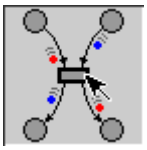
Державний комітет зв'язку та інформатизації України
Одеська національна академія зв'язку ім. О.С.Попова

Д.А.Зайцев



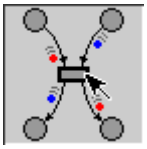
МАТЕМАТИЧНІ МОДЕЛІ ДИСКРЕТНИХ СИСТЕМ

Навчальний посібник з дисципліни
«Математичне моделювання інформаційних систем»
для підготовки магістрів у галузі зв'язку

Одеса 2004

Державний комітет зв'язку та інформатизації України
Одеська національна академія зв'язку ім. О.С.Попова

Д.А.Зайцев



МАТЕМАТИЧНІ МОДЕЛІ ДИСКРЕТНИХ СИСТЕМ

Навчальний посібник з дисципліни
«Математичне моделювання інформаційних систем»
для підготовки магістрів у галузі зв'язку

ЗАТВЕРДЖЕНО
методичною радою
академії
Протокол № 8
від 9.03.2004 р.

Одеса 2004

УДК 51.681.3

План НМВ 2004 р.

Зайцев Д.А. Математичні моделі дискретних систем: Навч. посібник. – Одеса: ОНАЗ ім. О.С. Попова, 2004. – 40 стор.

Подано моделі, що вони відіграють ключову роль у комп'ютерних науках та теорії телекомунікацій: скінченні автомати, сітки Петрі, машини Тюринга. Викладено основні теоретичні результати в означеній галузі, подано методи аналізу й синтезу систем, розглянуто області застосовування. Для закріплення знань наведено контрольні запитання та задачі. Посібник призначено для підготовки магістрів у галузі зв’язку.

Відповідний редактор –

Рецензент – канд.фіз.-мат.наук, доцент **І.В.Стрелковська**

СХВАЛЕНО
на засіданні кафедри
мереж зв'язку й
рекомендовано до друку
Протокол № 10
від 24.04.2003 р.

Одеса 2004

Зміст

Введення

У даному посібнику подано моделі, що вони відіграють центральну роль у комп'ютерних науках. Дійсно, практично всі керуючі й обчислювальні пристрої, різноманітні контролери, використовувані в засобах зв'язку, являють собою скінченні автомати. Сітки Петрі широко застосовують для опису й дослідження паралельних процесів та систем, наприклад протоколів обміну інформацією, операційних систем. Абстрактна машина Тюринга узагальнює попередні моделі й, на відміну від них, включає у своє визначення нескінченні елементи. Основне значення машини Тюринга полягає в формалізації інтуїтивного поняття алгоритму, що дозволяє математично довести алгоритмічну нерозв'язність певних проблем.

1 Скінченні автомати

1.1 Визначення скінченого автомата

Розглянемо абстрактну систему, традиційно називану математиками "чорна скриня" (рис. 1.1), що перетворює подавані на її вхід символи x певної абетки X на вихідні символи y абетки Y. Таким чином можна подати поводження більшості дискретних систем, застосовуваних у різних галузях техніки, надто обчислювальної.

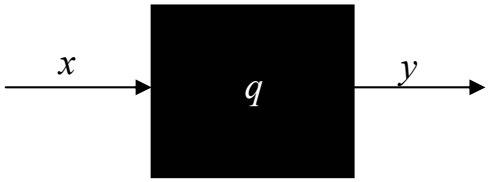


Рисунок 1.1 – „Чорна скриня”

Автомат – це така система, функцію якої може бути описано за допомогою використання внутрішніх станів $Q = \{q\}$ у такий спосіб, що для кожного внутрішнього стану зазначено, до якого наступного стану потрапить система за отримання вхідного символу x і який символ y формується на виході. Зазначається початковий стан системи. У разі скінчених множин X, Y, а, надто, Q, автомат називають скінченним.

Побудуємо скінченний автомат, керуючий ліфтом у триповерховому будинку. Вхідна абетка автомата складається з натискання кнопки виклику відповідного поверху: $X = \{C1,\ C2,\ C3\}$; вихідна абетка складається з переміщень на один чи два поверхи нагору чи донизу, а також зупинки ліфта: $Y = \{U1,\ U2,\ D1,\ D2,\ S\}$; стан відповідає поверхові, на якому перебуває автомат: $Q = \{q_1,\ q_2,\ q_3\}$; для визначеності оберемо початковий стан q_1. Функцію переходів автомата зручно подавати діаграмою станів (рис. 1.2).

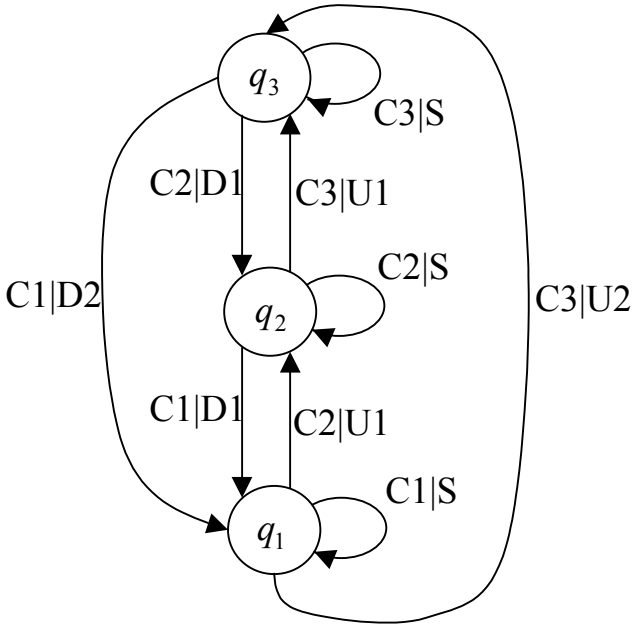


Рисунок 1.2 – Автомат для керування ліфтом

Отже, скінченний автомат – це п'ятірка $A = (X,\ Q,\ Y,\ q_0,\ F)$, де $X = \{x\}$ – скінченна вхідна абетка, $Y = \{y\}$ – скінченна вихідна абетка, $Q = \{q\}$ – скінченна множина внутрішніх станів, q_0 – початковий стан, F: $X \times Q \to Y \times Q$ – функція переходів. Як було розглянуто вище, функцію переходів може бути наочно подано діаграмою станів. Зручним буває також матричне подання функції переходів. Наведемо матрицю переходів для побудованого раніше автомата:

Q/X	C1	C2	C3
q_1	q_1|S	q_2|U1	q_3|U2
q_2	q_1|D1	q_2|S	q_3|U1
q_3	q_1|D2	q_2|D1	q_3 |S

Під синтезом, як правило, розуміють побудову скінченного автомата за певною заданою його специфікацією. Специфікація, як у нашому прикладі з ліфтом, може бути словесною. Іноді використовують формальні специфікації, наприклад, у вигляді регулярних виразів. Синтез автомата полягає в діставанні його формалізованого опису – діаграми станів або матриці переходів. У схемотехніці під синтезом автомата іноді розуміють реалізацію автомата на заданій елементній базі.

1.2 Автомати Мілі та Мура

Подані в попередньому підрозділі скінченні автомати носять ще назву автоматів Мілі. Іноді зручним стає більш просте визначення автомата, в якому функція переходів задає наступний стан, а вихід автомата залежить лише від його поточного стану. Такі автомати названо автоматами Мура. Далі буде показано, що обидва спо-

соби визначення скінченного автомата є еквівалентні в тому розумінні, що будь-який автомат Мілі може бути перетворено на відповідний автомат Мура й навпаки.

Отже, автомат Мура – це шістка $B=(X,\ S,\ Y,\ s_0,\ P,\ R)$, де $X=\{x\}$ – скінченна вхідна абетка, $Y=\{y\}$ – скінченна вихідна абетка, $S=\{s\}$ – скінченна множина внутрішніх станів, s_0 – початковий стан, P: $X\times S\to S$ – функція переходів, $R: S\to Y$ – функція виходів.

Наведемо приклад простого автомата Мура, що розпізнає цілі числа в двійковій системі числення. Абетка вхідних символів: $X=\{0,\ 1,\ -,\ +\}$, абетка вихідних символів $Y=\{0,\ 1\}$; видача символу *0* відповідає нерозпізнаному ланцюжку символів, видача символу *1* відповідає правильно розпізнаному ланцюжку символів. Отже, множина станів поділяється на дві підмножини: проміжні й заключні. Діаграму станів автомата подано на рис. 1.3.

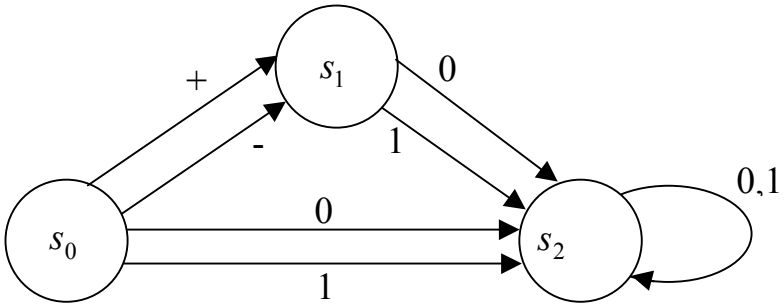


Рисунок 1.3 – Автомат для розпізнавання цілих чисел

Початковий стан автомата s_0; функція виходів: $R(s_0)=0$, $R(s_1)=0$, $R(s_2)=1$. Відзначимо, що переходи зі стану s_2 визначено не для всіх символів алфавіту. Щоби побудувати цілковито визначений автомат, розпізнавач часто доповнюють неповоротним станом, у який переходять за дістанні будь-якого неприпустимого символу.

Покажемо, що за допомогою нескладних побудов довільний автомат Мілі може бути перетворено на автомат Мура. Оскільки в автоматі Мура видавання вихідного символу відбувається за потраплянням до певного стану, то для кожної пари автомата Мілі $(q,\ y)$ утворимо стан автомата Мура *s* зі значенням функції виходів $R(s)=y$. Для переходу автомата Мілі $(q,\ q')$ додаємо переходи з кожного стану *s*, що відповідає *q*, у стан s', визначений парою (q', y). Проілюструємо описані дії на прикладі побудови автомата Мура за автоматом Мілі керування ліфтом (рис. 1.2); діаграму станів дістаного автомата Мура зображено на рис. 1.4.

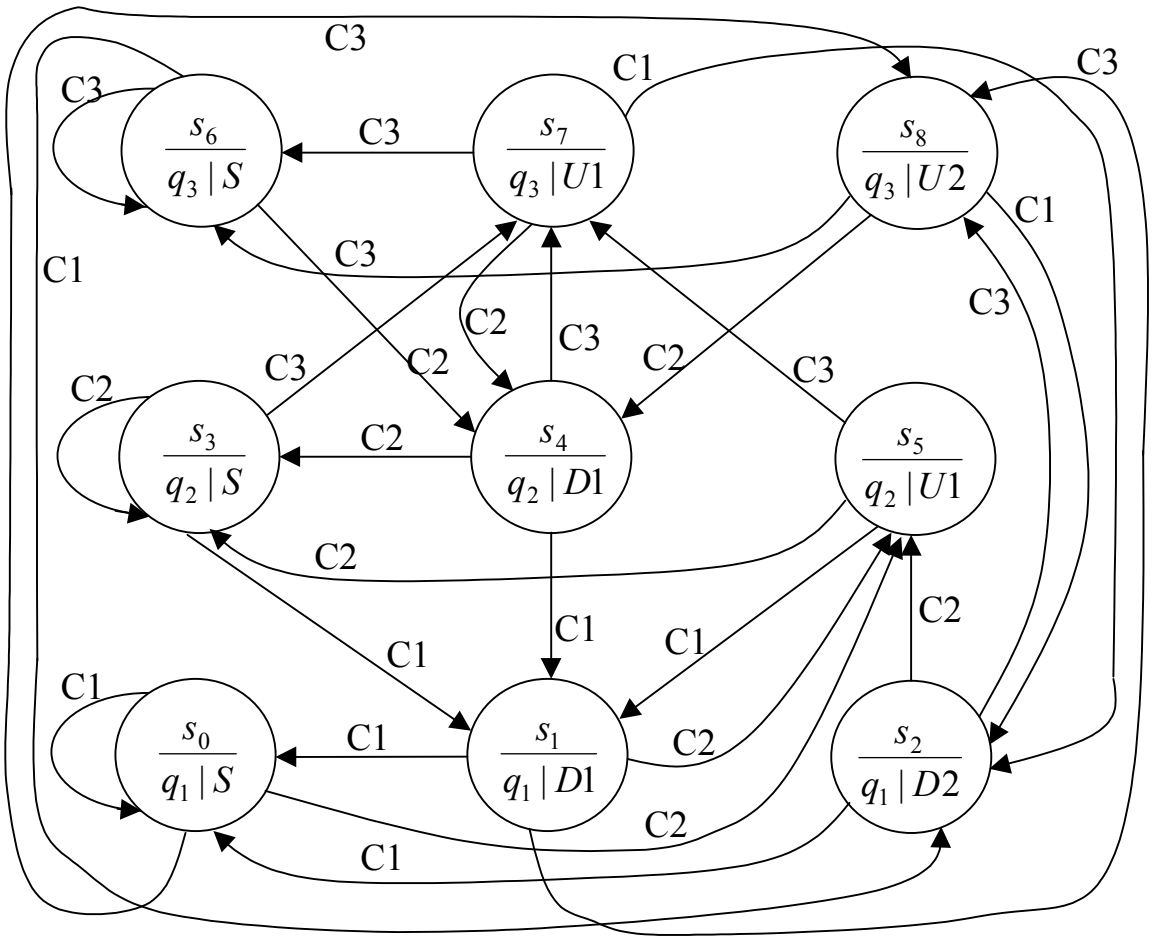


Рисунок 1.4 – Автомат Мура для керування ліфтом

Дістаний автомат має дев'ять станів; відзначимо, що нові стани слід утворювати лише для тих комбінацій станів та вихідних символів, дуги для яких є присутні у вихідному автоматі. Хоча побудований автомат Мура вийшов на перший погляд складнішим за вихідного автомата, використання автоматів Мура, особливо в схемотехніці, виявляється зручнішим.

Формально перетворення можна подати у такий спосіб. Нехай заданий автомат Мілі $A = (X,\ Q,\ Y,\ q_0,\ F)$. Відповідний йому автомат Мура $B = (X,\ S,\ Y,\ s_0,\ P,\ R)$ визначимо як $S = Q \times Y$, $P(s,\ x) = s' \Leftrightarrow F(q,\ x) = (q',\ y)$, $R(s) = y \Leftrightarrow F(q,\ x) = (q',\ y)$; початковий стан s_0 оберемо довільне зі станів, відповідних q_0. Дійсно, такий вибір є обґрунтований, тому що видавання вихідного символу в стані s_0 ми не будемо розглядати, вважаючи його виконаним «до початку» роботи автомата.

