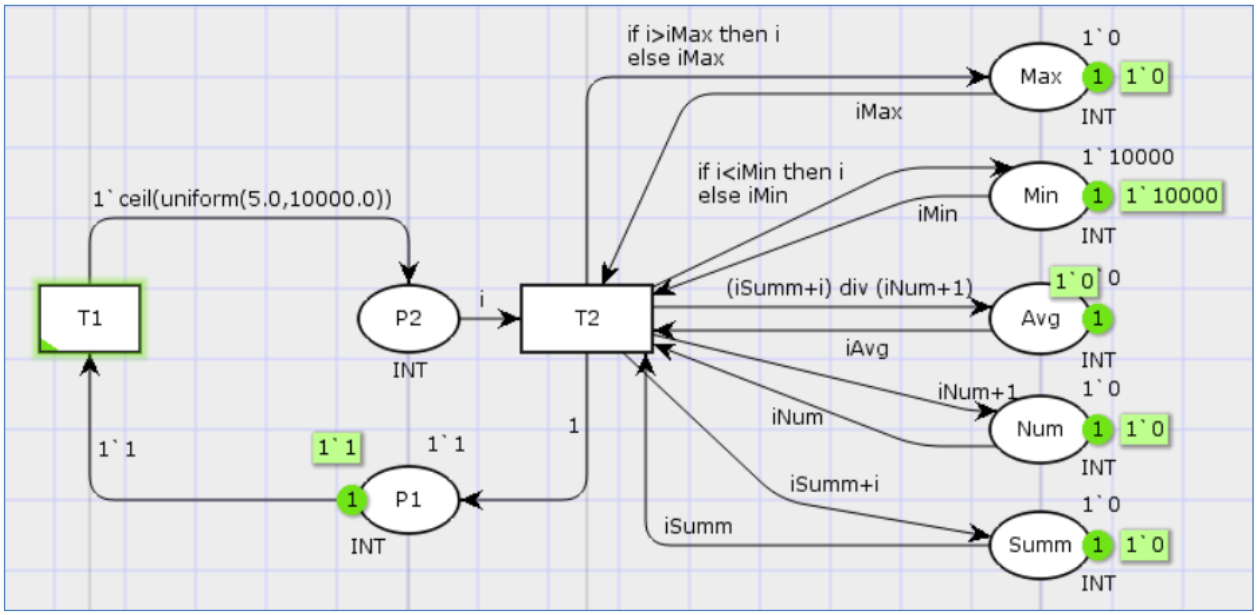


Рис. 23. Накопичення статистики

В даному розділі представлено лише невеликий перелік характеристик, які можуть бути отримані на етапі моделювання. Скориставшись вбудованими функціями *CPN Tools*, можливо отримати більш складні характеристики, однак це підвищить складність розрахунків та збільшить час моделювання.

Завдання для практичної роботи №3

Побудувати модель, що дозволяє аналізувати роботу центру технічної підтримки. Існує багатоканальний телефонний номер, на який надходять дзвінки користувачів з питаннями. Так як дзвінків багато, деяким користувачам доводиться чекати, поки один з операторів звільниться. Користувачі розділені на групи за пріоритетами (*N* груп), для кожної групи реалізована окрема черга.

У центрі технічної підтримки працюють кілька операторів (*M* людей). Вільний оператор повинен вибрати користувача з найбільшим пріоритетом у черзі. Кожен користувач звертається в службу технічної підтримки з однією з *K* проблем. Кожній проблемі відповідає час, за яке оператор може знайти рішення *KMin* і *KMax* для рівномірного розподілу часу відповіді та *KN* і *KV* для нормально розподіленого часу відповіді. У разі, якщо кількість користувачів, які одночасно очікують дзвінка, досягло величини *UMax*, то всім наступним користувачам буде відмовлено в технічній підтримці.

Змоделювати роботу центру технічної підтримки з обслуговування *1000* клієнтів. Завдання відсоткового співвідношення кількості клієнтів в кожній групі пріоритетів здійснюється студентом самостійно, при цьому число клієнтів в кожній групі не повинно бути менше *5%* від загального числа.

Передбачувана структура моделі представлена на рисунку 24.

Змоделювати роботу центру технічної підтримки протягом *24* годин при тих же параметрах. Значення параметрів *N, M, K, KMin, KMax, KN, KV, UMax* і закон розподілу часу відповіді оператора відповідає номеру варіанта, що представлено в таблиці 7. Час відповіді задано в секундах, студенту необхідно самостійно визначити відповідну кількість тактів модельного часу.

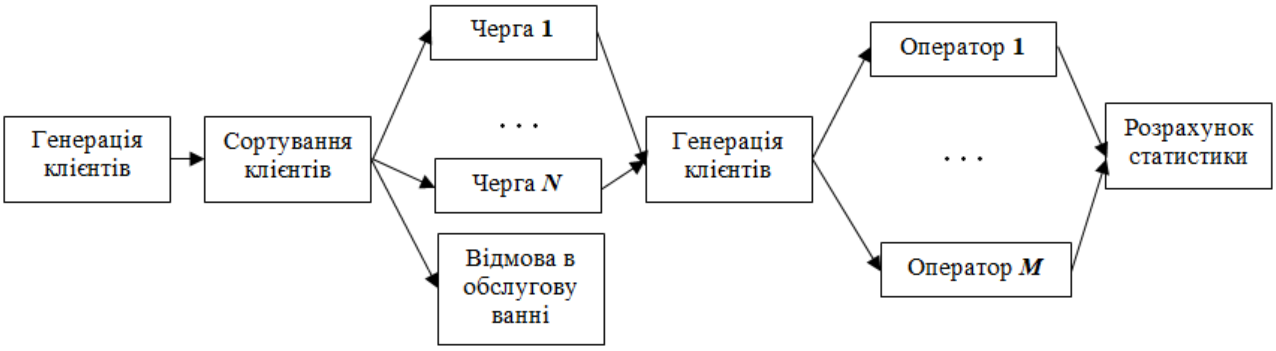


Рис. 24. Передбачувана структура моделі центру технічної підтримки

Таблиця 7 - **Значення параметрів для побудови моделі**

№	*N*	*M*	*K*	*KMin*	*KMax*	*KN*	*KV*	*UMax*
1	2	6	10	-	-	60	20	50
2	3	5	12	45	80	-	-	35
3	4	4	15	-	-	70	25	45
4	2	6	13	50	90	-	-	55
5	3	5	16	-	-	55	25	40
6	4	4	11	40	85	-	-	50
7	2	6	14	-	-	80	30	35
8	3	5	12	30	90	-	-	50
9	4	4	15	-	-	50	20	45
10	2	6	17	55	75	-	-	50
11	3	5	13	-	-	65	15	50
12	4	4	11	45	85	-	-	55
13	2	6	12	-	-	90	35	45
14	3	5	14	45	70	-	-	35
15	4	4	15	-	-	65	25	40
16	2	6	13	50	75	-	-	45
17	3	5	17	-	-	75	15	60
18	4	4	11	30	80	-	-	55
19	2	6	12	-	-	85	25	40
20	3	5	14	40	75	-	-	45

Зміст звіту по практичній роботі №3

1. Завдання на практичну роботу з обраним варіантом. У завданні вказати обрані значення параметрів з таблиці 7.
2. Скріншот моделі, визначення типів даних та змінних, використаних в моделі.
3. У звіті повинні бути представлені наступні характеристики, пов'язані з роботою центру технічної підтримки (за результатами обслуговування *1000* клієнтів):
    - Середній час очікування клієнтів в черзі для кожної групи пріоритетів;
    - Загальна кількість обслужених клієнтів;
    - Кількість обслужених клієнтів по групах пріоритету;
    - Кількість клієнтів яким було відмовлено в обслуговуванні.
4. Результати аналогічних пункту 3 для моделювання *24* годин роботи центру.
5. Відповіді на контрольні питання.
6. Висновок, що відбиває зміст і результати виконаної роботи.

