УДК 004.832; 004.852; 004.891

П.О. Кравець, О.М. Проданюк
Національний університет "Львівська політехніка"
кафедра інформаційних систем та мереж

ПРИЙНЯТТЯ ОПТИМАЛЬНИХ РІШЕНЬ МЕТОДОМ НАВЧАННЯ З НЕЧІТКОЮ ЛОГІКОЮ

Досліджується проблема оптимального прийняття рішень в умовах невизначеності за допомогою марківських методів навчання з нечіткою логікою. Описано структуру та функції системи прийняття рішень на основі продукційних правил нечіткої логіки. Специфіковано етапи перетворення нечітких даних під час логічного виведення рішень. Наведено та проаналізовано результати комп'ютерного моделювання нечіткого прийняття рішень у клітинному просторі.

Ключові слова – марківські методи навчання, нечітка логіка, комп'ютерне моделювання нечіткого прийняття рішень.

The problem of optimum decision-making in the conditions of uncertainty by the Markovian learning methods with the fuzzy logic is investigated. The structure and functions of fuzzy decision-making system on the basis of productional rules of fuzzy logic are described. The stages of the fuzzy data transformation in the course of a logic conclusion of decisions are specify. The results of computer simulation of a fuzzy logic decision-making process in cellular space are resulted and analysed.

Keywords – Markovian learning methods, the fuzzy logic, computer simulation of a fuzzy logic decision

Вступ

Процес прийняття рішень в умовах невизначеності повинен бути ітераційним, адаптивним, спрямованим на зменшення середньої похибки оптимального значення характеристичної функції системи [1 – 3]. Цілеспрямована поведінка процесу прийняття рішень досягається за рахунок навчання, здатності запам'ятовувати ефективність дій у передісторії та забезпечувати локально-оптимальну, підпорядковану глобальній меті динаміку системи на основі інформації про її поточний стан [4].

Для селективного опрацювання даних про ефективність рішень за відсутності математичної моделі системи використовується її числова ідентифікація у просторі стан-дія. Для формування числової моделі системи використовуються різноманітні методи марківського навчання, серед яких за критеріями ефективності, простоти реалізації та популярності застосування варто виділити групу методів під узагальнюючою назвою "заохочувальне" або "підкріплене" навчання [5 – 8].

Структура системи прийняття рішень на основі заохочувального навчання складається з середовища та інтелектуального агента, під яким розуміють активну підсистему вироблення та реалізації рішень, яка впливає на стани середовища за допомогою керуючих дій та у разі потреби взаємодіє з іншими подібними агентами і людиною [9, 10]. Агент може мати інформаційну (у вигляді програми) або технічну (у вигляді робота) реалізацію. Як правило, агент наділяється антропоморфними властивостями – він має систему сенсорів для спостереження станів середовища, систему виведення рішень, яка перетворює вхідну інформацію на керуючі дії згідно із заданою метою, систему реалізації керуючих дій. За методом заохочувального навчання, крім спостереження станів системи, агент після реалізації керуючої дії отримує поточний виграш, який

використовується як ознака правильності виробленого рішення, його відповідності поставленій меті функціонування системи. Ітераційні процедури методів заохочувального навчання ґрунтуються на моделі динамічного програмування і складаються з етапу локального розвідування станів середовища з метою пошуку оптимальних варіантів прийняття рішень та етапу використання розвіданих оптимальних станів [6].

У разі відсутності або неточності математичної моделі керованого середовища існує необхідність автоматичного вироблення та реалізації оптимальних рішень на основі баз експертних знань, поданих у вигляді операцій над неповними або розмитими даними. У такому разі інтелектуальний агент повинен змоделювати процес прийняття рішень людиною, спеціалістом у вибраній галузі знань. Для відтворення процесу розмірковування експерта щодо вибору адекватної керуючої дії при спостереженні поточного стану системи використовується математичний апарат нечіткої логіки [11 – 14]. Застосування нечіткого логічного виведення дає змогу використати експертну апроксимацію середовища у моделях заохочувального навчання і тим самим до деякої міри компенсувати невизначеність системи прийняття рішень.

Започатковане у [15] заохочувальне навчання з нечіткою логікою розвинуто у роботах [16 – 21] з диференціацією неперервного та дискретного простору станів і дій, структури системи прийняття рішень, механізмів адаптації до невизначеностей, схем застосування правил нечіткого логічного виведення та варіантів їх узагальнення, алгоритмів формування керуючих дій.