Обернене перетворення вимагає набагато менше зусиль і полягає в перенесенні вихідного символу обраного стану на кожну з вхідних дуг. На рис. 1.5 подано автомат Мілі, котрий відповідає автоматові Мура для розпізнавання цілих чисел у двійковій системі числення (рис. 1.3).

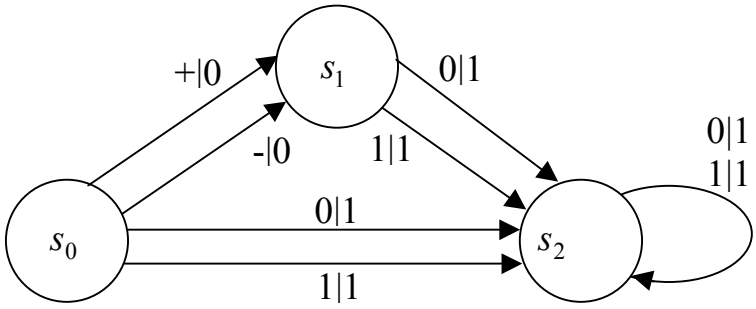


Рисунок 1.5 – Автомат Милі для розпізнавання цілих чисел

Формально перетворення можна подати у такий спосіб. Нехай задано автомат Мура $B = (X,\ S,\ Y,\ s_0,\ P,\ R)$. Відповідний йому автомат Мілі $A = (X,\ Q,\ Y,\ q_0,\ F)$ визначимо, встановивши взаємнооднозначну відповідність станів автоматів: $q \leftrightarrow s$. Зазначимо, що в цьому випадку початковий стан q_0 відповідає станові s_0; до функції переходів внесемо генерацію вихідних символів: $F(q,\ x) = (q',\ y) \Leftrightarrow (P(s,\ x) = s' \vee R(s) = y)$.

Отже, ми засвідчили, що автомати Мілі й Мура мають однакову зображальну потужність, обґрунтувавши перехід від одного автомата до іншого й навпаки. Виникає природне запитання про доцільність застосовування двох типів автоматів. З погляду на зручність побудови, більш прийнятним буває автомат Мілі. Автомат Мура широко використовують у схемотехніці для синтезу реальних обчислювальних та логічних пристроїв на основі елементарних автоматів – тригерів. У таких пристроях стан здебільшого зберігається й модифікується, та видається зовні по спеціальній команді.

1.3 Мінімізація скінченних автоматів

У попередніх підрозділах ми ввели поняття скінченного автомата, побудували конкретні автомати для керування ліфтом та розпізнавання чисел, згадали про можливості апаратної реалізації автоматів. Розглядання простого прикладу,

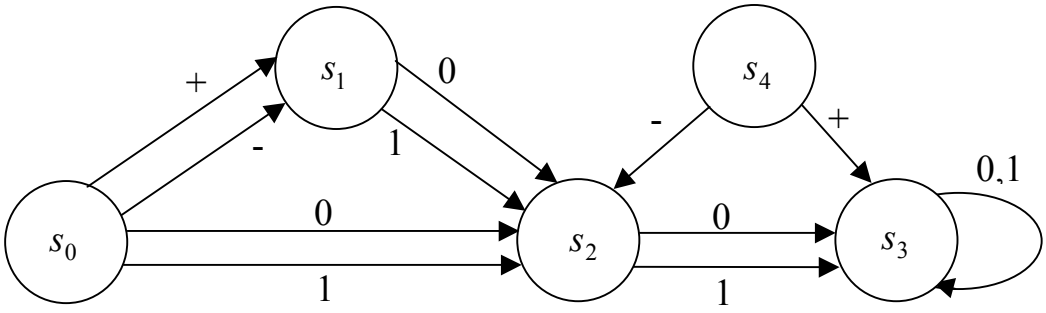


Рисунок 1.6 – Надлишковий автомат для розпізнавання цілих чисел

поданого на рис. 1.6, дозволяє зробити висновок, що той самий автомат може мати безліч подань з різною кількістю станів та переходів. Так, наприклад, стан s_4 є недосяжний з початкового стану і тому може бути вилучений, а стани s_2 та s_3 можна

поєднати; у результаті одержимо раніше розглянутий автомат для розпізнавання цілих чисел (рис. 1.3). Дістати мінімальне подання автомата є надто важливо, якщо надалі його буде реалізовано апаратно. Окрім того, виникає задача для «схожих» автоматів визначити формально, чи є вони еквівалентні.

Надалі розглядатимемо автомати, які не мають недосяжних станів, оскільки такі стани може бути вилучено з діаграми за допомогою методів теорії графів. Методи мінімізації й визначення еквівалентності буде побудовано для автоматів Мура, що, як було відзначено в попередньому розділі, не обмежує їхньої загальності.

Два стани автомата s_1 та s_2 називатимемо *n-еквівалентними*, якщо для довільного вхідного ланцюжка σ довжиною n символів вихідні ланцюжки символів збігаються. Відношення n-еквівалентності позначимо $\overset{n}{\equiv}$.

Два стани автомата s_1 та s_2 називатимемо еквівалентними, якщо вони є *n-еквівалентні* для будь якого цілого n. Відношення еквівалентності позначимо $\equiv$.

Теорема 1.1 У скінченному автоматі з m станами два довільних стани є еквівалентні тоді й лише тоді, коли вони $(m-2)$-еквівалентні.

Доведення. Необхідність умови теореми є прямим наслідком визначень еквівалентності та n-еквівалентності.

Доведемо достатність цієї умови. Покажемо, що має місце наступний ланцюжок включень:

$$\equiv \subseteq \overset{m-2}{\equiv} \subseteq \overset{m-3}{\equiv} \subseteq \ldots \subseteq \overset{2}{\equiv} \subseteq \overset{1}{\equiv} \subseteq \overset{0}{\equiv}$$

Нульеквівалентними будемо вважати стани, якщо вони мають однакові вихідні символи: $q_1 \overset{0}{\equiv} q_2 \;\Leftrightarrow\; R(q_1) = R(q_2)$. Окрім того, зазначемо, що має місце рекурентне відношення:

$$q_1 \overset{k}{\equiv} q_2 \;\Leftrightarrow\; \left(q_1 \overset{k-1}{\equiv} q_2 \right) \;\&\; \left(\forall x \in X: \; P(x, q_1) \overset{k-1}{\equiv} P(x, q_2) \right). \qquad (1.1)$$

Зазначимо, якщо абетка Y нетривіальна й містить більш одного символа, то відношення $\overset{0}{\equiv}$ визначає не меньш аніж 2 класи еквівалентних станів. Якщо $\overset{k+1}{\equiv} \neq \overset{k}{\equiv}$, то $\overset{k+1}{\equiv}$ містить по крайній мірі на 1 клас більше аніж $\overset{k}{\equiv}$.

Тоді послідовність відношень k-евівалентності містить не більш аніж $(m-2)$ різних відношень, тому що у випадку $\overset{k}{\equiv} = \overset{k+1}{\equiv}$ для певного k в силу (1.1) має місце рівність $\overset{k+1}{\equiv} = \overset{k+2}{\equiv} = \ldots$

Отже, відношення $\equiv$ співпадає з першим із відношень $\overset{k}{\equiv}$, для якого виконується рівність $\overset{k}{\equiv} = \overset{k+1}{\equiv}$.

□

Доведена теорема дозволяє запропонувати простий алгоритм мінімізації скінченних автоматів. Алгоритм складається в послідовній побудові розбиття множини станів на одно-, дво-, три- й так далі еквівалентні. Якщо поточне розбиття збігається з попереднім, то дістані класи еквівалентності й визначають мінімальний автомат. У таблиці розбиття зазначено номер класу наступного стану. Первісне розбиття визначається функцією виходу: нульеквівалентні стани мають однакові вихідні символи.

Проілюструємо описаний процес для автомата, поданого на початку цього підрозділу (недосяжний стан s_4 не розглядається):

Клас	Стан	+	-	0	1
K0	s_0	K0	K0	K1	K1
	s_1			K1	K1
K1	s_2			K1	K1
	s_3			K1	K1

Клас K0 розщеплюється на два нових еквівалентних класи, тому що його рядки не збігаються:

Клас	Стан	+	-	0	1
K0	s_0	K1	K1	K2	K2
K1	s_1			K2	K2
K2	s_2			K2	K2
	s_3			K2	K2

Подальшого розщеплення класів еквівалентності не відбувається, тому мінімальний автомат містить три стани; діаграма станів відповідає спочатку побудованій в підрозділі 1.2 й зображеній на рис. 1.3. Стани мінімального автомата відповідають класам еквівалентності.

Алгоритм мінімізації автоматів дозволяє також визначити їхню еквівалентність. Дійсно, якщо необхідно довідатися, чи є еквівалентні автомати B_1 та B_2, доволі виконати мінімізацію автомата $B_1 \bigcup B_2$ й визначити, чи попадають їхні початкові стани до того самого класу еквівалентності.

Отже, скінченні автомати є простим та ефективним засобом подання дискретних систем. Вони широко використовуються в різних галузях техніки. Слід пам'ятати, що будь-який, навіть надто потужний суперкомп'ютер являє собою скінченний автомат. Нехай кількість його станів обчислюється мільярдами, але вона завжди є скінченна.

2 Сітки Петрі

2.1 Сітки Петрі й моделювання систем

Сітка Петрі – це графічний і математичний засіб моделювання систем та процесів. Зазвичай сітками Петрі моделюють асинхронні паралельні системи та процеси. Спочатку запропоновані в докторській дисертації Карла Петрі 1962 року вони набули подальшого розвинення у роботах таких вчених як Тадао Мурата, Курт Йенсен, Віталій Котов, Анатолій Слєпцов. Останнім часом провадиться щорічна конференція «Застосовування й теорія сіток Петрі», видається в Бонні інформаційний бюлетень «Новини сіток Петрі» (Petri Net Newsletter), відомо кілька сот моделювальних систем для різних програмно-апаратних платформ, існують реалізації процесорів сіток Петрі.

Галузі застосовування сіток Петрі включають дослідження телекомунікаційних мереж, мережних протоколів, обчислювальних систем та обчислювальних процесів, виробничих та організаційних систем.

Доволі представницьким є набір реальних проектів, у яких використано моделювальну систему розфарбованих сіток Петрі Design/CPN: проектування інтелектуальних мереж Deutsche Telekom; проектування системи керування мережею в RC International A/S; проектування архітектури нових мобільних телефонів Nokia; система електронних платежів у США; сховище електронних документів Bull AG; система безпеки й контролю доступу Dalcotech A/S; європейська система керування рухом потягів; планування операцій у військово-повітряних силах США; військово-морська командна система в Канаді.

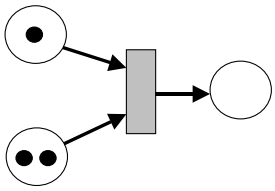
Елементами сітки Петрі є позиція, перехід, дуга й фішка, котрі мають зображене нижче графічне подання:

Дуги поєднують вершини протилежних типів: позицію з переходом та перехід з позицією; фішки перебувають усередині позицій і переміщуються сіткою внаслідок спрацьовування переходів.

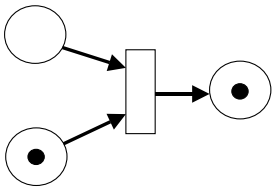
Перехід дозволено (збуджено), якщо всі його вхідні позиції мають фішки:

Перехід дозволено Перехід не дозволено

Спрацьовує довільний перехід з множини дозволених. При спрацьовуванні перехід вилучає фішки зі своїх вхідних позицій і розміщує фішки у свої вихідні позиції. Спрацьовування переходу відбувається миттєво.



Після спрацьовування



З множини дозволених спрацьовує лише один перехід, обираний довільно. Це зумовлює недетермінований характер поводження сітки. Отже, сітка Петрі описує безліч різних припустимих варіантів поводження модельованих систем та процесів. Відзначимо також, що недермінованість поводження може подавати як паралелізм, так і конфлікти (альтернативи). Приклад сітки Петрі зображено на рис. 2.1.

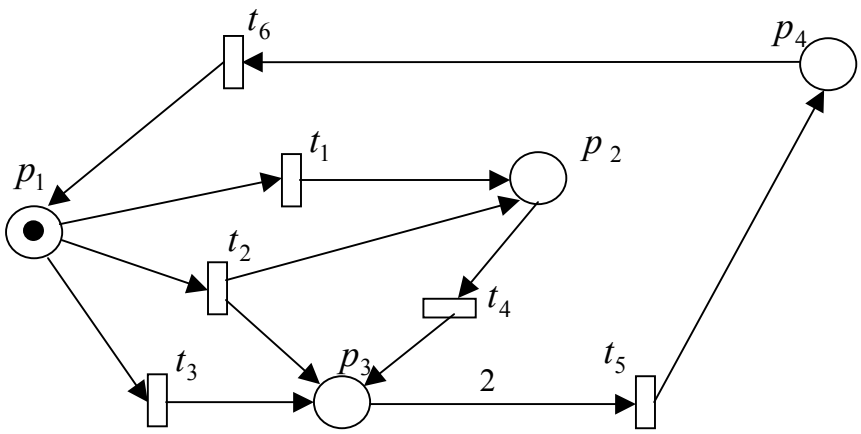
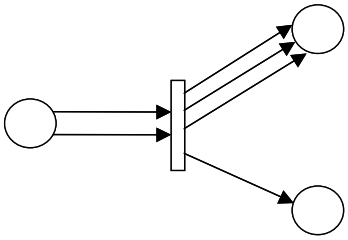
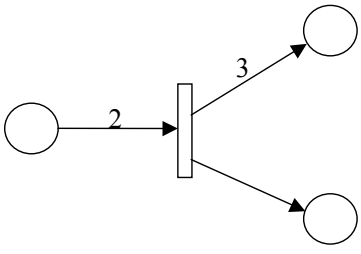


Рисунок 2.1 – Сітка Петрі N_1

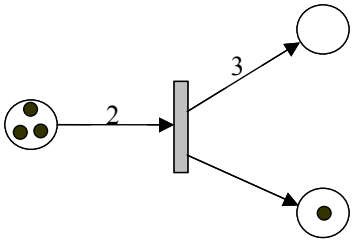
Для моделювання реальних об'єктів буває зручно використовувати кратні дуги, для яких умова спрацьовування має виконуватися по кожному екземплярові дуги. Графічно, як правило, зображують одну дугу, надписуючи над нею її кратність:



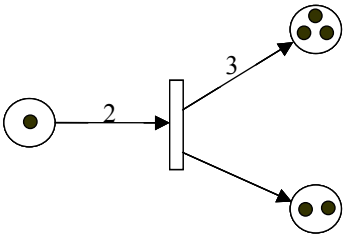
Зображують



Спрацьовування переходу відбувається в такий спосіб:



Після спрацьовування



Процес функціонування сітки Петрі може бути наочно подано графом досяжних станів. Стан сітки однозначно визначається її маркуванням – розподілом фішок по позиціях. Вершинами графа є припустимі маркування сітки Петрі, дуги позначено символом опрацьовуваного переходу. Дуга будується для кожного збудженого переходу. Побудова припиняється, коли ми одержуємо маркування, в якому не збуджено жодного з переходів, або маркування, котре містяться в графі. Відзначимо, що граф досяжних маркувань являє собою автомат. На рис. 2.2 подано граф досяжних маркувань сітки N_1, зображеної на рис. 2.1.