Контрольні питання. Блок №3

1. Яким чином здійснюється завдання типу даних з урахуванням часових міток?
2. Яким чином здійснюється установка значень часових міток маркерами в *CPN Tools*?
3. Яким чином може бути реалізована затримка спрацьовування переходу?
4. Яким чином здійснюється зв'язок реального і модельного часу?
5. Як змінюється порядок спрацьовування переходів з введенням часових міток в маркерах?
6. Як здійснюється генерація випадкових величин? Наведіть деякі функції, які використовуються для генерації випадкових величин з різними законами розподілу.

4. Практична робота №4. Ієрархія моделей

У даній практичній роботі розглядаються можливості CPN Tools побудови моделей з вкладеними компонентами. Описуються моделювання по схемі «зверху-вниз» і «знизу вверх», розглядається приклад використання моделей нижчого рівня. Вводиться поняття підстановочних переходів. Розглядаються альтернативні способи зв'язку частин моделі за допомогою спільних позицій.

4.1 Ієрархія моделей

Моделі реальних об'єктів часто досягають великих розмірів, що ускладнює їх візуальне сприйняття і модифікацію. Для збереження наочності модель може бути структурована, це можна зробити, виділивши головні та підлеглі частини моделі або представити модель у вигляді набору взаємодіючих компонентів. При цьому в кожному компоненті нижчого рівня будуть присутні елементи, що відповідають за обмін даними з іншими моделями, а також елементи, що реалізують основні функції.

Побудова моделей з ієрархічною структурою відбувається в три етапи.

1. Побудова моделі, яка задає структуру. Надалі в цій моделі деякі переходи будуть замінені на моделі нижчого рівня (рисунок 25). Якщо цей етап виконується першим, то такий підхід називається *низхідним* проектуванням. В даному підході спочатку реалізується верхній рівень ієрархії моделей, а потім відбувається моделювання окремих підсистем та їх функцій.

2. Побудова моделей нижчого рівня, що моделюють окремі функціональні елементи (рисунок 26). Якщо цей етап виконується першим, то такий підхід називається *висхідним* проектуванням. Реалізація моделі починається з побудови окремих функціональних елементів, які в подальшому об'єднуються в більш складну структуру.

3. Заміна переходів на моделі нижчого рівня за допомогою інструментів вкладки **Hierarchy** (рисунок 27). Коли готова модель, що представляє структуру системи, деякі переходи замінюються на моделі складових частин.

Побудова *батьківської* та *вкладеної* моделей здійснюється на різних сторінках проекту. Для додавання сторінок до проекту натисніть правою кнопкою миші на ім'я вашої моделі в області меню. У контекстному меню виберіть пункт **New Page**. На рисунку 25 перехід *SN* містить вкладену модель.

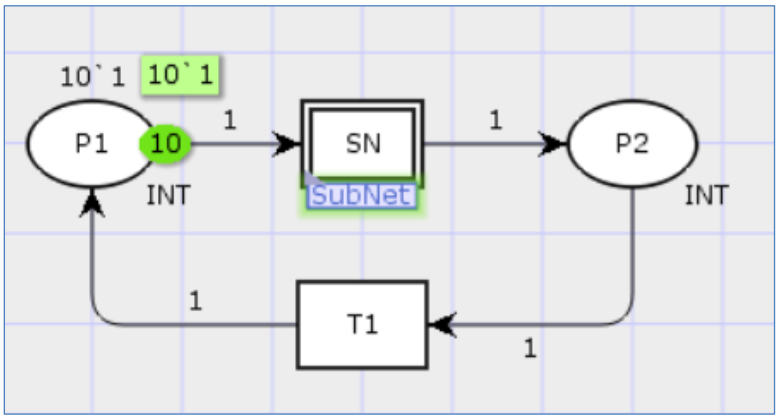


Рис. 25. Батьківська модель з вкладеною моделлю

На рисунку 26 представлена вкладена модель переходу *SN* (з рис. 25). Позиція *In* є вхідною, їй відповідає позиція *P1* моделі верхнього рівня. Позиція *Out* є вихідною, їй відповідає позиція *P2*.

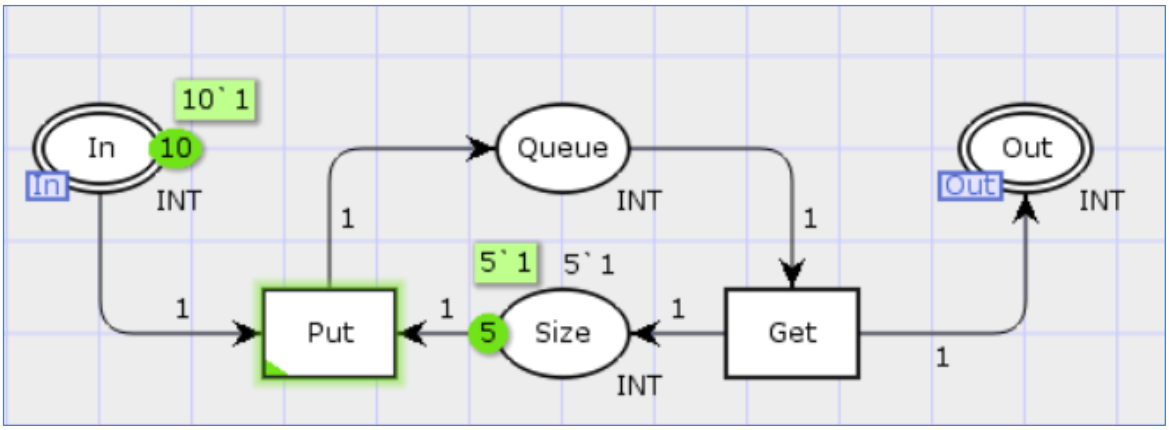


Рис. 26. Вкладена модель

Кожна вкладена модель створюється на окремій сторінці. Для створення нової сторінки необхідно натиснути правою кнопкою миші на імені моделі в області меню і вибрати пункт **New Page**. Після цього до моделі буде додано нову сторінку. Нову сторінку можна перетягнути в робочу область, при цьому буде створено вікно з її вмістом, або перетягнути в існуюче вікно, тоді у вікні з'явиться нова вкладка з елементами моделі даної сторінки.

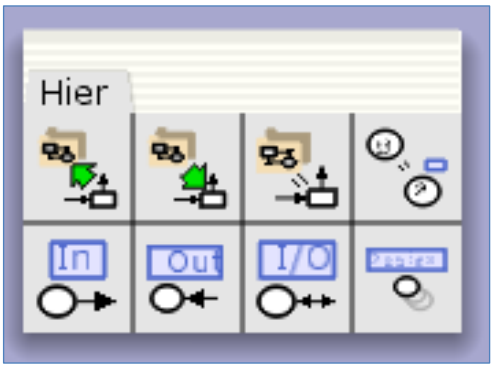


Рис. 27. Інструменти для побудови ієрархії моделей

Детальний опис інструментів вкладки **Hierarchy** представлено в таблиці 8.