Сучасні дослідження систем прийняття рішень з нечіткою логікою пов’язані з підвищенням ефективності методів заохочувального навчання, зростанням інтелектуальних здібностей агентів на основі нечітких продукційних правил, забезпечення несуперечливості та повноти баз правил, адаптивного поповнення баз нечітких правил, розроблення гібридних методів штучного інтелекту агентів – нечітких штучних нейронних мереж, нечітких генетичних методів, нечітких когнітивних карт та інших [22 – 26]. Незважаючи на широкий фронт дослідження, системи прийняття рішень з нечіткою логікою мають низку невирішених проблем, пов’язаних з особливостями структурно-функціональної організації агентів, урахування експертних знань у моделях заохочувального навчання, визначення умов збіжності та забезпечення точності методів адаптивного нечіткого логічного виведення.

Метою роботи є розроблення структури та функцій інтелектуальних агентів з нечіткою логікою на основі методів заохочувального навчання та їх застосування для розв’язування практичних проблемно-орієнтованих задач прийняття рішень в умовах невизначеності. Досягнення мети ґрунтується на застосуванні нечіткого логічного виведення рішень на основі баз продукційних правил для розроблення алгоритму функціонування інтелектуального агента. Емпірична перевірка працездатності розробленого алгоритму здійснюється для задачі пошуку агентом оптимального значення функції у клітинному просторі з визначенням напрямку переміщення за правилами нечіткої логіки.

Прийняття рішень на основі нечіткого логічного виведення

Нехай нечітка система здійснює вибір варіантів рішень на основі залежності вихідної величини від декількох вхідних величин. Допустимо, що математична модель залежності виходу від входів відсутня і замість неї використовується база експертних правил у вигляді нечітких висловлювань “ *if – then* ” у термінах лінгвістичних змінних та нечітких множин.

Тоді функціональність нечіткої системи прийняття рішень визначається такими кроками [24]:

1) перетворення чітких вхідних змінних на нечіткі, тобто визначення ступеня відповідності входів кожній із нечітких множин;
2) обчислення правил на основі використання нечітких операторів та застосування імплікації для отримання вихідних значень правил;
3) агрегування нечітких виходів правил у загальне вихідне значення;

4) перетворення нечіткого виходу правил на чітке значення.

Структура системи з нечіткою логікою зображена на рис. 1. Система побудована за схемою багатошарової штучної нейромережі, яка складається з вхідного, двох прихованих та вихідного шару.

Перший шар зображає входи системи, другий шар – нечіткі лінгвістичні змінні, третій шар – правила над нечіткими змінними, четвертий шар – виходи правил. Ваги усіх шарів, крім останнього, дорівнюють 1. Ваги зв'язків між шаром правил та вихідним шаром визначаються алгоритмом навчання.

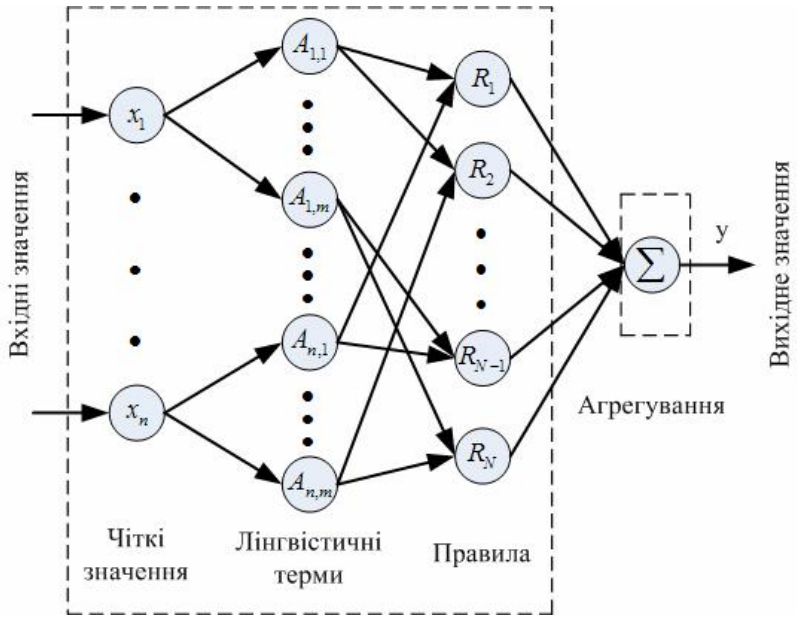

Рис. 1. Структура системи нечіткого логічного виведення

Входи $\bar{x} = (x_i \mid i = 1..n)$ (наприклад, тиск, об'єм) та вихід y (наприклад, температура) є чіткими контрольованими величинами. Кожен параметр x_i, $i = 1..n$ має нечіткий відповідник у вигляді лінгвістичної змінної $\overline{X}_i = \{A_{i,j} \mid j = 1..m_i\}$. Лінгвістична змінна $\overline{X}_i$ складається з m_i термів $A_{i,j}$, кожен з яких є нечіткою множиною.