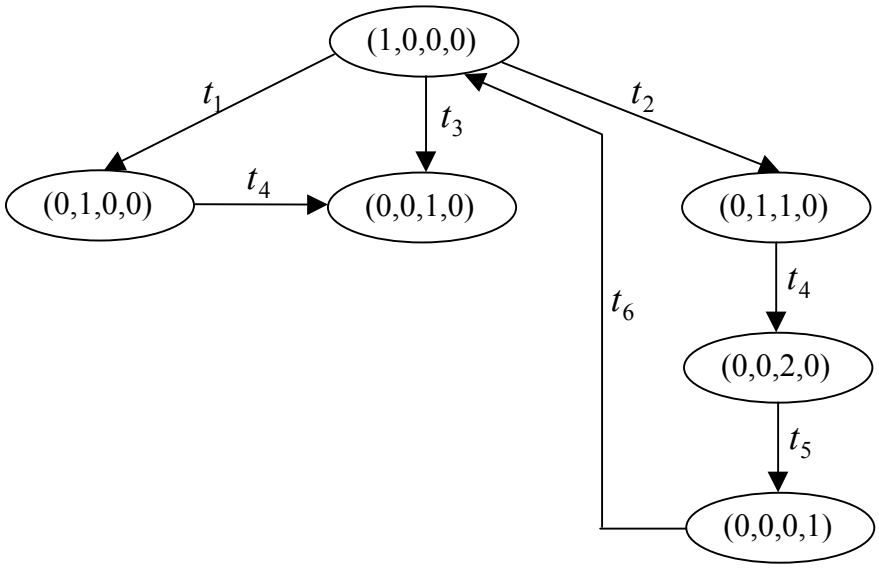


Рисунок 2.2 – Граф досяжних маркувань сітки N_1

Розглянемо приклади конкретних моделей, подані сітками Петрі. У моделі обчислювальної системи, зображеної на рис. 2.3, позиція p_6 моделює процесор, позиція p_9 – накопичувач на магнітних дисках, позиції p_1 та p_5 – вхідну й вихідну черги завдань відповідно; перехід t_4 відповідає завершенню кванта часу.

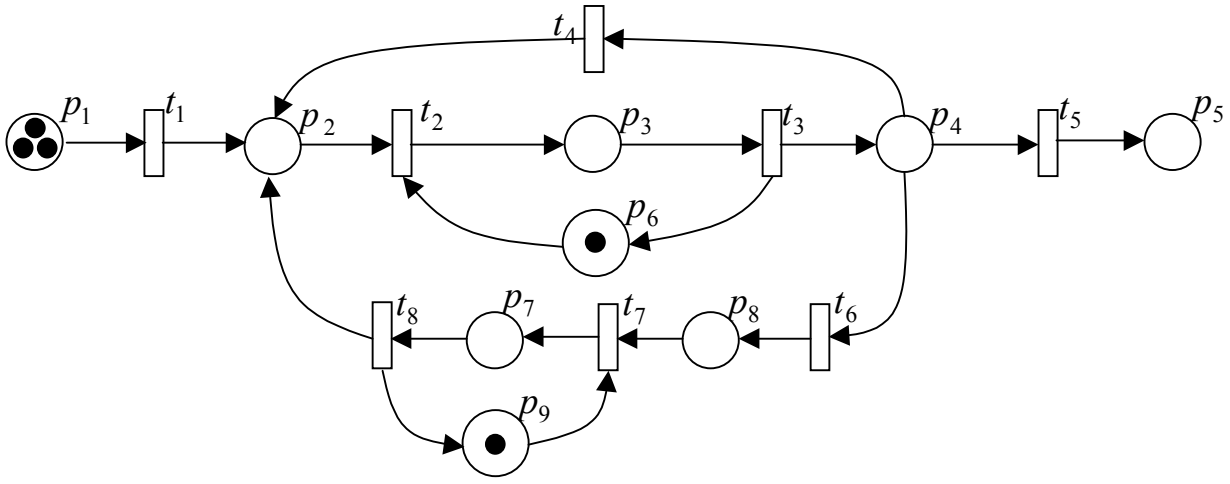


Рисунок 2.3 – Модель обчислювальної системи

У нижченаведеному прикладі сітку Петрі буде використано для верифікації простого протоколу передавання повідомлень.

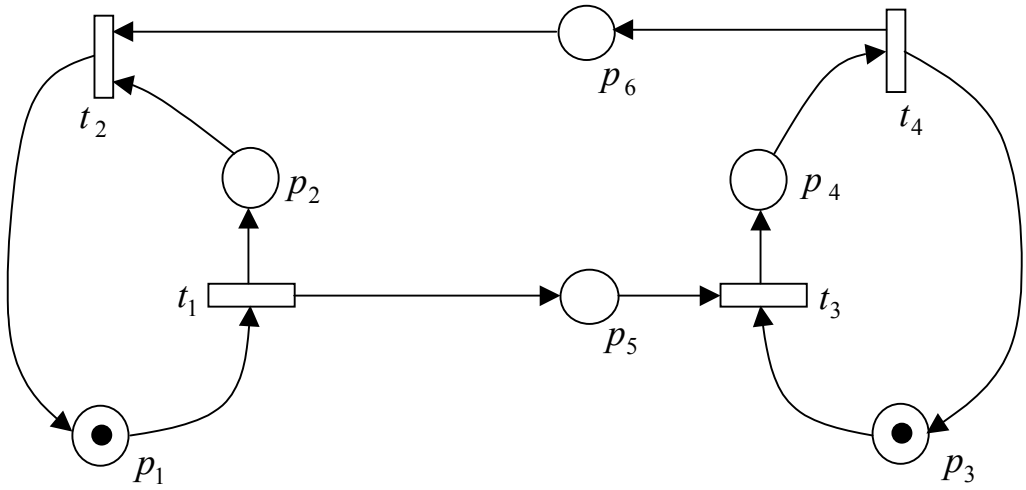


Рисунок 2.4 – Протокол одностороннього передавання повідомлень

В сітці, зображеній на рис. 2.4, перехід t_1 моделює надсилання повідомлень, перехід t_3 – одержання повідомлень, перехід t_4 – надсилання потвердження, перехід t_2 – одержання потвердження. Штатний цикл: $t_1 t_3 t_4 t_2$.

Розглянемо симетричний випадок, поданий на рис. 2.5.

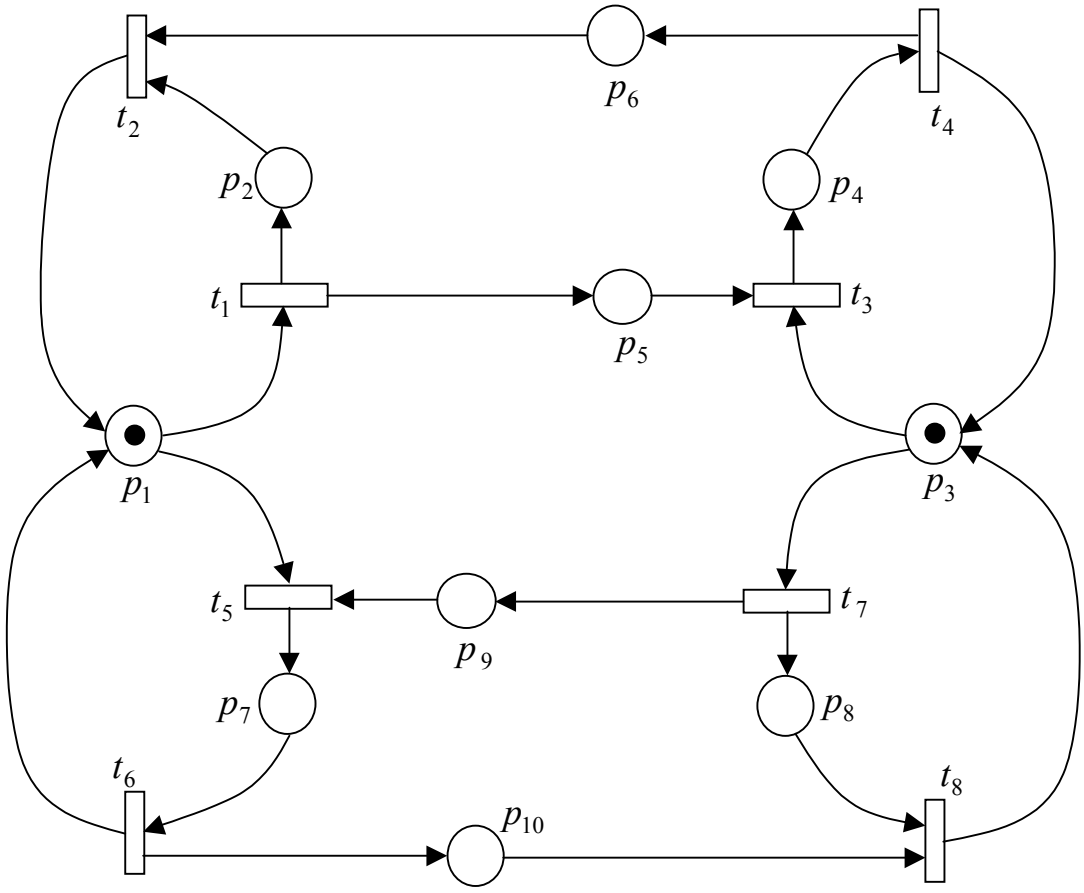


Рисунок 2.5 – Протокол двостороннього передавання повідомлень

За імітування динаміки сітки визначаємо, що можливий тупик, наприклад, внаслідок запуску послідовності $t_1 t_7$ чи $t_7 t_1$. Отже, протокол не слід застосовувати для передавання інформації, а треба змодифікувати. Найпростіша з модифікацій полягає в доповненні протоколу засобами керування доступом до каналу, наприклад за допомогою семафорної позиції p_{11} (рис. 2.6).

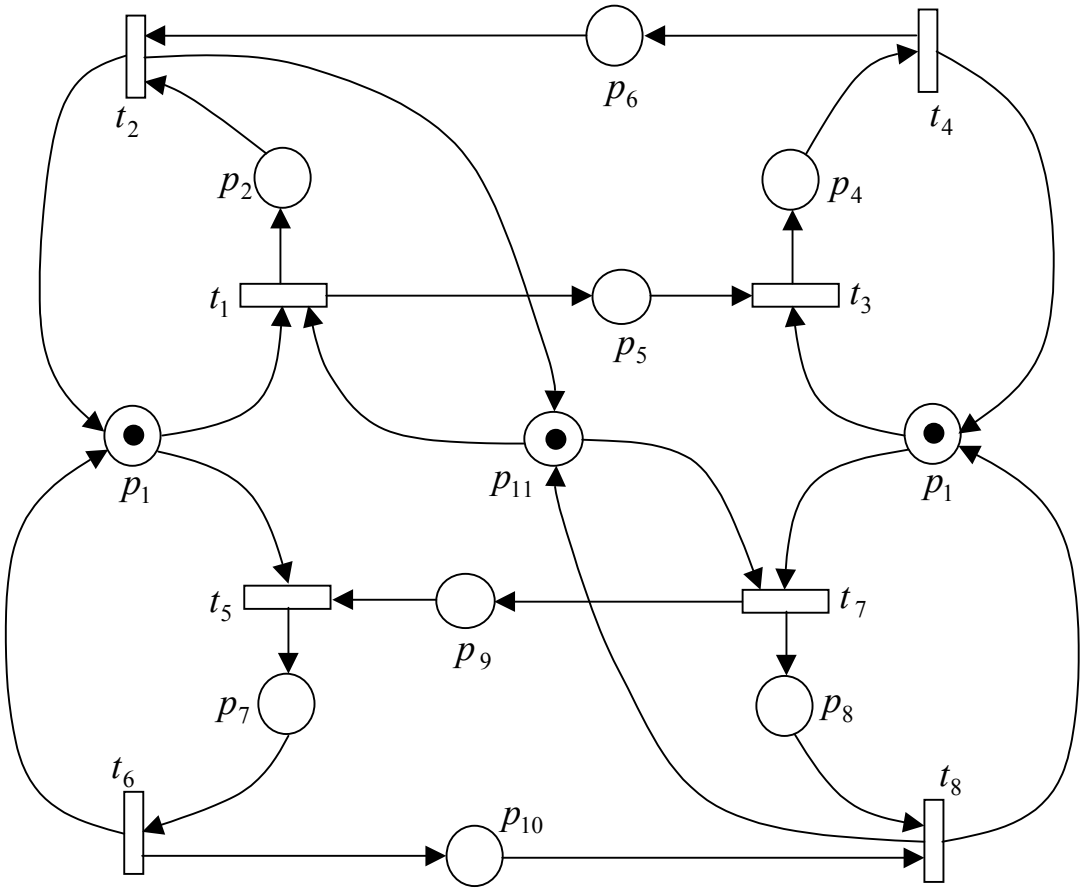
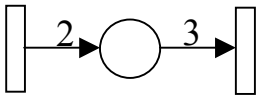


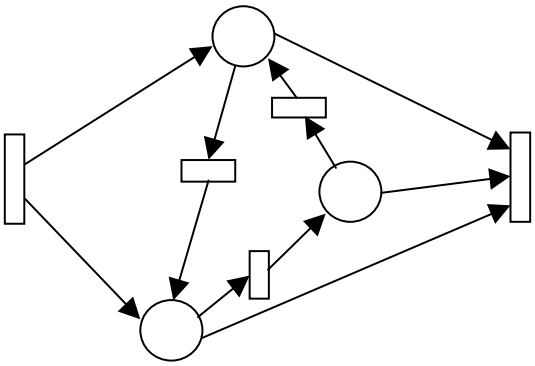
Рисунок 2.6 – Коректний протокол двостороннього передавання повідомлень

Побудова діаграми станів дозволяє дійти висновку, що сітка не містить тупиків; отже, протокол є коректним і його може бути використано для взаємодії систем.