Застосовуючи дані інструменти, можна побудувати модель з необхідною ієрархією і зв'язками компонентів.

4.2 Побудова батьківської моделі з вкладеними моделями нижчого рівня

Для побудови моделі з вкладеними компонентами можна скористатися методологією спадного проектування і виконати наступні дії:

1. Виконати розбиття системи на функціональні елементи, виявити ієрархію елементів та їх зв'язки.

2. Побудувати модель в термінах мережі Петрі, в якій всі компоненти будуть представлені переходами, виконати зв'язок компонентів.

Таблиця 8 - **Опис інструментів вкладки *Hierarchy***

Зображення інструмента	Опис інструменту
	*Помістити перехід в окрему модель нижчого рівня*. При цьому буде збережена функціональність моделі верхнього рівня та будуть автоматично створені вхідні і вихідні позиції, необхідні для зв'язку моделі нижчого рівня з моделлю верхнього рівня. Для цього необхідно, вибравши інструмент, натиснути на відповідному переході.
	*Замінити перехід вмістом відповідної моделі нижчого рівня*. При цьому всі елементи вкладеної моделі будуть вписані в модель верхнього рівня. Для цього необхідно вибрати інструмент та натиснути на відповідному переході.
	*Зв'язати модель нижчого рівня та перехід.* Для цього необхідно вибрати інструмент, натиснути на переході, який повинен стати обраним, потім вибрати сторінку з моделлю нижчого рівня, яка буде поміщена в даний перехід, натиснувши на імені сторінки у вкладці моделі в області меню.
	*Встановити зв'язок між позицією в моделі нижчого рівня і відповідної їй позиції в моделі верхнього рівня*. Позиції повинні бути одного типу. Також в моделі верхнього рівня повинна бути присутнім дуга, що зв'язує позицію та перехід, в якому знаходиться відповідна позиція. Для цього необхідно вибрати інструмент, натиснути на позиції всередині вкладеної моделі, після того як відбудеться автоматичний перехід до моделі верхнього рівня, натиснути на відповідній позиції.
	*Зробити позицію вхідною в даній моделі*. Для цього необхідно вибрати інструмент та натиснути на відповідній позиції.
	*Зробити позицію вихідною в даній моделі.* Для цього необхідно вибрати інструмент та натиснути на відповідній позиції.
	*Зробити позицію вхідною та вихідною одночасно в даній моделі*. Для цього необхідно вибрати інструмент та натиснути на відповідній позиції.
	*Зробити позицію загальної і помістити її в іменовану множину позицій.* При зміні маркування кожній позиції даної множини аналогічні зміни відбудуться і в інших позиціях множини. Для цього необхідно вибрати інструмент, натиснути на позиції, а потім в синьому прямокутнику, що з’явився, ввести ім'я множини спільних позицій.

3. Розробити моделі компонентів. Зробити переходи для моделі верхнього рівня, зв'язавши з моделями відповідних компонентів.

Якщо компоненти моделі також мають ієрархічну структуру або складаються з набору елементів, для них можуть бути виконані описані вище кроки.

4.3 Зв'язок моделей через множину спільних позицій

Для зв'язку окремих компонентів моделі може бути використаний альтернативний підхід, заснований на завданні множини спільних позицій. Загальні позиції в цьому випадку виступають в якості шини даних, яка з'єднує різні ділянки моделі. Однак будувати всю ієрархію компонентів тільки на основі спільних позицій не слід, так як такі зв'язки носять неявний характер, їх складно відстежувати, змінювати і аналізувати.

Найкращим рішенням буде комбінований підхід, в якому ієрархія компонентів буде задаватися за допомогою групових переходів, а функції управління і аналізу будуть реалізовані за допомогою спільних позицій. В такому випадку загальні позиції можуть бути винесені на окрему сторінку і використані для установки початкових значень параметрів моделі або для збору статистики про хід моделювання.

Завдання для практичної роботи №4

Побудувати модель за варіантом з практичної роботи №3 з вкладеними моделями нижчого рівня. Використовуйте будь-яку стратегію побудови ієрархічної моделі. Обґрунтуйте і опишіть всі ієрархічні рівні моделі.

Зміст звіту по практичній роботі №4

1. Завдання на лабораторну роботу з обраним варіантом. У завданні вказати обрані значення параметрів з таблиці 7.
2. Скріншот моделі, визначення типів даних і змінних, використаних в моделі.
3. У звіті повинні бути представлені наступні характеристики, пов'язані з роботою центру технічної підтримки (за результатами обслуговування *1000* клієнтів, стресове тестування):
   - Середній час очікування клієнтів в черзі для кожної групи пріоритетів;
   - Загальна кількість обслужених клієнтів;
   - Кількість обслужених клієнтів по групах пріоритету;
   - Кількість клієнтів яким було відмовлено в обслуговуванні.
4. Результати аналогічних пункту 3 для моделювання *24* годин роботи центру.
5. Відповіді на контрольні питання.

6. Висновок, що відображає зміст і результати виконаної роботи.

Контрольні питання. Блок №4

1. Наведіть приклади, в яких має сенс розбивати модель на компоненти (моделі нижчого рівня).
2. Яким чином в *CPN Tools* здійснюється побудова моделей нижчого рівня та їх зв'язок між собою?
3. Що таке спільні позиції, і в яких випадках вони можуть застосовуватися?

5. Практична робота №5. Оголошення функцій, використання кортежів

У практичній роботі наводиться приклад написання призначених для користувача функцій, приклад використання операторів розгалуження if і case, а також приклад завдання локальних змінних функцій. На завершення практичної роботи розглядається тип даних product (кортеж). Наводиться приклад роботи з кортежами, створення міток типу product і доступу до елементів кортежу.

5.1 Функції

CPN Tools представляє можливості по реалізації користувачем власних функцій, необхідних для побудови моделі. Оголошення функції виглядає наступним чином:

fun fun_name (parameters) = expression,

де *fun_name* - ім'я функції, *parameters* - список параметрів функції (може бути порожнім), *expression* - вираз, відповідно що повертається значенням функції. Виклик цієї функції буде виглядати наступним чином: *fun_name ().*

Розглянемо приклади деяких функцій. Функція *Summ* призначена для підсумовування значень параметрів, функція повертає маркер зі значенням, рівним сумі параметрів *a* и *b*.

fun Summ(a,b)=1`(a+b).

Результат функції залежить від значень параметрів. Для організації розгалуження можуть бути застосовані конструкції *case var_name of* та *if bool_expr then expr1 else expr2*.

```
fun ToString(a)=
case a of
1=> "One"
| 2 => "Two"
```

```
fun IfFun(a)=
if a=1  then 1`"One"
else 1`"Not one"
```

```
| 3 => "Three"
| _ => "Else"
```

Функції *Max2* і *Max3* призначені для знаходження максимального значення серед параметрів.

```
fun Max2(v1, v2)=
if v1>v2 then v1
else v2
```

```
fun Max3(v1, v2, v3)=
let
val pat1 = Max2(v1,v2)
in Max2(pat1,v3) end;
```

У функції *Max3* застосовується конструкція, призначена для завдання локальних змінних в тілі функції:

```
let
val pat1 = exp1 val pat2 = exp2
...
val patn = expn
in
exp
end,
```

де *pat1*, *pat2*, ... , *patn* – імена локальних змінних, *exp1*, *exp2*, ..., *expn* – вирази, що визначають значення відповідних локальних змінних, *exp* – вираз, що визначає значення, що повертається функцією.