Правила R_k, $k = 1..N$ перевіряють значення кожної лінгвістичної змінної, тому максимально можлива кількість правил дорівнює $N_{\max} = \prod_{i=1}^{n} m_i$. Реальну кількість правил позначимо через $N \le N_{\max}$.

Вихід правила – це лінгвістична змінна $\overline{Y} = \{B_j \mid j = 1..m\}$, яка набуває значення одного із термів B_j.

Для узагальнення правил відбувається агрегування їх нечітких виходів в одну нечітку множину з її подальшим перетворенням на чітке вихідне значення.

Фазифікація полягає у перетворенні чітких вхідних величин $\bar{x} = (x_1, x_2, ..., x_n)$ на нечіткі множини $A' = (A'_1, A'_2, ..., A'_n)$. Здебільшого для цього використовуються синглетонні моделі. Синглетон чіткого значення x_i є нечіткою множиною $A'_i(x, m_{A'_i}(x))$ з функцією належності

$$m_{A'_i}(x) = \begin{cases} 1, x = x_i; \\ 0, x \neq x_i. \end{cases}$$

При фазифікації чіткого входу x_i визначають ступені його відповідності кожному лінгвістичному терму $A_{i,j}$ з функціями належності $m_{A_{i,j}}(x)$, $j=1..m_i$. Ці ступені є значеннями функцій належності $m_{A_{i,j}}(x)$ у точці $x=x_i$, або інакше – значенням $A_{i,j}(x_i)$, $i=1..n$.

Нечіткі вхідні значення системи перетворюються на вихідні на основі правил нечіткої логіки, що характерно для експертних систем прийняття рішень. Нехай система прийняття рішень здійснює перетворення значень n вхідних лінгвістичних змінних $\vec{X} = \{\vec{X}_i \mid i=1..n\}$ на вихідну лінгвістичну змінну $\vec{Y} = R(\vec{X})$ згідно з базою правил $R = \{R_k \mid k=1..N\}$. Правила R акумулюють знання експертів у вигляді нечіткої імплікації $R = A \to B$, яку можна розглядати як нечітку множину на декартовому добутку носіїв вхідних та вихідних розмитих множин. Процес отримання нечіткого результату B' з нечітких вхідних множин A' на основі знань $A \to B$ можна зобразити у такому вигляді

$$B' = A' \bullet R = A' \bullet (A \to B),$$

де $\bullet$ – композиційне правило нечіткого виведення.

На практиці для нечіткого виведення використовується максимінна композиція, а нечітка імплікація реалізується знаходженням мінімуму функцій належності.

Для імітації роботи експертної системи за схемою імплікації застосовується множина нечітких продукційних правил, кожне з яких будується у вигляді умовного оператора:

if *логічний вираз* ***then*** *оператор,*

де логічний вираз – висловлювання, побудоване на основі базових логічних операцій над нечіткими величинами; оператор – результуюче рішення. Правила можуть визначати відношення відповідності (is) між вхідними лінгвістичними змінними $\vec{X}$ та їх нечіткими термами $\{A_{i,j} \mid i=1,..,n; j=1..m_i\}$. Використання нечітких умовних правил є природним для подання знань експертами і спрощує їх машинне опрацювання.

Загалом правило може містити усі можливі комбінації лінгвістичних термів для усіх вхідних змінних, об'єднаних логічними операціями.

Варто зазначити, що за допомогою перетворень нечітких множин будь-яке правило, що містить у лівій частині як кон'юнкції, так і диз'юнкції, можна перетворити на систему правил, у лівій частині яких будуть або тільки кон'юнкції, або тільки диз'юнкції. Для визначення нечіткої кон'юнкції можна використати знаходження мінімуму, а для нечіткої диз'юнкції – знаходження максимуму двох функцій належності. Не зменшуючи загальності, будемо розглядати правила, побудовані на основі кон'юнкції.