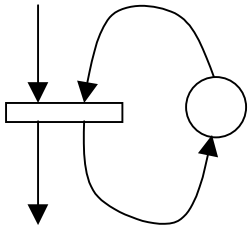
У теоретичних викладеннях, не обмежуючи загальності, можна розглядати сітки без кратних дуг, тому що сітку з кратними дугами може бути перетворено на сітку без кратних дуг шляхом заміни кожної позиції, якій інцидентні є кратні дуги, циклом з k позицій та k переходів, де k – максимальна кратність інцидентної дуги. Наприклад:



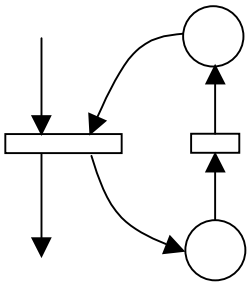
Подаємо



В аналогічний спосіб сітки, що вони містять петлі, може бути подано як сітки без петель:



Подаємо



Сітки без петель і кратних дуг називатимемо ординарними.

2.2 Рівняння станів і властивості сіток Петрі

Зробимо формальне визначення введених у попередньому підрозділі сіток Петрі. *Графом сітки Петрі* називатимемо трійку $G = (P,\ T,\ A)$, де $P = \{p\}$ – скінченна множина позицій; $T = \{t\}$ – скінченна множина переходів; відображення $A : P \times T \bigcup T \times P \to \mathrm{N}_0$, де $\mathrm{N}_0 = \mathrm{N} \bigcup \{0\}$, задає дуги та їхню кратність. *Маркуванням* сітки називатимемо відбиття $\mu : P \to \mathrm{N}_0$, що задає розподіл фішок поза позиціями. Тоді *сітка Петрі* це $N = (G,\ \mu_0)$, або $N = (P,\ T,\ A,\ \mu_0)$, де μ_0 – початкове маркування.

Побудуємо формальне подання сітки, зображеної на рис. 2.1:

$$N_1 = (P,\ T,\ A),\ P = \{p_1, p_2, p_3, p_4\},\ T = \{t_1, t_2, t_3, t_4, t_5, t_6\},$$
$$A = \{(p_1,t_1,1),\ (p_1,t_2,1),\ (p_1,t_3,1),\ (p_2,t_4,1),\ (p_3,t_5,2),\ (p_4,t_6,1),$$
$$(t_1,p_2,1),\ (t_2,p_2,1),\ (t_2,p_3,1),\ (t_3,p_3,1),\ (t_4,p_3,1),\ (t_5,p_4,1),\ (t_6,p_1,1)\},$$
$$\mu_0 = \{(p_1,1)\}.$$

Зауважимо, що для стислості зазначено лише ненульові значення функцій.

Уведемо спеціальні позначення для вхідних, вихідних та інцидентних дуг позиції (переходу) сітки:

$$^{\bullet}p = \{t \,|\, A(t,p) > 0\}\ ,\ p^{\bullet} = \{t \,|\, A(p,t) > 0\}\ ,\ ^{\bullet}p^{\bullet} = {}^{\bullet}p \cup p^{\bullet};$$
$$^{\bullet}t = \{p \,|\, A(p,t) > 0\}\ ,\ t^{\bullet} = \{p \,|\, A(t,p) > 0\}\ ,\ ^{\bullet}t^{\bullet} = {}^{\bullet}t \cup t^{\bullet}.$$

Поряд з раніше розглянутими графічним та теоретико-множинним визначеннями сітки Петрі широко використовується матричне подання сітки. Занумеруємо позиції й переходи $P = \{p_i \,|\, i = \overline{1,m}\}$, $T = \{t_j \,|\, j = \overline{1,n}\}$ у такий спосіб, що $|P| = m$, $|T| = n$. Відношення A подамо матрицями B й D або, для сіток без петель, матрицею C.

$$B = \|b_{i,j}\|,\ \ i = \overline{1,m},\ \ j = \overline{1,n},\ \ b_{i,j} = A(p_i, t_j).$$
$$D = \|d_{i,j}\|,\ \ i = \overline{1,m},\ \ j = \overline{1,n},\ \ b_{i,j} = A(t_j, p_i).$$
$$C = D - B.$$

Маркування сітки подаватимемо вектором-рядком:

$$\overline{\mu} = (\mu(p_1),\ \mu(p_2),\ ...,\ \mu(p_m))\ .$$

Тоді сітку Петрі однозначно подає набір $(B,\ D,\ \overline{\mu}_0)$, або, в разі сіток без петель – $(C,\ \overline{\mu}_0)$.

Побудуємо матричне подання для сітки N_1, зображеної на рис. 2.1:

$$N_1 = (B,\ D,\ \overline{\mu}_0),$$
$$B = \begin{vmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}, \qquad D = \begin{vmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{vmatrix}, \qquad \mu_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}.$$

Оскільки сітка не містить петель, можливе подання:

$$N_1 = (C,\ \overline{\mu}_0),$$
$$C = \begin{vmatrix} -1 & -1 & -1 & 0 & 0 & 1 \\ 1 & 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 1 & 1 & -2 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{vmatrix}.$$

Для побудови рівняння станів сітки, що воно формально подає її динаміку, впровадимо допоміжні позначення.

Індикатор переходу – це вектор-стовпець $\overline{u}^{t_j} = \|u_k\|$, $u_k = \begin{cases} 1, & k = j, \\ 0, & k \neq j. \end{cases}$

Індикатор позиції – це вектор-рядок $\overline{u}^{p_i} = (u_k)$, $u_k = \begin{cases} 1, & k = i, \\ 0, & k \neq i. \end{cases}$

Тоді вхідні дуги переходу $t \in T$ можна подавати як $t^- = B\overline{u}^t$, а вихідні дуги переходу $t \in T$ – як $t^+ = D\overline{u}^t$. Аналогічно: входи позиції $p \in P$ – як $p^- = \overline{u}^p D$; виходи позиції $p \in P$ – як $p^+ = \overline{u}^p B$.

Розглянемо приклади для сітки N_1, зображеної на рис. 2.1.

$$\overline{u}^{t_2} = \begin{vmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{vmatrix}, \quad \overline{u}^{t_5} = \begin{vmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{vmatrix};$$

$$t_2^- = (1 \quad 0 \quad 0 \quad 0), \; t_5^- = (0 \quad 0 \quad 2 \quad 0), \; t_2^+ = (0 \quad 1 \quad 1 \quad 0), \; t_5^+ = (0 \quad 0 \quad 0 \quad 1).$$

Тоді умова збудження переходу $t \in T$ можна подати як $\overline{\mu} \geq B\overline{u}^t$, а спрацьовування переходу $t \in T$ – як $\overline{\mu}' = \overline{\mu} - B\overline{u}^t + D\overline{u}^t$. Отже, формальний опис процесу функціонування сітки Петрі подає система:

$$\begin{cases} \overline{\mu}^{k+1} = \overline{\mu}^k + C\overline{u}^t, \\ \overline{\mu} \geq B\overline{u}^t, \; t \in T, \\ \overline{\mu}^0 = \overline{\mu}_0. \end{cases}$$

Зауважимо, що система містить рівняння й нерівності; це відповідає недетермінованості поводження сітки Петрі. Побудовану систему традиційно називають *рівнянням станів* сітки Петрі.

Впровадимо фундаментальні характеристики поводження сітки Петрі. Для цього використовуємо такі допоміжні позначення: позначимо, що перехід $t \in T$ є збуджений (дозволений) у маркуванні $\overline{\mu}$, як $\overline{\mu} \overset{t}{\rightarrow}$; спрацьовування переходу $t \in T$ позначимо як $\overline{\mu} \overset{t}{\rightarrow} \overline{\mu}'$. Аналогічні позначення використовуватимемо також для послідовностей спрацьовування переходів $\sigma \in T^*$ ($\sigma = t_{j_1} t_{j_2} \ldots t_{j_k}$): послідовність $\sigma \in T^*$ дозволена в маркуванні $\overline{\mu} : \overline{\mu} \overset{\sigma}{\rightarrow}$; послідовність $\sigma \in T^*$ спрацьовує в маркуванні $\overline{\mu} : \overline{\mu} \overset{\sigma}{\rightarrow} \overline{\mu}'$. Фундаментальними характеристиками поводження сітки Петрі є *вільна*

мова сітки $L(N) = \{\sigma \mid \overline{\mu}_0 \overset{\sigma}{\rightarrow}\}$ та *множина досяжних маркувань* сітки $R(N) = \{\overline{\mu} \mid \exists \sigma \in T^* : \overline{\mu}_0 \overset{\sigma}{\rightarrow} \overline{\mu}\}$.

Проілюструємо введені поняття для сітки N_1, зображеної на рис. 2.1.

Нехай $\overline{\mu} = (0,1,1,0)$, $\overline{\mu}' = (0,0,0,1)$, $\sigma = t_4 t_5$. Тоді виконується $\overline{\mu} \overset{\sigma}{\rightarrow}$ та $\overline{\mu} \overset{\sigma}{\rightarrow} \overline{\mu}'$. Побудуємо множину досяжних маркувань і вільну мову сітки N_1:

$$R(N_1) = \{(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,1,1,0),(0,0,2,0),(0,0,0,1)\},$$
$$L(N_1) = \{t_1, t_3, t_2, t_1 t_4, t_2 t_4, t_2 t_4 t_5, t_2 t_4 t_5 t_6, t_2 t_4 t_5 t_6 t_2, t_2 t_4 t_5 t_6 t_2 t_4, \ldots\}.$$

Набір основних властивостей сіток Петрі сформувався як набір характеристик, притаманних досліджуваному об'єктові, корисних чи небажаних (котрих слід уникати). Тому властивості можна умовно поділити на позитивні і негативні. Необхідність побудови методів для визначення, чи має сітка шукану властивість, формулюють у вигляді задач (проблем). Іноді справедливо зворотне: у виді властивості формулюють певну важливу проблему дослідження сіток. Розрізнюють поведінкові й структурні властивості сіток. Поведінкові властивості пов'язано з конкретним початковим маркуванням. Структурні властивості виконуються за будь-якого початкового маркування.

Перелічимо основні властивості сіток Петрі:

- *Еквівалентність*: для двох довільних сіток Петрі – N_1 та N_2 – визначити, чи збігаються їхні вільні мови: $L(N_1) = L(N_2)$.
- *Включення*: для двох довільних сіток Петрі – N_1 та N_2 – визначити, чи виконується умова $L(N_1) \subseteq L(N_2)$. Проблеми еквівалентності й включення може бути також сформульовано для множини досяжних маркувань.
- *Досяжність* маркування: для заданої сітки Петрі N й маркування $\overline{\mu}$ визначити чи виконується умова $\overline{\mu} \in R(N)$.
- *Покриваність*: для заданої сітки Петрі N й маркування $\overline{\mu}$ визначити чи містить $R(N)$ певне маркірування $\overline{\mu}'$, таке що $\overline{\mu}' \geq \overline{\mu}$.
- *Потенційна живість*: перехід $t \in T$ є потенційно живий, якщо існує дозволена послідовність спрацьовувань, котра містить цей перехід: $\exists \sigma \in T^* : \overline{\mu}_0 \overset{\sigma}{\rightarrow} \& \; t \in \sigma$.
- *Живість*: перехід є живий, якщо він є потенційно живий у будь-якому досяжному маркуванні; отже $t \in T$ живий, якщо $\forall \overline{\mu} \in R(N), \exists \sigma \in T^* : \overline{\mu} \overset{\sigma}{\rightarrow} \& \; t \in \sigma$. Сітка Петрі є жива, якщо живі усі її переходи.
- *Тупик*: маркування $\overline{\mu}$ сітки називають t-тупиковим, якщо перехід $t \in T$ не є в ній потенційно живим; маркування $\overline{\mu}$ сітки називають тупиковим, якщо воно є t-тупикове для всіх переходів сітки. Іноді розглядають таку властивість як *безтупиковість*, що означає відсутність у сітки тупиків. Ця властивість є більш слабкою, аніж живість. Сітка хоча й може функціонувати нескінченно, але при цьому

певні переходи не може бути запущено. Тому безтупиковість часто розглядають як негативну властивість.

- *Обмеженість*: позиція $p \in P$ є обмежена (*l-обмежена*), якщо існує таке ціле число *l*, що маркування позиції не перевищує його: $\forall \overline{\mu} \in R(N) : \mu(p) \le l$. Сітка є обмежена, якщо обмежено усі її позиції.
- *Безпечність*: безпечною називають 1-обмежену сітку.
- *Консервативність*: консервативною називають сітку, що вона зберігає зважену суму фішек стосовно певного вагового вектора $\overline{w}$ з натуральними компонентами: $\overline{w}\overline{\mu} = \overline{w}\overline{\mu}_0 = const$. Строго консервативною називають сітку, сума маркерів якої є стала, тобто консервативну сітку стосовно одиничного вагового вектора.
- *Повторюваність*: послідовність запускання переходів σ є повторювана, якщо її може бути запущено довільну кількість разів; тобто з $\overline{\mu} \xrightarrow{\sigma}$ випливає $\overline{\mu} \xrightarrow{\sigma^*}$.
- *Стаціонарна повторюваність*: послідовність запускання переходів σ є стаціонарно повторювана, якщо вона є повторювана і приводить сітку до вихідного маркування: $\overline{\mu} \xrightarrow{\sigma} \overline{\mu}$.
- *Стійкість*: сітка є стійка, якщо для двох будь-яких дозволених переходів запускання одного з них не призводить до заборони спрацьовування іншого: $\forall t, t' \in T, \forall \overline{\mu} \in R(N) : (\overline{\mu} \xrightarrow{t} \overline{\mu}^1 \ \& \ \overline{\mu} \xrightarrow{t'} \overline{\mu}^2) \Rightarrow (\overline{\mu}^1 \xrightarrow{t} \& \ \overline{\mu}^2 \xrightarrow{t'})$.
- *Оборотність*: сітка є оборотня, якщо за $\forall \mu \in R(G, \overline{\mu}_0)$ виконується $\mu_0 \in R(G, \overline{\mu})$.
- *Стан приймання*: маркування називають станом приймання (базовим станом), якщо воно є досяжне з будь-якого досяжного в сітці маркування.