Для поліпшення наочності моделі функції користувача слід привести окремо від визначень типів і оголошень змінних в спеціальному блоці (рис. 28).

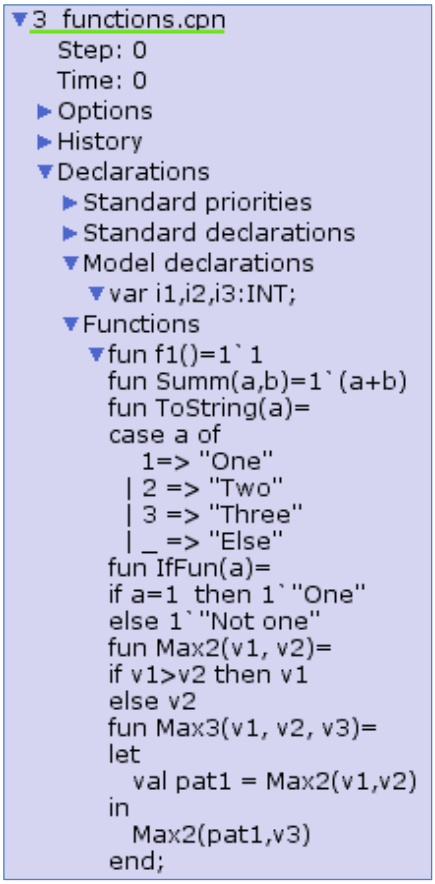


Рис. 28. Оголошення функцій користувача в інтерфейсі CPN Tools

Для розгляду прикладу роботи з функціями були оголошені наступні змінні:

var i1,i2,i3:INT.

Приклад використання функцій користувача наведено на рисунку 29. Позиції *P1*, *P2*, *P3* містять маркери, які використовуються в якості значень змінних *i1*, *i2*, *i3* на вхідних дугах переходу *T1*. *Summ* – позиція, яка містить

суму двох змінних *i1 і i2, StringVal* – позиція, яка містить маркер з результатом функції *ToString*, з параметром *i3*, *Max2*, *Max3* – позиції, що містять маркери з найбільшим значенням вхідних параметрів функцій *Max2 (v1, v2) і Max3 (v1, v2, v3)* відповідно.

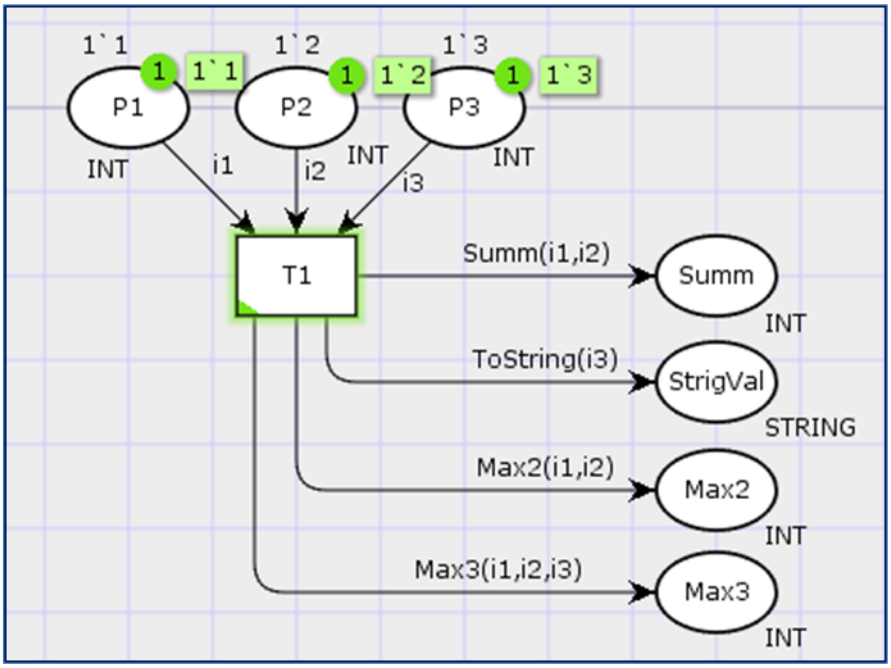


Рис. 29. Приклад використання функції користувача

Застосування функцій дозволяє скоротити кількість елементів в моделі, підвищити наочність і наблизити модель до предметної області, для якої вона будується.

5.2 Використання кортежів

При побудові складних моделей часто виникає потреба в завданні маркера масиву значень. В якості альтернативи типу *record* зручно використовувати тип *product*, що представляє собою кортеж (*tuple*) значень різних типів. Синтаксис оголошення типу даних кортежів виглядає наступним чином:

*colset CS_NAME = product TYPE_1 * TYPE_2 * ... * TYPE_n;*

n>=2,

де *CS_NAME* – ім'я нового типу даних, *TYPE_1, TYPE_2, ..., TYPE_n* – імена типів даних, елементи яких входять в кортеж, *n* - кількість елементів кортежу.

Як можна бачити з синтаксису кортежу, основна його відмінність від типу *record* полягає у відсутності іменованих елементів. У кортежі звернення до елементів здійснюється за індексом:

#index product_variable,

де *index* – ціле число, індекс адресується елемента (індексація починається з одиниці), *product_variable* – ім'я змінної типу *product.*

Генерація міток типу *product* в виразах на дугах здійснюється наступним чином:

1`(v1, v2, …, vn),

де *v1, v2, ..., vn* – значення елементів кортежу, при цьому їх кількість і типи повинні відповідати типам і кількості елементів, присутніх у визначенні.

Приклад використання типу *product* наведено на рисунку 30. У даному прикладі використаний тип *PTEST* та змінна *pt.*

*colset PTEST = product INT * INT;*

var pt:PTEST;

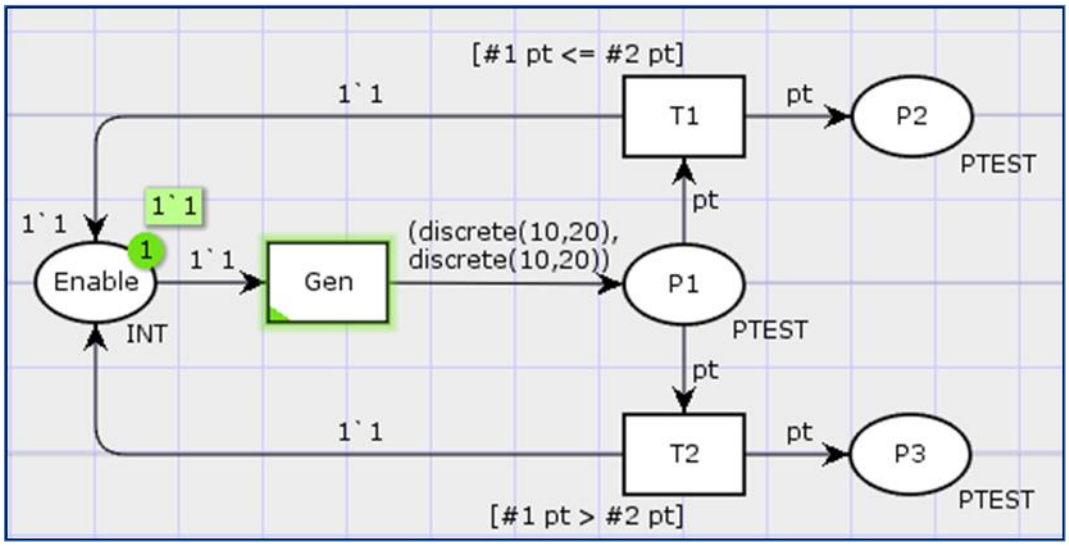


Рис. 30. Приклад використання типу product в моделі

Gen – перехід, який здійснює генерацію маркерів типу *PTEST*, елементам кортежу присвоюються випадкові значення. *P1* – позиція, в яку надходять згенеровані маркери. *T1, T2* – переходи, які здійснюють переміщення маркерів в позиції *P2* та *P3* відповідно. Який з переходів при цьому спрацює, визначається на основі охоронного виразу. *Enable* – позиція, що відповідає за те, щоб в кожен крок модельного часу в позиції *P1* знаходилося не більше одного маркера.