Зазначимо властивості сітки N_1, зображеної на рис. 2.1:

- маркування (0,1,1,0) та (0,0,1,0) є досяжні в сітці, а маркування (0,1,1,1) є недосяжне;
- сітка є потенційно жива, тому що кожний з переходів може бути запущено, принаймні, одноразово; зазначимо відповідні дозволені послідовності запускання переходів – t_1, t_2, t_3, $t_1 t_4$;
- сітка не є жива, тому що в ній є досяжне тупикове маркування (0,0,1,0);
- сітка є обмежена; зазначимо відповідні мінімальні обмеження – (1,1,2,1), з яких випливає, що сітка є небезпечна;
- сітка не є консервативна, оскільки система $\overline{w}\overline{\mu} = \overline{w}\overline{\mu}_0$ є нерозв'язна в цілих невід'ємних числах;
- послідовність $t_2 t_4 t_5 t_6$ є стаціонарно повторюваною;
- сітка не є стійка; дійсно, запускання переходу t_1 в початковому маркуванні унеможливлює спрацьовування переходу t_2, дозволеного в цьому маркуванні;
- сітка є необоротня, наприклад через те що початкове маркування є недосяжне з маркування сітки (0,0,1,0); тому сітка також не має стану приймання.

Слід зазначити, що проблеми еквівалентності та включення є алгоритмічно нерозв'язні для базових сіток Петрі. Доведено еквівалентність проблем досяжності й живості, а також їхню експоненційну складність по пам'яті.

У наступних підрозділах подано основні методи аналізу властивостей сіток Петрі.

2.3 Структурний аналіз сіток Петрі

Рівняння станів сітки Петрі хоча і є повним формальним поданням динаміки сітки, містить нерівності й є рекурентним. Подамо маркування в довільному такті *k*, дістане внаслідок спрацьовування дозволеної послідовності σ, $\overline{\mu}_0 \overset{\sigma}{\to} \overline{\mu}$, через початкове маркування:

$$\overline{\mu} = \overline{\mu}^k = \overline{\mu}_0 + C \cdot \sum_{t \in \sigma} \overline{u}^t \quad .$$

Вектор $\overline{\sigma} = \sum_{t \in \sigma} \overline{u}^t$ називатимемо *вектором рахування спрацьовувань* послідовності σ, а рівняння

$$\overline{\mu} = \overline{\mu}_0 + C\overline{\sigma}$$

– *фундаментальним рівнянням* сітки Петрі.

При побудові фундаментального рівняння відбувається втрата інформації. Однак фундаментальне рівняння є лінійне, що є зручно для винайдення його розв'язків. Можливість розв'язання фундаментального рівняння в цілих невід'ємних числах є необхідною умовою досяжності маркування: якщо маркування $\overline{\mu}$ є досяжне в сітці, то фундаментальне рівняння має цілий невід'ємний розв'язок $\overline{\sigma}$, такий що $\overline{\mu}_0 \overset{\sigma}{\to} \overline{\mu}$.

Іноді зручно також використовувати множини $\overline{L}(N) = \{\sigma \mid \overline{\mu}_0 + C\overline{\sigma} \geq 0\}$, $\overline{R}(N) = \{\mu \mid \mu = \overline{\mu}_0 + C\overline{\sigma}, \overline{\sigma} \geq 0\}$. За побудовою маємо $L(N) \subseteq \overline{L}(N)$, $R(N) \subseteq \overline{R}(N)$.

Інваріантом позицій сітки, чи то *p-інваріантом*, називатимемо невід'ємний цілий розв'язок $\overline{x}$ рівняння

$$\overline{x}C = 0 .$$

Інваріантом переходів сітки, чи то *t-інваріантом*, називатимемо невід'ємний цілий розв'язок $\overline{y}$ рівняння

$$C\overline{y} = 0 .$$

Якщо існує відповідний інваріант, усі компоненти якого є натуральні, то сітку називають *p*- або *t*- інваріантною.

Помножимо фундаментальне рівняння сітки ліворуч на інваріант позицій

$$\bar{x} \cdot \bar{\mu} = \bar{x} \cdot \bar{\mu}_0 + \bar{x} \cdot C\bar{\sigma}$$

і, внаслідок властивостей інваріанта, дістанемо

$$\bar{x} \cdot \bar{\mu} = \bar{x} \cdot \bar{\mu}_0 = const\,.$$

Отже, зважена сума маркерів *p*-інваріантної сітки є стала. Ця властивість виконується за будь-якого початкового маркування й, отже, є структурною.

Доволі просто довести, що *p*-інваріантна сітка є обмежена; для цього відокремимо компоненту для довільної позиції $q \in P$:

$$x_q \cdot \mu(q) + \sum_{p \neq q} x_p \cdot \mu(p) = \bar{x} \cdot \bar{\mu}_0\,.$$

Зважаючи на невід'ємність сум, запишемо

$$x_q \cdot \mu(q) \leq \bar{x} \cdot \bar{\mu}_0\,.$$

І далі

$$\mu(q) \leq \frac{\bar{x} \cdot \bar{\mu}_0}{x_q}\,.$$

Основну властивість *t*-інваріанта легко помітити, використовуючи інваріант як вектор рахування спрацьовувань:

$$\bar{\mu} = \bar{\mu}_0\,.$$

Отже, *t*-інваріант подає стаціонарно повторювані послідовності спрацьовувань переходів.

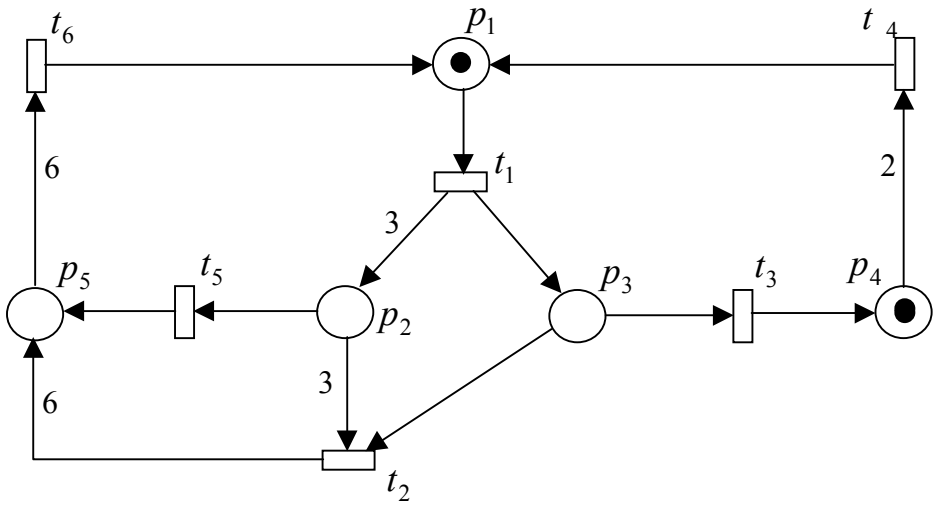


Рисунок 2.7 – сітка Петрі N_2

Віднайдемо інваріанти для сітки N_2, зображеної на рис. 2.7:

- інваріанти позицій: $x = k_1 \cdot (6,1,3,3,1)$;
- інваріанти переходів: $y = k_1 \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} + k_2 \cdot \begin{bmatrix} 2 \\ 0 \\ 2 \\ 1 \\ 6 \\ 1 \end{bmatrix}$, $y = \begin{bmatrix} 3 \\ 1 \\ 2 \\ 1 \\ 6 \\ 2 \end{bmatrix}$.

Дійсно, сітка N_2 є обмежена і має стаціонарно повторювану послідовність спрацьовування переходів $\sigma = t_1 t_2 t_6 t_1 t_3 t_4 t_1 t_5^6 t_3 t_6$; зважена сума маркерів дорівнює 9.

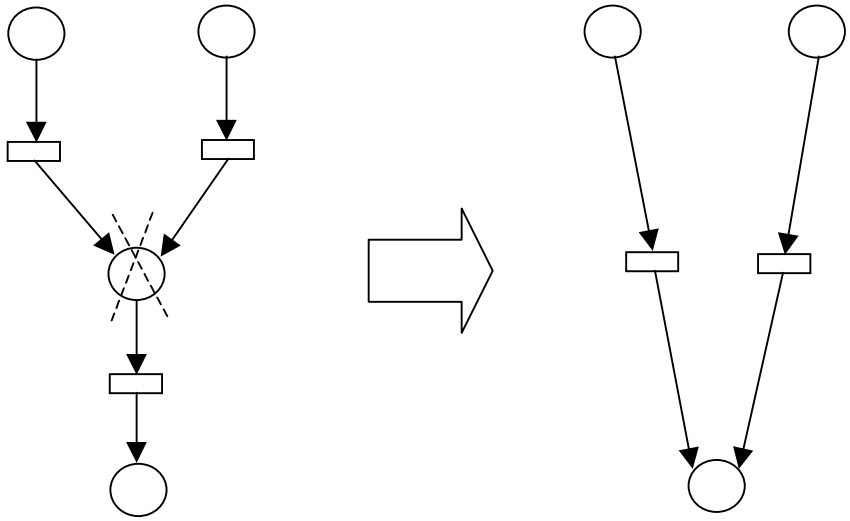
Редукція – це окремий випадок еквівалентних перетворювань, що знижують розмірність сітки. Як критерій еквівалентності виступає набір властивостей сітки, тобто дві сітки вважаємо за еквівалентні, якщо вони мають однаковий набір властивостей. Як правило, розглядають немарковані сітки, тобто досліджуються їхні структурні властивості.

Нехай $G' = (P',\ T',\ A')$ дістана із $G = (P,\ T,\ A)$ шляхом редукції стосовно набору властивостей $Z = \{z_1, z_2, ..., z_l\}$. Тоді $|P' \cup T'| \le |P \cup T|$ й $z_i(G) \Leftrightarrow z_i(G')$. Найбільш вивченою є *LBS* редукція сіток, яка зберігає живість, обмеженість та консервативність. Ці властивості є важливі для моделей багатьох реальних систем: транспортних систем, протоколів обміну інформацією, паралельних програм. Впровадимо множину перетворювань сіток Петрі $R = \{R_1,\ R_2,\ R_3\}$. Опишемо кожне із зазначених перетворювань.

R_1 – підстановка позиції.

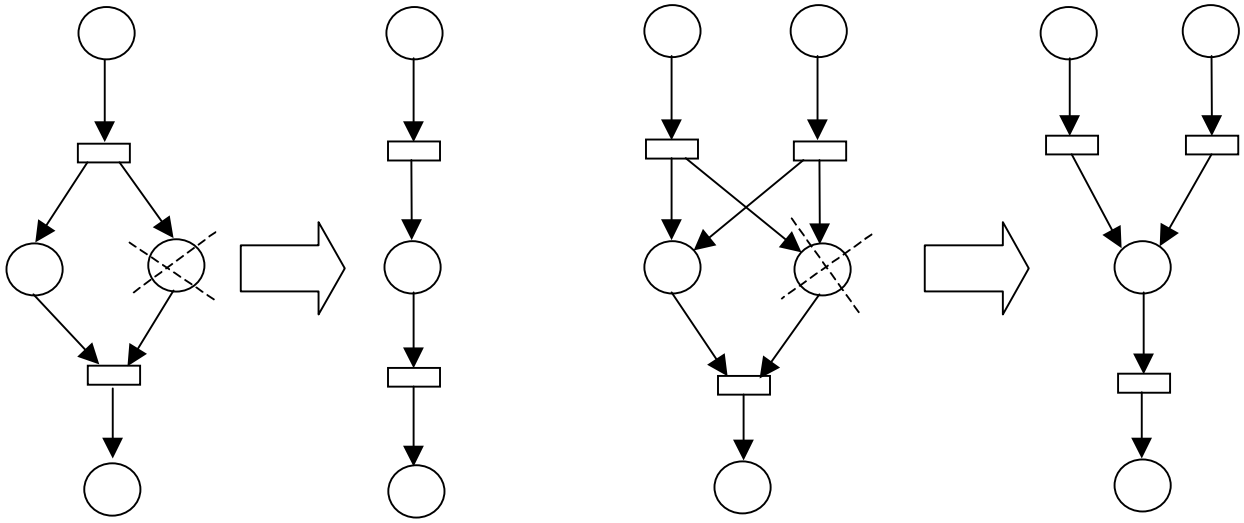
Позиція $p \in P$ є підстановочна, якщо вона є єдиною вхідною позицією для всіх переходів $t \in p^\bullet$. Підстановочна позиція вилучається разом з усіма переходами

$t \in {}^{\bullet}p$, $t \in p^{\bullet}$; додаються нові переходи й дуги, що вони зберігають співвідношення маркувань.



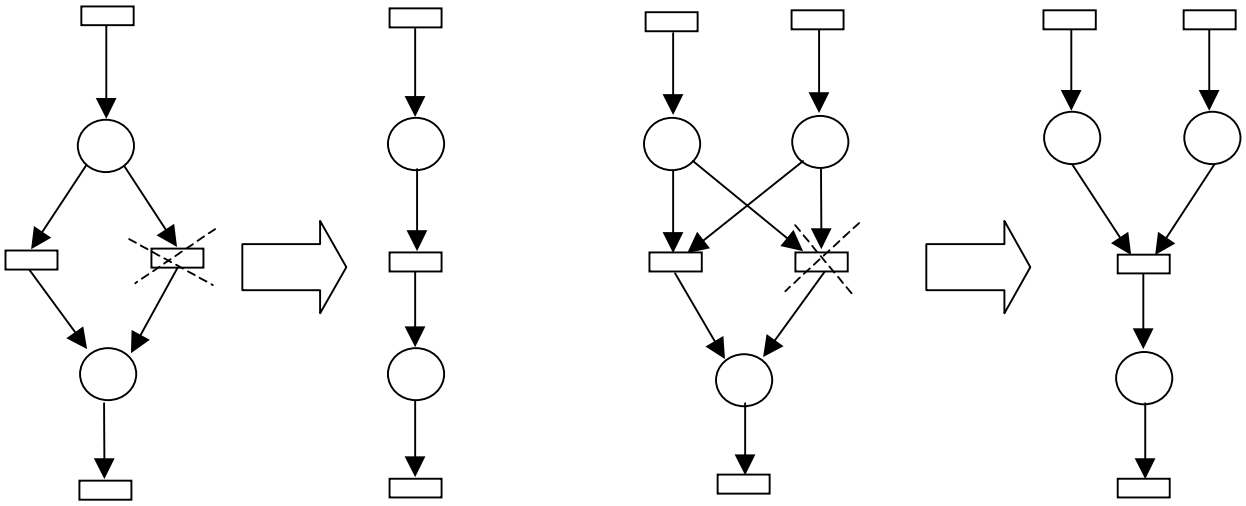
R_2 – вилучення надлишкової позиції.

Позиція $p \in P$ є структурно надлишкова в сітці, якщо за всіх досяжних маркувань нестача маркерів у позиції p не може стати причиною того, що перехід $t \in p^{\bullet}$ буде незбуджений.



R_3 – вилучення надлишкового переходу.

Перехід $t \in T$ є структурно надлишковий, якщо будь-яке маркування, що його збуджує, збуджує також і певну послідовність спрацьовування переходів, яка не утримує t, але зводить до того ж самого результату.



Впроваджені перетворювання можна використовувати як набір графічних шаблонів або як універсальні правила, застосовувані до різних фрагментів сітки. Слід також зазначити, що графічним перетворюванням відповідають алгебричні перетворення рівняння станів.

Дослідимо властивості сітки, зображеної на рис. 2.8, за допомогою редукції:

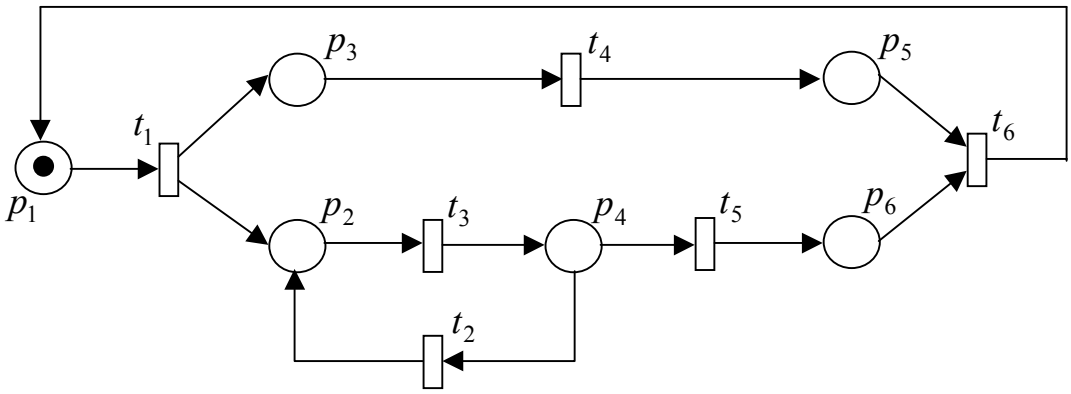
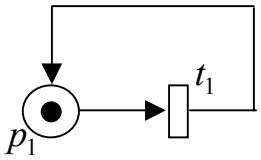


Рисунок 2.8 – Схема паралельної програми

За допомогою ланцюжка перетворювань $R_1(p_3)R_1(p_2)R_3(t_2)R_1(p_4)R_2(p_5)R_1(p_6)$ сітка приводиться до вигляду



Дістана сітка є жива, обмежена, безпечна, тобто усі ці властивості є притаманні й вихідній сітці.

2.4 Граф покривних маркувань

Як було зазначено раніше, граф досяжних маркувань сітки Петрі в загальному випадку є нескінченний. Тому для дослідження поведінкових властивостей сітки необхідні спеціальні методи. Доведено, що досліджуване в поточному розділі дерево (граф) покривних маркувань є скінченне для будь-якої сітки Петрі.

Впровадимо псевдочисло ω з такими властивостями: псевдочисло є більше за будь-яке натуральне число і не змінюється при додаванні до нього довільного натурального числа: $\forall n \in \mathrm{N} : \omega > n,\ \omega + n = \omega,\ \omega - n = \omega$. Нехай $\mathrm{N}^{\omega} = \mathrm{N} \bigcup \{\omega\}$. Впровадимо псевдомаркування як $\widetilde{\mu} : P \to \mathrm{N}^{\omega}$.

Розглянемо *алгоритм побудови дерева покривних маркувань* $\widetilde{R}(N)$, що використовує псевдомаркування сітки:

Крок 0. Нехай $\overline{\mu}_0$ – *гранична* вершина.

Крок 1. Оберемо довільну граничну вершину $\overline{\mu}$. Можливі є такі варіанти:

- якщо $\overline{\mu}$ є тупикова, то позначаємо $\overline{\mu}$ *термінальною* вершиною;
- якщо існує вершина $\overline{\mu}'$, що не є граничною, така що $\overline{\mu}' = \overline{\mu}$, то позначаємо $\overline{\mu}$ *дублювальною* вершиною;
- якщо на шляху з $\overline{\mu}$ у коріння дерева існує таке маркування $\overline{\mu}'$, що $\overline{\mu} \geq \overline{\mu}'$, то для всіх позицій, таких що $\mu(p) > \mu'(p)$, додаємо $\mu(p) = \omega$;
- для кожного збудженого в $\overline{\mu}$ переходу $t \in T$ додаємо до дерева нову граничну вершину $\overline{\mu}'$, таку що $\overline{\mu} \xrightarrow{t} \overline{\mu}'$ й позначаємо дугу, що веде з вершини $\overline{\mu}$ у вершину $\overline{\mu}'$ символом переходу *t*; вершину $\overline{\mu}$ позначаємо як внутрішню.

Крок 2. Якщо є нові граничні вершини, повторити крок 1, інакше – кінець.

Якщо опрацьовану вершину, не відзначати як дублювальну, а спрямовувати дугу в раніш побудовану вершину, то дістанемо граф покривних маркувань; можлива також побудова допоміжних дуг при долучанні в маркування символу ω.

Розглянемо сітку N_3, зображену на рис. 2.9:

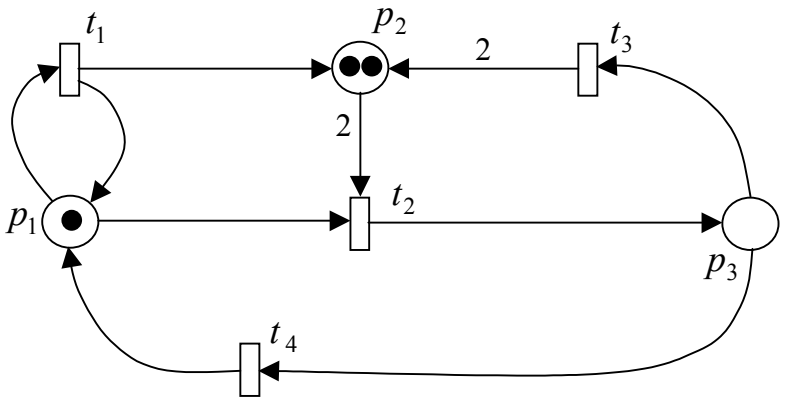


Рисунок 2.9 – Сітка Петрі N_3

Побудуємо дерево покривних маркувань для сітки N_3. При зображенні дерева (рис. 2.10) виділимо графічно дублювальні й термінальні вершини.

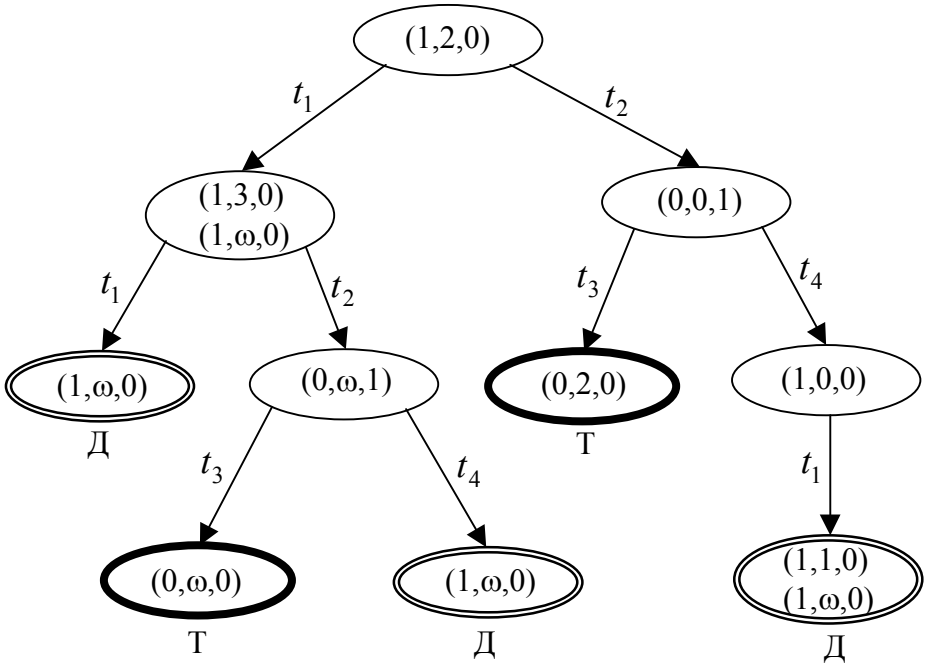


Рисунок 2.10 – Дерево покривних маркувань сітки N_3

Дерево покривних маркувань дозволяє досліджувати багато властивостей сіток Петрі. Насправді, сітка Петрі є обмежена тоді й лише тоді, коли її дерево покривних маркувань не містить символу ω; сітка Петрі є безпечна тоді й лише тоді, коли маркування вершин дерева покривних маркувань містять лише цифри 0 та 1; перехід є пасивний (не є живий) коли його символом не позначено жодної з дуг дерева; якщо маркування $\overline{\mu}$ є досяжне в сітці, то дерево покривних маркувань містить таке маркування $\overline{\mu}'$, що $\overline{\mu} \le \overline{\mu}'$; дерево покривних маркувань безтупикової сітки не містить термінальних вершин. Окрім того, дерево покривних маркувань дозволяє знайти повторювані й стаціонарно повторювані послідовності спрацьовувань переходів.

Дослідимо властивості сітки N_3 за допомогою дерева покривних маркувань, зображеного на рис. 2.10:

- сітка є необмежена, тому що дерево містить символи ω;
- сітка є потенційно жива, тому що має дуги, позначені символами кожного з переходів;
- сітка не є жива, тому що дерево містить термінальні (тупикові) вершини;
- маркування (0,7,1) є досяжне в сітці; насправді, дерево містить покривне маркування (0,ω,1) за яким може бути побудовано послідовність спрацьовування переходів $t_1^7 t_2$, що переводить сітку в зазначене маркування; маркування (0,1,7) є недосяжне в сітці, тому що дерево не містить для нього покривне маркування;
- послідовність спрацьовування t_1 є повторюваною;
- послідовність спрацьовування $t_2 t_4 t_1 t_1$ є стаціонарно повторюваною.

Отже, сітки Петрі є ефективним засобом аналізу й синтезу паралельних асинхронних систем, що вони мають низку важливих переваг, таких як наочне графічне подання, аналітичні й імітаційні методи дослідження, набір розширень базової моделі з високою зображувальною потужністю, можливість комплексного налаштовування моделі об'єкта й алгоритму керування, апаратну реалізацію процесорів сіток Петрі.

3 Машини Тюринга

3.1 Інтуїтивне поняття алгоритму

Використання інтуїтивного поняття алгоритму, вірніше, перше письмове згадування про його використання бере свій початок у роботах узбецького математика Мухаммада ібн Муси аль-Хорезмі й датується VIII сторіччям. Впродовж наступних дванадцяти сторіч алгоритми успішно застосовувалися в різних галузях діяльності людини. Однак до початку XX сторіччя сформувалася безліч задач, для розв'язання яких, за уявленнями вчених, не існує алгоритмів. Деякі з таких задач згадано у відомому списку проблем, поданому Гільбертом. Математичне доведення алгоритмічної нерозв'язності вимагало формалізування інтуїтивного поняття алгоритму. Машина Тюринга саме і є однією з таких формалізацій. З'явившись практично водночас з іншими моделями, такими як рекурсивні функції Кліні й нормальні алгоритми Маркова, вона дозволила довести, що для цілої низки задач не існує алгоритму їхнього розв'язання.

Інтуїтивно під *алгоритмом* розуміють точно описаний покроковий процес обчислення, в якому строго окреслено порядок виконання кроків і на кожнім кроці точно зазначено, як з вихідних даних здобути результат.

Класичним прикладом алгоритму є алгоритм Евкліда віднайдення найбільшого сумісного дільника двох натуральних чисел *a* та *b*. Алгоритм полягає в послідовному побудуванні різниці чисел доти, поки результат не збіжиться з меншим числом. Його може бути сформульовано у вигляді такої послідовності кроків:

Крок 0. Для визначеності призначимо: $a = \max(a,\ b)$, $b = \min(a,\ b)$.

Крок 1. Якщо $a = b$, то шуканий найбільший загальний дільник знайдено й він дорівнює, для визначеності, числу *a*.

Крок 2. Віднайдемо $c = a - b$.

Крок 3. Призначимо $a = \max(b,\ c)$, $b = \min(b,\ c)$. Переходимо до кроку 1.

Виконаємо алгоритм для пари чисел *275* та *150*:
a=275, b=150, c=125;
a=150, b=125, c=25;
a=125, b=25, c=100;
a=100, b=25, c=75;
a=75, b=25, c=50;
a=50, b=25, c=25;

a=b=25.

Отже, найбільший загальний дільник чисел *275* та *150* дорівнює *25*.

Обґрунтування алгоритму Евкліда міститься на відомому в математиці факті, що найбільший сумісний дільник двох чисел є також дільником їхньої різниці.

Сформулюємо більш докладно основні риси, притаманні алгоритмові:

1) Масовість. Полягає в тому, що алгоритм призначено для розв'язування не одиничної задачі (наприклад, 2+2=?), а цілої серії однотипних задач (x+y=?).

2) Детермінованість. Іншими словами, – визначеність, полягає в тім, що після виконання кожного з кроків ми точно знаємо, який крок слід виконувати наступним.

3) Конструктивність. Передбачається, що алгоритм завжди виконується за скінченну кількість кроків. Кроків може бути вельми багато, однак загальна їхня кількість завжди є скінченна. При побудові алгоритму ми виходимо з гіпотези потенційної втілювальності, вважаючи, що при виконанні алгоритму ми нічим не є обмежені: ані часом, ані доступними ресурсами.

4) Елементарність кроків. Кожен крок алгоритму має бути настільки простий, аби його можна було виконати однозначно точно.

Отже, ми розглянули інтуїтивне поняття алгоритму. Досліджувана далі машина Тюринга є однією з математичних моделей, запропонованих для формалізації цього поняття.

3.2. Опис машини Тюринга

Машина Тюринга (рис. 3.1) складається з нескінченної в обидва боки стрічки, розбитої на комірки й керуючої головки. Головка рухається уздовж стрічки. У певний момент часу головка оглядає одну з комірок стрічки. Головка читає символ у поточній комірці й, залежно від комбінації прочитаного символу й поточного стану, виробляє символ, що його записує до комірки й виконує одну з трьох дій: зсувається на одну комірку ліворуч, залишається на місці, зсувається на одну комірку праворуч. Окрім цього, головка переходить до наступного стану. Отже, елементарний крок машини Тюринга складається з читання, запису, руху й переходу до наступного стану. Загальна кількість станів та абетка символів стрічки є скінченні. Спеціально відокремлюють початковий і завершальний стани машини.