Завдання для практичної роботи №5

Побудувати модель, що дозволяє оцінювати затримки, що виникають при роботі користувача з *web*-додатком. Модель повинна включати в свій склад наступні компоненти:

• **Модель користувача**. Завданням даного компонента є генерація запитів до сервера з *web*-додатком.

• **Модель каналу даних**. Канал даних може бути змодельовано за допомогою внесення постійної затримки до часової мітки маркерів, що проходять через нього.

• **Модель мережевого пристрою**, через які проходять пакети даних з запитами користувача. Модель мережевого пристрою являє собою чергу, в якій пакети даних очікують відправки в мережу.

• **Модель сервера**, на якому розгорнуто *web*-додаток. Сервер *web*-додатку додає випадкову затримку до часової мітки маркера, що моделює запит (пакет даних) та відправляє маркер назад користувачеві.

Рекомендована структура моделі представлена на рисунку 31.

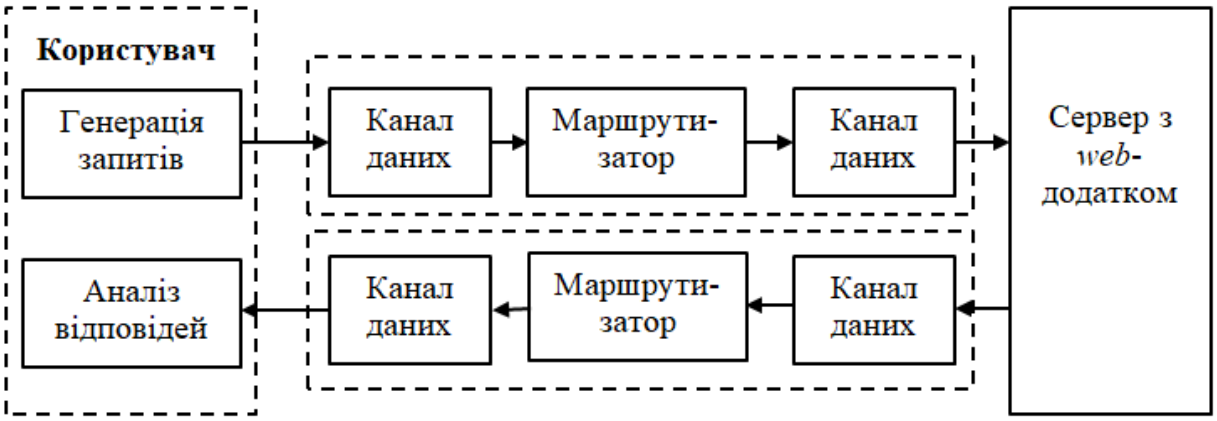


Рис. 31. Рекомендована структура моделі

Користувацький запит і відповідь сервера розміщуються в одному пакеті даних. Всі пакети даних мають однакову довжину (*1500* байт), а також відповідно і однаковий час передачі через канал даних. У маршрутизаторах організована одна черга обмеженої довжини *QMax* для всіх вхідних пакетів. У разі, якщо під час надходження чергового пакету на вхідний інтерфейс маршрутизатора чергу заповнена повністю, то пакет, що надходить відкидається без повідомлення користувача. Повторної відправки втрачених запитів не відбувається.

На шляху від користувача до сервера і назад знаходиться по одному маршрутизатору, який з'єднаний каналами даних з пропускною спроможністю

10 мегабіт / секунду.

Сервер може одночасно обробляти *1* запит користувача, при цьому *K* запитів чекатимуть у черзі. У разі, якщо досягнуто граничне число запитів, що чекають у черзі, то всі нові запити відкидаються.

Інтервал між створенням запитів є випадковою величиною, з рівномірним розподілом від *GMin* до *GMax*. Час обробки запитів є випадковою величиною, з рівномірним розподілом від *RMin* до *RMax*.

Провести стресове тестування *web*-додатку. Необхідно підібрати такі значення затримки генерації пакетів, щоб запити відправлялися з максимальним завантаженням каналу даних.

Значення параметрів *K, GMin, GMax, RMin, RMax, QMax* і закон розподілу часу обробки запитів відповідно до номера варіанта представлені в таблиці 9. Часові інтервали задані в мілісекундах. Студенту необхідно самостійно визначити відповідну кількість тактів модельного часу.

Зміст звіту по практичній роботі №5

1. Завдання на лабораторну роботу з обраним варіантом. У завданні вказати обрані значення параметрів з таблиці 9.
2. Скріншот моделі, визначення типів даних та змінних, використаних в моделі.
3. Для кожного сценарію виконання моделі (з параметрами таблиці 9, стресове тестування) привести наступні результати:
   - Кількість запитів оброблених *web*-додатком;
   - Середнє, максимальне, мінімальне час обробки запиту, з урахуванням затримки передачі (передача 1 + обробка + передача 2);
   - Кількість відкинутих пакетів даних маршрутизаторами;
   - Кількість призначених для користувача запитів відхилених сервером.
4. Відповіді на контрольні питання.
5. Висновок, що відбиває зміст і результати виконаної роботи.

Таблиця 9 - **Значення параметрів для побудови моделі**

№	*K*	*GMin*	*GMax*	*RMin*	*RMax*	*QMax*
1	50	0.8	2.0	1.0	1.5	100
2	55	1.0	1.5	0.5	1.9	120
3	70	1.1	2.3	0.7	1.8	110
4	45	0.9	2.0	1.0	1.5	150
5	65	0.8	2.3	0.8	2.3	140
6	60	0.7	2.8	0.9	2.6	110
7	50	0.9	2.5	1.1	2.0	130
8	45	0.9	2.2	0.9	2.2	120
9	65	1.0	1.8	1.1	1.7	150
10	75	0.8	2.1	0.9	1.9	140
11	45	1.1	2.7	0.6	3.2	100
12	70	1.0	2.0	0.9	2.1	110
13	60	1.1	2.3	1.1	2.3	130
14	55	0.8	2.5	0.9	2.6	140
15	45	1.0	2.1	0.8	2.7	150
16	65	0.8	2.0	1.0	1.5	140
17	55	1.0	1.5	0.5	1.9	120
18	50	1.1	2.3	0.7	1.8	130
19	70	0.9	2.0	1.0	1.5	110
20	80	0.8	2.3	0.8	2.3	140

Контрольні питання. Блок №5

1. Що таке підстановочні переходи?
2. Яким чином здійснюється написання призначених для користувача функцій в *CPN Tools*? Наведіть приклад функції.
3. Наведіть приклад функції з використанням операторів розгалуження *if then else i case*.
4. Яким чином здійснюється завдання локальних змінних у функціях?
5. Що таке тип даних *product*? Яким чином його оголосити?
6. Як здійснюється доступ до елементів кортежів? Як створюються мітки типу *product*?