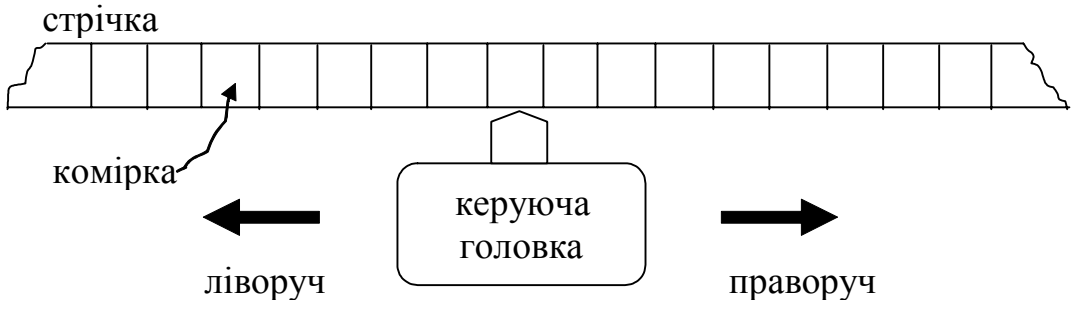


Рисунок 3.1 – Машина Тюринга

Передбачається, що кожного моменту часу лише в скінченній кількості комірок записано символи абетки, в решті комірок записано спеціальний порожній символ λ (ламбда). Найменшу частину стрічки, в якій містяться всі комірки з символами абетки, називатимемо *робочою зоною*.

Для зручності вважаємо, що на початку роботи машини й при її завершенні головка перебуває над самою лівою коміркою робочої зони. Спеціально зазначають початковий стан машини. Робота машини триває доти, поки вона не прийде до завершального стану.

Машину Тюринга, як і скінченний автомат, може бути подано діаграмою станів або матрицею переходів. Розглянемо зазначені способи подання на наступному прикладі.

Приклад 1. Побудувати машину Тюринга, котра виконує додавання натуральних чисел в унарній системі числення. В унарній системі числення число *n* подають відповідною кількістю паличок (одиниць), записаних на стрічці. Для поділу чисел використовуємо символ зірочки. Розглянемо початковий та завершальний стани стрічки для комбінації чисел 3+4=7:

а) На початку:

…	λ	λ	1	1	1	*	1	1	1	1	λ	λ	…

б) При завершенні:

…	λ	λ	λ	1	1	1	1	1	1	1	λ	λ	…

Оберемо такий алгоритм: зсуваємося праворуч до зірочки; замінюємо зірочку на паличку; зсуваємося ліворуч до ламбди; замінюємо крайню ліворуч паличку на ламбду; зупинка. Діаграму станів машини Тюринга подано на рис. 3.2. Таблиця 3.1 містить матричне подання машини.

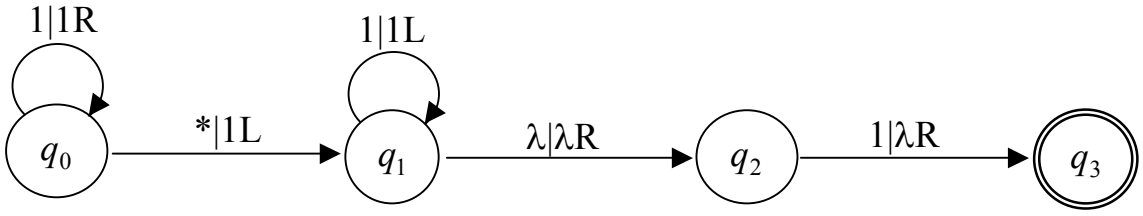


Рисунок 3.2 – Діаграма станів машини Тюринга для додавання чисел

Таблиця 3.1 – Матричне подання машини Тюринга для додавання чисел

Стан/символ	1	*	λ
q_0	1,R, q_0	1,L, q_1	-
q_1	1,L, q_1	-	λ,R, q_2
q_2	λ,S, q_3	-	-
q_3	-	-	-

Виконаємо формальне теоретико-множинне визначення машини Тюринга. Отже, *машина Тюринга* – це п'ятірка $M = (X,\ Q,\ q_0,\ q_f,\ P)$, де $X = \{x_i\}$ – скінченна абетка символів стрічки; $Q = \{q_j\}$ – скінченна множина станів головки; q_0 та q_f – початковий і завершальний стани відповідно; $P : Q \times X \to X \times R \times Q$ – програма машини; $R = \{L, S, R\}$ – припустимі переміщування головки: ліворуч, на місці, праворуч.

Побудуємо більш складну машину Тюринга, створення якої зміцнить упевненість в нашій можливості подавати у такий спосіб довільний алгоритм.

Приклад 2. Побудувати машину Тюринга, котра виконує множення чисел в унарній системі числення.

Алгоритм полягає в послідовному додаванні множеного до часткового результату; кількість додавань задається множником; для запам'ятовування доданої (дописаної) одиниці використовуватимемо символ *e*, а для поділу множеного й результату – символ *b*. Опишемо коротко основні кроки:

1. зсуваємося праворуч до λ, ставимо роздільник *b*;
2. зсуваємося ліворуч до λ;
3. стираємо паличку першого числа (множника); якщо палички відсутні – йти до 5;
4. дописуємо друге число (множене) праворуч, переходимо до 3;
5. стираємо множене та роздільники.

Отже $X = \{1, *, \lambda, e, b\}$, $Q = \{q_0, ..., q_9\}$, $q_0 = q_0$, $q_f = q_9$, P:

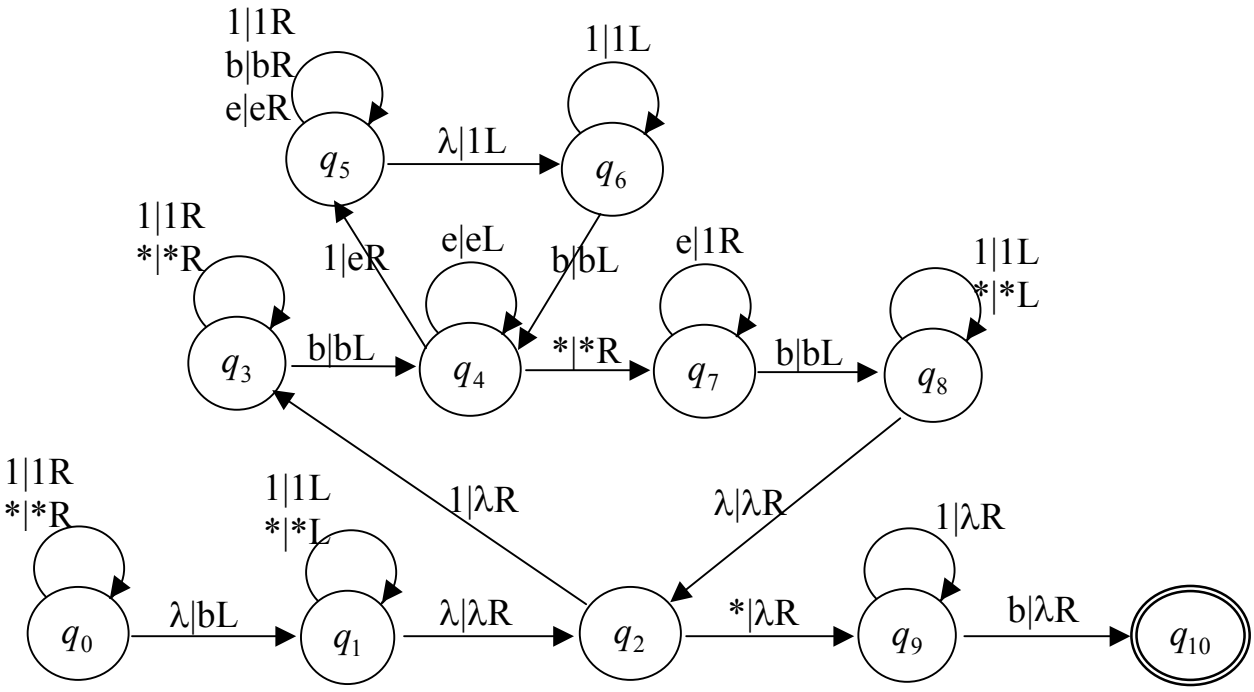


Рисунок 3.3 – Машина Тюринга для множення чисел

Для розуміння алгоритму (рис. 3.3) є суттєво, що узяту в стані q_4 одиницю ми замінюємо на символ *e*, переносимо праворуч до краю в стані q_5 та замінюємо на одиницю перший праворуч порожній символ; у стані q_6 повертається до роздільника *b*,

а потім у стані q_4 – до першої неперенесеної одиниці. У стані q_7 множене відновлюється, і в стані q_8 ми переміщуємося ліворуч до краю для віднімання чергової одиниці множника.

Рекомендуємо виконати ручне прокручування побудованої машини для розуміння її роботи.

Машину Тюринга можна в певному розумінні розглядати як попередник комп'ютера. При побудові складних машин Тюринга використовують прийоми, звичайні для технології програмування: послідовне з'єднання, розгалуження, організацію циклів. Слід зазначити, що, хоча множина команд машини Тюринга є доволі обмежена порівнянно із сучасними процесорами, стрічка машини є нескінченна, що зумовлює її абстрактний, гіпотетичний характер. Це робить її незрівнянно більш потужною навіть стосовно суперкомп'ютерів, ресурси яких хоча є й великі, але завжди скінченні. Система команд комп'ютера формується, як правило, виходячи з критеріїв зручності складання програм. Система команд машини Тюринга є мінімальна й призначена для виконання формальних доведень існування алгоритмів.

Слід зауважити, що сітка Петрі зі спеціальними інгібіторними дугами, тако само як і машини Тюрінга, є універсальною алгоритмічною системою. Інгібіторна дуга дозволяє перевірити маркування позиції на нуль. Перехід спрацьовує, якщо вхідна позиція не містить маркерів. Еквівалентними до машини Тюринга є також синхронні сітки Петрі й сітки Петрі з пріоритетами.

Завдяки успішній побудові машин Тюринга для розв’язання різноманітних задач було сформульовано гіпотезу, названу тезою Чорча: кожний алгоритм може бути реалізовано за допомогою відповідної машини Тюринга. Зауважимо, що спочатку тезу було сформульовано для рекурсивних функцій Кліні, але, як ми вже згадували раніш, різні моделі, запропоновані для формалізації інтуїтивного поняття алгоритму, виявилися еквівалентними. Слід зазначити, що тезу Чорча неможливо довести математично, оскільки клас задач, подаваний інтуїтивним поняттям алгоритму, не визначено формально. Можна було б її спростувати, якби вдалося побудувати такий алгоритм, для реалізації якого не існує машини Тюринга. Але цього нікому не вдалося зробити й навіть, більш того, побудова безлічі машин Тюринга для розв’язання різних задач свідчить про слушність тези. Можна навести певну аналогію з відомим у фізиці законом збереження енергії, який також неможливо довести, але й дотепер нікому не вдалося його спростувати.

3.3 Алгоритмічно нерозв'язні проблеми

Створення машин Тюринга було необхідне, власне кажучи, для того, аби довести, що для певних задач не існує алгоритмів їхнього розв’язання. Розглядання таких задач розпочнемо з уявлюваної, дещо штучної проблеми самозастосовності машини Тюринга. Однак цю проблему буде використано нами для доведення алгоритмічної нерозв'язності інших задач та побудови загального методу доведення алгоритмічної нерозв'язності.

Закодуємо програму машини Тюринга. Для цього подамо функцію переходів як послідовність команд вигляду $(q,\ x,\ q',\ x',\ r)$. Занумеруємо символи абетки стрічки, стани й переміщування головки. Для поділу компонентів команди використ-

товуватимемо символ *, для поділу команд – крапку з комою. Тоді програму довільної машини Тюринга можна подати на стрічці у формі:

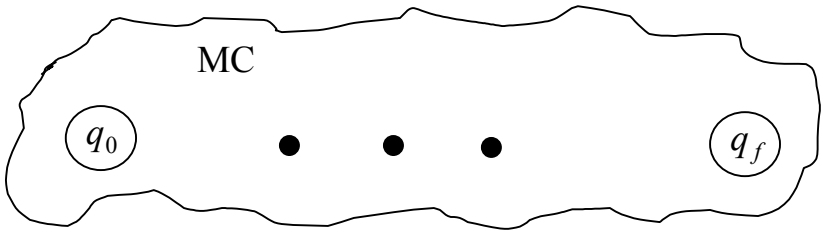
…	λ	q	*	x	*	q`	*	x`	*	r	;	…	λ	…

Також вважатимемо, що перша команда відповідає початковому стану машини, а остання – завершальному.

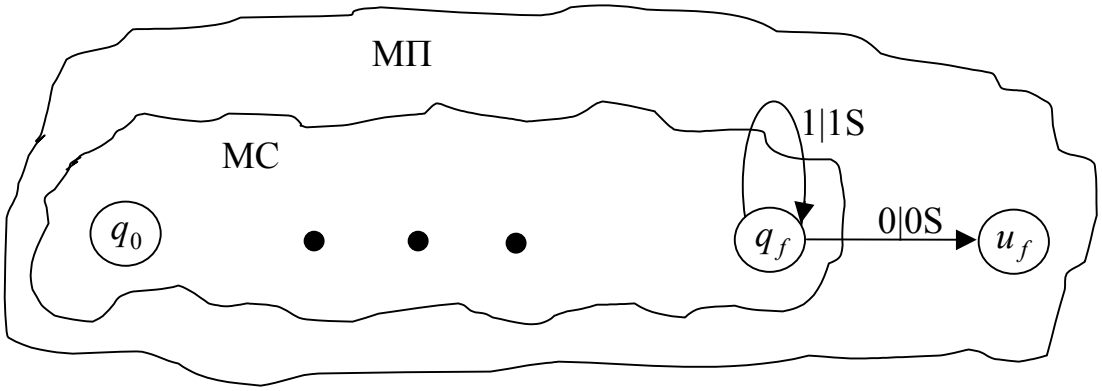
Шифр *Ш(М)* машини *М* запишемо на стрічку, і стрічку подамо машині. Машина є *самозастосовною*, якщо вона зчитує свій шифр і через скінченну кількість кроків зупиняється. Якщо машина ніколи не зупиняється, вона є *несамозастосовною*. Завдання полягає в тім, аби за заданою машиною *М* визначити, чи є вона самозастосовною. Для певності вважатимемо, що машина *МС* розв'язує задачу самозастосовності, якщо ця машина, зчитуючи *Ш(М)*, видає на стрічці 1, якщо *М* є самозастосовна, й 0 – у противному разі.