6. Практична робота №6. Вивід результатів моделювання, простір станів, робота з файлами

У даній практичній роботі розглядаються стандартні засоби CPN Tools, призначені для виведення інформації про хід моделювання. Розглянуто приклад виведення даних в файл. Розглядається аналіз кольорових мереж Петрі на основі простору станів і засобів CPN Tools, призначені для його

6.1 Файли

У *CPN Tools* є можливість виведення даних безпосередньо в файл. Дана можливість може бути використана в разі, якщо можливостей моніторів мало, або необхідно виводити дані, дотримуючись певного форматування. Перед початком роботи з файлом необхідно оголосити глобальну змінну, яка буде посиланням на відкритий файл:

globref outfile = TextIO.stdOut.

Глобальні змінні оголошуються за допомогою ключового слова *globref*, запис *TextIO.stdOut* означає початкове значення даної змінної. Змінна *outfile* має тип *TextIO.outstream* (вихідний потік). Для отримання значення глобальної змінної використовується запис *global_variable*, де *global_variable* – ім'я глобальної змінної, для присвоєння значення запис *global_variable: = new_value*; де *new_value* – вираз, значення яке буде присвоєно глобальної змінної.

Вивід в файл здійснюється при спрацьовуванні переходу *Action*, в тексті річного сегменту (рис. 32). Для доступу до кодового сегменту переходу необхідно виділити перехід та три рази натиснути клавішу *tab*. Кодовий сегмент виглядає наступним чином:

```
input (i);
action
outfile := (TextIO.openAppend("OutFile.txt"));
TextIO.output(!outfile, ("Value: "^Int.toString(i)^"\n"));
TextIO.closeOut(!outfile)
```

Даний кодовий сегмент складається з двох секцій *input* і *action*. У секції *input* вводяться імена змінних, які використовуються в кодовому сегменті. У секції *action* відбувається безпосереднє виконання команд. Приклад виведення даних в файл представлений на рисунку 33.

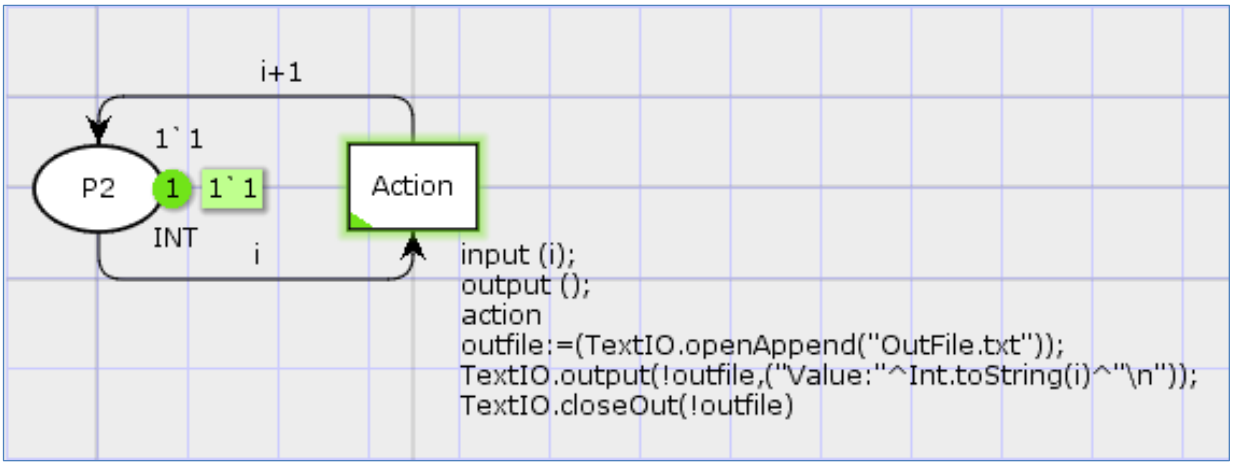


Рис. 32. Приклад виведення в файл

Кодовий сегмент складається з трьох команд:

- *outfile: = (TextIO.openAppend ("OutFile.txt"))*; – відкриття текстового файлу в каталозі з моделлю. Файл відкривається на додавання. Посилання на цей файл присвоюється глобальної змінної *outfile*.
- *TextIO.output (! Outfile, ("Value:" ^ Int.toString (i) ^ "\ n"))*; – вивід рядка в файл *outfile*, першим параметром є посилання на відкритий файл. Рядок, що виводиться є результатом об'єднання трьох рядків *"Value:"*, значення змінної *i*, яке перетворено в рядок *Int.toString (i)*, та рядку *"\ n"*, що призначений для запису в файл символу кінця рядка
- *TextIO.closeOut(!outfile)* – закриття файлу *outfile*.

У момент спрацьовування переходу *Action* здійснюється запис рядка в файл "*Out- File.txt*". Після п'яти кроків моделювання вміст файлу виглядає, як показано на рисунку 33.

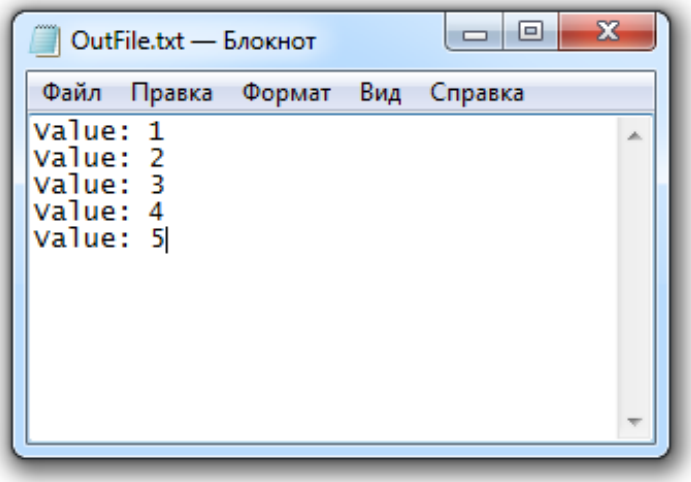


Рис. 33. Текстовий файл з результатами моделювання

Використовуючи даний метод, можна зібрати необхідну статистичну інформацію про процес моделювання та представити її в форматі, зручному для обробки.

6.2 Простір станів

За аналогією з деревом досяжних маркувань в ординарних мережах Петрі, для кольорових мереж вводиться поняття *простору станів*. Простір станів представляється у вигляді графа, вузлами якого є досяжні маркування мережі Петрі. Відрізняє простір станів від дерева досяжності те, що в дереві досяжності маркування представляється кількістю міток в позиції, а в просторі станів при завданні маркування враховується тип позиції, значення та кількість міток.

На рисунку 34 представлений зовнішній вигляд панелі інструментів, призначених для побудови простору станів кольорової мережі Петрі.

У таблиці 11 наведені основні інструменти, необхідні для побудови простору станів кольорової мережі Петрі в програмному середовищі *CPN Tools*.

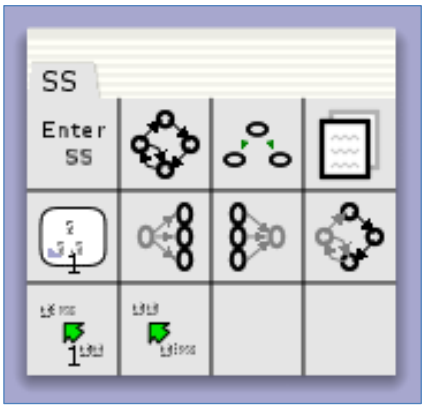


Рис. 34. Панель інструментів побудови простору станів

Перед використанням інструменту «вхід в простір станів» необхідно перевірити модель і переконатися в тому, що вона не містить синтаксичних помилок (елементи з синтаксичними помилками обведені червоним кольором). Усі позиції і переходи повинні мати унікальні імена. У моделі не повинно бути позицій і переходів з порожнім ім'ям. У разі, якщо перераховані вище умови не виконані, почати роботу з простором станів буде неможливо. Як приклад розглянемо мережу Петрі, що моделює взаємне блокування процесів при роботі з ресурсами, що розділені (рисунок 35).

Простір станів для даної моделі представлено на рисунку 36. У просторі станів легко помітити вершину (№ 5), що не має нащадків і яка є тупиковою. Таким чином можна зробити висновок про те, що два процеси, які моделюються, можуть потрапити в ситуацію взаємного блокування. Крім того, вивчивши послідовність спрацьовування переходів, яка привела модель до тупикової маркування, можна виділити неприпустиму послідовність дій.

Таблиця 11 - **Інструменти побудови простору станів**

Зображення інструмента	Опис інструменту
	*Інструмент для входу в простір станів*. Цей інструмент необхідно використовувати перед початком роботи, натиснувши на ньому лівою кнопкою миші, а потім натиснувши на сторінці моделі, для якої буде побудовано простір станів. Даний інструмент здійснює підготовку *CPN Tools* до побудови простору станів.
	*Інструмент побудови простору станів*. Після того як був виконаний вхід в простір станів, *CPN Tools* здійснює його розрахунок.
	*Інструмент для відображення вузла простору стану із заданим номером*. За замовчуванням відображається коренева вершина. Для початку побудови простору станів натиснути лівою кнопкою миші на цьому інструменті, а потім на сторінці моделі.
	*Інструмент для відображення нащадків* (*досяжних маркувань*) *заданого вузла простору станів*. Для виконання необхідно вибрати інструмент і натиснути на необхідному вузлі простору станів.
	*Інструмент для відображення батька даного вузла простору станів.* Для виконання необхідно вибрати інструмент і натиснути на необхідному вузлі простору станів.

Дана мережа Петрі моделює роботу двох процесів, які здійснюють паралельний доступ до двох ресурсів, що розділяються №1 і №2 (позиції *R1* і *R2* відповідно).

Робота процесу характеризується трьома змінами стану:

1. *отримання доступу до ресурсу №1* (переходи *T1_1* і *T2_2* для першого і другого процесу відповідно);
2. *отримання доступу до ресурсу ресурсу №2* (переходи *T1_2* і *T2_1* для першого і другого процесу відповідно);
3. *звільнення ресурсів №1 і №2* (переходи *T1_3* и *T2_3* для першого і другого процесу відповідно).

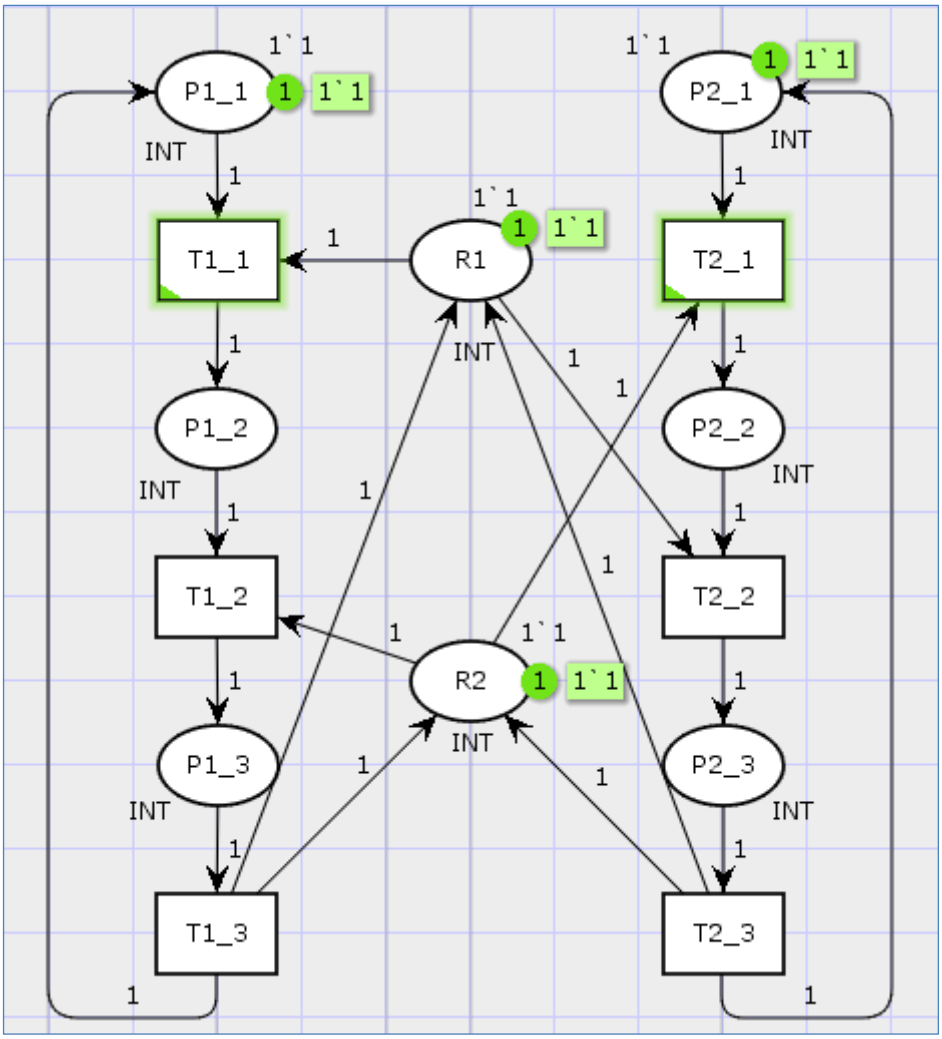


Рис. 35. Мережа Петрі, що моделює взаємне блокування процесів при роботі з ресурсами, що розділяються

Розглянемо простір станів докладніше і виділимо можливі послідовності спрацьовування переходів. Послідовності (*T1_1* → *T1_2* → *T1_3*) і (*T2_1* → *T2_2* → *T2_3*) не призводять до блокування процесів, в той час як послідовність (*T1_1* → *T2_1*) або послідовність (*T2_1* → *T1_1*) призводять до блокування, так як кожен процес отримує доступ до одному з ресурсів, залишаючи при цьому другий процес у стані очікування ресурсу.

Для отримання більш повної інформації про просторі станів передбачена можливість відображення імені переходу, що спрацював, на дузі між двома вузлами та маркування, що відповідає конкретному вузлу простору станів. Для відображення *імені переходу* необхідно натиснути лівою кнопкою миші на дузі. Для відображення *маркування вузла простору станів* необхідно натиснути лівою кнопкою миші на трикутнику, який розташований в лівому нижньому куту зображення відповідно до необхідного вузла простору станів.

Простір станів з відображенням маркувань в вузлах та спрацюванням переходів представлено на рисунку 37.