Теорема 3.1. Проблема самозастосовності машини Тюринга є алгоритмічно нерозв'язна, тобто не існує такої машини Тюринга *МС*, яка розв'язує цю проблему.

Доведення. Побудуємо доведення від противного. Припустімо, що існує машина Тюринга *МС*, яка розв'язує проблему самозастосовності:



Добудуємо *МС* до *МП* у такий спосіб:



Тобто *МП*, читаючи *Ш(М)*, якщо *М* є самозастосовна, доходить до q_f і циклиться; якщо *М* є несамозастосовна, то *МП* зупиняється. Тоді виникає запитання: чи є самозастосовною *МП*.

Припустімо, що *МП* є самозастосовна. Тоді, зчитуючи свій власний шифр, *МП* має зупинитися. З іншого боку, машина *МС* зчитуючи *Ш(МП)*, видає 1 на стрічці й потрапляє до q_f, але тоді *МП* не потрапляє до заключного стану u_f і не зупиняється. Отже, вона є несамозастосовна.

Нехай *МП* є несамозастосовна. Зчитуючи свій власний шифр *Ш(МП)*, машина *МП* не зупиняється. З іншого боку, *МС* зчитуючи шифр *Ш(МП)*, переходить до q_f й записує на стрічці 0. Тоді *МП* переходить до u_f й має зупинитися.

Отже, ми маємо суперечність, оскільки припустили існування машини *МС*, яка розв'язує задачу самозастосовності. З набутої суперечності випливає, що такої машини *МС* не існує.

□

3.4 Метод зведення

Проблема самозастосовності є першою з проблем, для яких ми довели алгоритмічну нерозв'язність. Незважаючи на свою удавану відокремленість і далекість від практичних задач, вона відіграє ключову роль, оскільки буде використана в доведенні алгоритмічної нерозв'язності інших проблем. Метод, за допомогою якого ми доводитимемо алгоритмічну нерозв'язність одних проблем, використовуючи раніше доведену алгоритмічну нерозв'язність певної іншої проблеми, зватиметься методом зведення. Цей метод є основним для доведення алгоритмічної нерозв'язності.

Відома історія про те як розігрівають чайник інженер і математик, наочно ілюструє особливості застосовування цього методу. Задачу першу про те, як нагріти воду, якщо чайник є порожній, обидва розв'язують однаково: налити воду, поставити на пальник, запалити газ, очікувати закипання. Якщо ж на пальнику стоїть теплий чайник, інженер запалює газ і очікує закипання. Математик же виливає воду і зводить задачу до попередньої.

Нехай Z – певна проблема, а $\{z_i\}$ – її конкретні задачі. Наприклад, якщо Z – проблема самозастосовності, то z_i полягає в перевірці самозастосовності для певної конкретної машини M, а сама Z полягає в перевірці самозастосовності для якої завгодно машини Тюринга.

Нехай маємо $Z = \{z_i\}$, $Z' = \{z'_j\}$. Говоритимемо, що Z зводиться до Z' ($Z \to Z'$), якщо існує перехід, реалізований певним алгоритмом, від кожної z_i до певної z'_j такий, що відповідь для z_i є позитивна тоді й лише тоді, коли відповідь для z'_j теж є позитивна.

Якщо доведено, що Z є алгоритмічно нерозв'язна, тоді Z' теж має бути алгоритмічно нерозв'язна. Насправді, в тому разі, якщо Z' є розв'язна за допомогою, наприклад, машини *M0*, послідовне з'єднання машин дозволяє розв'язати проблему Z: $z_i \overset{M1}{\to} z'_j \overset{M0}{\to} \{0,1\}$.

Нехай треба довести алгоритмічну нерозв'язність Z, тоді ми намагаємося віднайти вже доведену алгоритмічно нерозв'язну проблему *Z0* й, окрім цього, намагаємося побудувати послідовність проблем, що вони зводяться: $Z0 \to Z1 \to \ldots \to Z$. Якщо це вдається, то тим самим ми доводимо алгоритмічну нерозв'язність проблеми Z.

Проілюструємо використання методу зведення на прикладі доведення алгоритмічної нерозв'язності для проблеми зупинки машини Тюринга.

Проблема зупинки машини Тюринга полягає в тому, щоби для довільної машини Тюринга *M* й довільного запису на стрічці *X* визначити, чи зупиниться машина *M* при оброблянні вхідної стрічки *X*.

Теорема 3.2. Проблема зупинки машини Тюринга є алгоритмічно нерозв'язна.

Доведення. Побудуємо доведення від противного. Нехай проблема зупинки розв'язна. Тоді, обравши замість *X* шифр *Ш(M)*, ми зможемо розв’язувати проблему самозастосовності. Отже, ми доходимо до суперечності, що й доводить теорему.

□

Наведемо кілька відомих прикладів алгоритмічно нерозв'язних проблем:

1. Проблема розпізнавання істинності формул елементарної арифметики.

Формули будуються за допомогою арифметичних дій (додавання та множення), логічних операцій (логічних зв'язувань та кванторів) та знака рівності з константи 0 і натуральнозначних змінних. Проблема полягає у вимозі віднайти алгоритм, який за кожною такою формулою визначав би, виконується вона на натуральному ряді чи ні.

2. Проблема сполучуваності Поста.

Нехай задано скінченну множину *V* пар слів у певній абетці. Назвемо цю множину сполучуваною, якщо для певних пар (A_1,B_1), (A_2,B_2)... (A_s,B_s) з *V* виконується рівність $A_1...A_s = B_1...B_s$. Треба за кожною скінченною множиною *V* пар слів у даній абетці алгоритмічно установити, сполучувана вона є чи ні.

3. Проблема подавання матриць.

Розглянемо квадратні матриці порядку *n* з цілими коефіцієнтами. За означенню матриця *U* є подавана через матриці $U_1,U_2,...,U_q$, якщо для певних $r_1,r_2,...,r_t$ виконується рівність $U = \prod_{i=1,t} U_{r_i}$. Загальна проблема подавання полягає у вимозі зазначити алгоритм, за допомогою якого можна було б указати для будь-якої системи матриць *U*, U_1, U_2, ..., U_q порядку *n*, чи є *U* подавана через U_1, U_2, ..., U_q. Загальна проблема подавання для матриць порядку *n* не має розв’язку за $n \geq 3$.

4. Десята проблема Гільберта: задача про можливість розв'язання діофантова рівняння.

Нехай задано діофантове рівняння з довільними невідомими і цілими раціональними числовими коефіцієнтами. Зазначити спосіб, за допомогою якого можливо після скінченної кількості операцій установити, чи розв'язне є це рівняння в цілих раціональних числах.

5. Проблема еквівалентності сіток Петрі.

Для двох довільних заданих сіток Петрі встановити, чи збігаються їхні вільні мови.

Надто цікаві є наслідки з результатів, пов'язаних з алгоритмічною нерозв'язністю, здобуті Райсом: за двома довільними програмами алгоритмічною мовою не можна з'ясувати, обчислюють вони ту ж саму функцію чи ні. Або в ще більш вражаючій формі: за довільної програми алгоритмічною мовою не можна з’ясувати, чи обчислює ця програма функцію, визначену хоча б в одній точці.

Не слід надто засмучуватися, розширивши свої знання поданнями про алгоритмічно нерозв'язні проблеми. Власне термін алгоритмічної нерозв'язності не пов'язано з повною безпорадністю людини. Нерозв'язність певної проблеми на інтуїтив-

ному рівні означає лише, що проблема має надто загальне формулювання. Для різних окремих випадків проблеми можуть існувати алгоритми розв’язування. Отже, необхідно лише звузити постановку задачі.

Контрольні запитання

1. З чого складається скінченний автомат?
2. Назвіть способи подання скінченного автомата.
3. Що зображують на діаграмі станів скінченного автомата?
4. Яка є структура матриці переходів скінченного автомата?
5. Чим відрізняється автомат Мілі від автомата Мура?
6. Сформулюйте основні дії з перетворення автомата Мілі на автомат Мура.
7. Сформулюйте основні дії з перетворення автомата Мура на автомат Мілі.
8. Зазначте основні області застосовування скінченних автоматів.
9. Назвіть скінченні автомати в оточуючих нас штучних об'єктах.
10. У чому полягає доцільність використання двох різних типів скінченних автоматів?
11. Які автомати називають еквівалентними?
12. Схарактеризуйте основні кроки алгоритму мінімізації скінченного автомата?
13. Як визначити, чи є скінченні автомати еквівалентними?
14. Перелічіть основні елементи сітки Петрі.
15. Назвіть основні області застосовування сіток Петрі.
16. Сформулюйте умову збудження переходу сітки Петрі.
17. У чому полягає спрацьовування переходу сітки Петрі?
18. Опишіть коротко процес функціонування сітки Петрі.
19. Зазначте способи наочного подання динаміки сітки Петрі.
20. Що являє собою граф досяжних маркувань сітки Петрі?
21. Які сітки Петрі називають ординарними?
22. Які матриці дозволяють подавати сітку Петрі?
23. Що являє собою рівняння станів сітки Петрі?
24. Перелічіть основні властивості сіток Петрі.
25. Які сітки називають обмеженими й безпечними?
26. У чому полягає властивість консервативності сітки Петрі?
27. У чому полягає властивість живості сітки Петрі?
28. Яке маркування сітки Петрі називають тупиковим?
29. Окресліть взаємозв'язок сіток Петрі й скінченних автоматів.
30. Схарактеризуйте структуру фундаментального рівняння сітки Петрі.
31. Що являє собою інваріант позицій сітки Петрі?
32. Що являє собою інваріант переходів сітки Петрі?
33. Схарактеризуйте властивості інваріантних сіток Петрі.
34. У чому полягає редукція сіток Петрі?
35. Зазначте основні правила редукції сіток Петрі.
36. Назвіть особливості побудови псевдомаркувань сітки Петрі.
37. Сформулюйте основні кроки алгоритму побудови дерева покривних маркувань сітки Петрі.

38. Які властивості сіток Петрі дозволяє визначити дерево покривних маркувань?
39. Назвіть переваги використання сіток Петрі для моделювання паралельних систем та процесів.
40. Сформулюйте інтуїтивне поняття алгоритму.
41. Назвіть основні риси, притаманні алгоритмові.
42. З яких елементів складається машина Тюринга?
43. Коротко опишіть процес функціонування машини Тюринга.
44. У чому полягає теза Чорча?
45. Сформулюйте проблему самозастосовності машини Тюринга.
46. Чи є проблема самозастосовності машини Тюринга алгоритмічно розв'язною?
47. У чому полягає метод зведення для доведення алгоритмічної нерозв'язності.
48. Які алгоритмічно нерозв'язні проблеми Вам відомі?
49. Назвіть алгоритмічно нерозв'язні проблеми в теорії сіток Петрі.
50. В чому полягає взаємозв'язок машини Тюринга й розширених сіток Петрі.

Задачі

1. Побудуйте скінченний автомат для продажу шоколадок. До автомата можна опускати монети достоїнством 5, 10 та 25 копійок. Вартість шоколадки – 25 копійок. Якщо опущена сума є достатня для придбання шоколадки, то загоряється лампочка і за допомогою кнопок можна обрати одну з трьох начинок: ромову, полуничну чи вишневу.
2. Перетворіть автомат, побудований в задачі 1, на форму автомата Мура.
3. Виконайте мінімізацію автомата із задачі 2.
4. Побудуйте скінченний автомат для розпізнавання двійкових чисел з фіксованою крапкою. Виконайте мінімізацію автомата.
5. Побудуйте сітку Петрі, що вона моделює протокол множинного доступу з контролем несучої.
6. Віднайдіть інваріанти сітки, побудованої в задачі 4.
7. Побудуйте дерево покривних маркувань сітки, побудованої в задачі 4.
8. Виконайте редукцію сітки, побудованої в задачі 4.
9. Визначіть, чи є сітка, побудована в задачі 4, обмеженою, живою, консервативною.
10. Побудуйте сітку Петрі, що вона моделює сеанс радіозв'язку в напівдуплексному режимі. Назвіть властивості сітки.
11. Побудуйте машину Тюринга для розпізнавання чисел у двійковій системі числення.
12. Побудуйте машину Тюринга для підрахунку нулів у двійковому числі.

Список рекомендованої літератури

1. **Глушков В.М.** Синтез цифровых автоматов. – М.: Физматгиз, 1962. – 476 с.

2. **Шоломов Л.А.** Основы теории дискретных логических и вычислительных устройств. – М.: Наука, 1980. – 400 с.
3. **Питерсон Дж.** Теория сетей Петри и моделирование систем. – М.: Мир, 1984. – 264 с.
4. **Котов В.Е.** Сети Петри. – М.: Наука, 1984. – 160 с.

Список додаткової літератури

1. Грунский И.С., Козловский В.А., Пономаренко Г.П. Представление конечных автоматов фрагментами поведения. – Киев: Наук. думка, 1990. – 232 с.
2. Слепцов А.И., Юрасов А.А. Автоматизация проектирования управляющих систем гибких автоматизированных производств / Под ред. Б.Н.Малиновского. – К. Техніка, 1986. – 160 с.
3. Ачасова С.М., Бандман О.Л. Корректность параллельных вычислительных процессов. – Новосибирск: Наука, 1990. – 254 с.
4. Марков А.А., Нагорный Н.М. Теория алгорифмов. – М.: Наука, 1984. – 432 с.
5. Успенский В.А., Семёнов А.Л. Теория алгоритмов: основные открытия и приложения. – М.: Наука, 1987.– 288 с.
6. Мурата Т. Сети Петри: Свойства, анализ, приложения // ТИИЭР. – т. 77, № 4, 1989. – с. 541-580.
7. Зайцев Д.А., Слепцов А.И. Уравнения состояний и эквивалентные преобразования временных сетей Петри // Кибернетика и системный анализ. – 1997, № 5. – с. 59-76.
8. Зайцев Д.А. Инварианты функциональных подсетей // Научные труды ОНАС им. А.С. Попова. – № 4, 2003. – с. 57-63.
9. Зайцев Д.А. Инварианты временных сетей Петри // Кибернетика и системный анализ. – № 2, 2004. – с. 92-106.
10. Зайцев Д.А., Шмелёва Т.Р. Моделирование коммутируемой локальной сети раскрашенными сетями Петри // Зв'язок. – № 2 (46), 2004. – с. 56-60.

Навчальне видання

Зайцев Дмитро Анатолійович

Математичні моделі дискретних систем

Навчальний посібник

Відповідальний редактор:

Редактор: І.В. Ращупкіна