Простір станів є потужним засобом аналізу кольорових мереж Петрі. Його можна використовувати для виявлення багатьох властивостей моделі (*наявність тупиків, безпека, обмеженість, досяжність* та інші). Однак для складних моделей розміри простору станів можуть бути надзвичайно великими. У таких випадках варто скористатися статистичними методами для аналізу даних, зібраних в ході моделювання

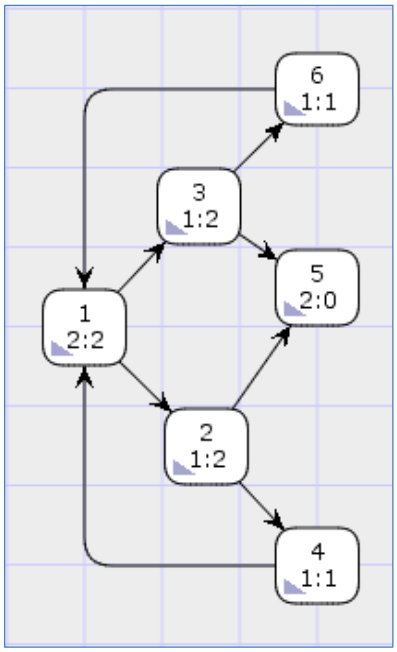


Рис. 36. Простір станів моделі взаємного блокування процесів

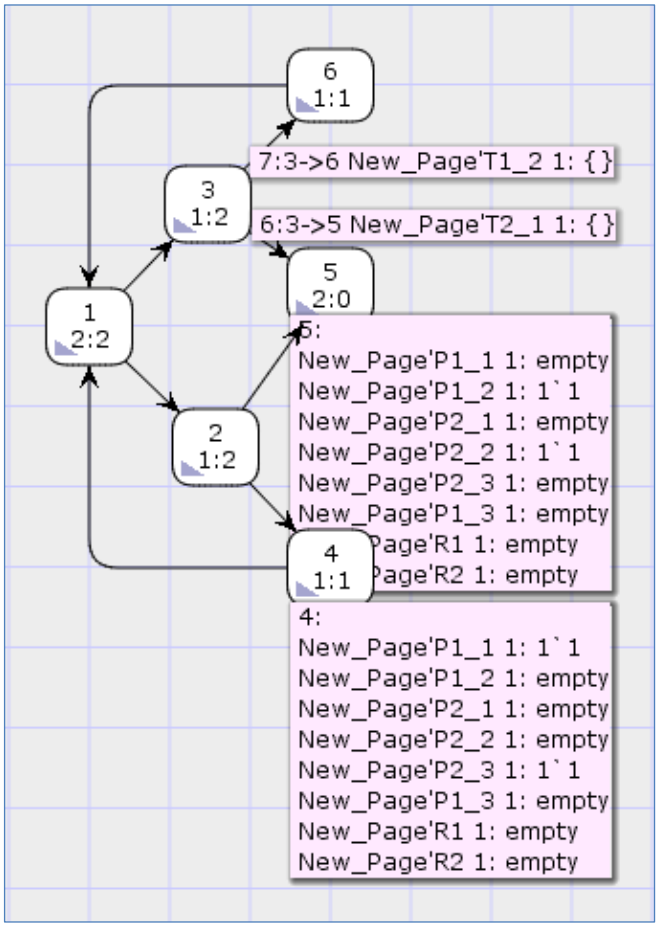


Рис. 37. Простір станів, відображення переходів на дугах і маркування вузлів

Завдання для практичної роботи №6

Побудувати мережу, що моделює конкуренцію довільної кількості процесів за будь-яку кількість ресурсів. В якості основи може бути використана модель, рисунку 35. Модель повинна бути побудована таким чином, щоб для завдання кількості ресурсів і процесів досить було змінити початкове маркування відповідних позицій. Побудувати простір станів для отриманої моделі. Для того щоб процес виконав роботу, йому потрібні всі ресурси. Кількість ресурсів і процесів в залежності від варіанту завдання представлено в таблиці 12.

Таблиця 12 – **Варіанти завдань**

№ варіанта	Кількість процесів	Кількість ресурсів
1	2	4
2	4	3
3	4	2
4	3	3
5	3	4
6	3	2
7	2	3
8	4	4
9	5	2
10	2	5
11	3	5
12	4	5
13	5	2
14	3	6
15	5	3
16	3	6
17	2	6
18	4	6
19	5	6
20	2	6

Зміст звіту по практичній роботі №6

1. Завдання на лабораторну роботу з обраним варіантом.
3. Представити зображення розробленої моделі.
4. Представити зображення отриманого простору станів.
5. Відповіді на контрольні питання.
6. Висновок, що відбиває зміст і результати виконаної роботи.

Контрольні питання. Блок №6

1. Які існують інструменти в *CPN Tools* для виведення інформації про хід виконання моделей. Які характеристики можуть бути отримані?

2. Які засоби передбачені в *CPN Tools* для зупинки процесу моделювання в залежності від стану мережі?

3. Що таке простір станів кольорової мережі Петрі?

4. У чому відмінність простору станів кольорової мережі Петрі від дерева досяжності ординарної мережі Петрі?

5. Яким чином здійснюється побудова простору станів в *CPN Tools*?

6. Яким чином здійснюється робота з файлами в *CPN Tools*?

ЛІТЕРАТУРА

1. Бідюк П.І., Гожий О.П. Ймовірнісно-статистичні методи моделювання і прогнозування: [монографія]. – Миколаїв: Вид-во ЧДУ ім. Петра Могили, 2014. – 440 с.
2. Бідюк П.І., Гожий О.П., Коршевнюк Л.О. Комп'ютерні системи підтримки прийняття рішень [Текст] : навч. посіб.; Нац. техн. ун-т України "Київ. політехн. ін-т", Ін-т приклад. систем. аналізу, Чорномор. держ. ун-т ім. Петра Могили. - Миколаїв : Вид-во ЧДУ ім. Петра Могили; К., 2012. - 379 с.
3. Зайцев Д.А., Шмелева Т.Р. Моделирование телекоммуникационных систем в CPN Tools. Учебное пособие по курсу «Математическое моделирование систем и сетей» для подготовки магистров в отрасли связи – Одесса: ОНАС им. А.С. Попова, 2009. – 72 с.
4. Імітаційне моделювання систем та процесів: Електронне навчальне видання. Конспект лекцій / В. Б. Неруш, В. В. Курдеча. – К.: НН ІТС НТУУ «КПІ», 2012. – 115с.
5. Сети Петри. URL: http://ru.wikipedia.org/wiki/Сети Петри .
6. CPN Tools. URL: http://cpntools.org/
7. Стеценко, І.В. Моделювання систем: навч. посіб. [Електронний ресурс, текст]; М-во освіти і науки України, Черкас. держ. технол. ун-т. – Черкаси: ЧДТУ, 2010. – 399 с.
8. Табунщик Г.В., Каплієнко Т.І., Петрова О.А. Проектування та моделювання програмного забезпечення сучасних інформаційних систем Навч. посібник,. – Запоріжжя, 2016. – 259 с.
9. Томашевський В. М. Моделювання систем . — К.: BHV, 2015.- 352 с.
10. Томашевский В., Жданова Е. Имитационное моделирование в среде GPSS. — М.: Бестселлер, 2013. - 416 с.
11. Томашевський В. М., Жданова О. Г., Жолдаков О. О. Вирішення практичних завдань методами комп'ютерного моделювання. — К.: Корнійчук, 2011. — 267 с.
12. Петрик М.Р. Моделювання програмного забезпечення Науковометодичний посібник /М.Р.Петрик, О.Ю. Петрик. Тернопіль: Вид-во ТНТУ, 2015.- 200 с.