Розрізняють дві моделі логічного виведення: Мамдані (Mamdani) та Такагі–Суджено (Takagi-Sugeno) [13].

Модель Мамдані оперує лише з лінгвістичними змінними та нечіткими множинами і перетворює нечіткі входи на нечіткі виходи. Наприклад, для моделі Мамдані правила мають вигляд:

$$R_k : if\ \vec{X}_1\ is\ A_{1,k}\ and\ ...and\ \vec{X}_n\ is\ A_{n,k}\ then\ \vec{Y}\ is\ B_k,$$

де $A_{i,k} \in \vec{X}_i$ – нечіткі множини для вхідних та $B_k \in B$ – нечіткі множини для вихідної лінгвістичної змінної, які використовуються в *k*-му правилі ($k=1..N$). Операція *and* інтерпретується як *t*-норма нечітких множин.

Модель Такагі–Суджено оперує з чіткими величинами, лінгвістичними змінними та нечіткими множинами і перетворює чіткі входи на чіткі виходи. Правила моделі Такагі–Суджено мають вигляд:

$$R_k : if\ x_1\ is\ A_{1,k}\ and\ ...and\ x_n\ is\ A_{n,k}\ then\ y = f_k(x_1, x_2, ..., x_n), \quad (1)$$

де $y = f_k(x_1, x_2, ..., x_n)$ – функція заключної частини *k*-го правила.

У частковому випадку правила (1) можуть бути задані у вигляді:

$$R_k : if\ x_1\ is\ A_{1,k}\ and\ ...and\ x_n\ is\ A_{n,k}\ then\ y = o_k, \quad (2)$$

де o_k – константне значення *k*-го правила.

Для повноти бази нечітких правил повинні виконуватися такі умови:

1) для будь-якого терму вхідної змінної існує хоча б одне правило, в якому цей терм використовується у лівій частині правила;

2) існує хоча б одне правило для кожного лінгвістичного терму вихідної змінної.

Для багатовходових систем застосовується механізм логічного виведення, характерною рисою якого є використання рівнів істинності передумов правил.

Для кожного правила R_k, $k = 1..N$ визначається рівень його істинності a_{R_k} стосовно входів. Рівень істинності є дійсним числом, яке характеризує ступінь відповідності нечітких входів системи A_i', $i = 1..n$ заданим у правилах нечітким множинам $A_{i,j}$ ($j = 1..m_i$):

$$a_{R_k} = \min_{i=1}^{n} \left[\max_{X_i} \left(A_i' \wedge A_{i,j} \right) \right],$$

де X_i – простір визначення входів A_i', $i = 1..n$; операція $\wedge$ – нечітка кон'юнкція.

У разі використання вхідних синглетонів механізм логічних виведень спрощується, оскільки ступінь істинності правил може бути визначений на основі фазифікованих входів:

$$\max_{X_i} \left(A_i'(x_i) \wedge A_{i,j} \right) = A_{i,j}(x_i).$$

У цьому випадку обчислення рівня істинності *k*-го правила буде формуватися за формулою:

$$a_{R_k} = \min_i (A_{i,j}(x_i)).$$

Кожне із правил є нечіткою імплікацією, яка визначає вихідне значення залежно від рівня істинності лівої частини правила. Ступінь впевненості виведення задається функцією належності відповідного вихідного терму B_k. Використовуючи один зі способів побудови нечіткої імплікації, одержимо нові нечіткі змінні, або відповідні ступені впевненості в значенні виходів із застосуванням відповідного правила до заданих входів. Так, на основі визначення нечіткої імплікації за Мамдані як мінімуму лівої й правої частин правила, маємо:

$$B_k' = \min(a_k, B_k), \quad k = 1..N,$$

де B_k' – зрізи вихідних нечітких множин на рівні a_k.

Завершальним кроком нечіткого логічного виведення є агрегування виходів правил. Один з основних способів акумуляції – нечітка диз'юнкція вихідних множин, або, інакше, знаходження максимуму отриманих функцій належності. Як результат одержимо значення агрегованого виходу:

$$B' = \max_k \left(B_k' \right), \quad k = 1..N.$$

Під час нечіткого логічного виведення виконується паралельне опрацювання великої кількості правил з подальшим їх агрегуванням у заключне рішення. Правила можуть будуватися на основі досвіду та знань експертів, створенням моделі дій оператора, методом навчання. Під час проектування пристроїв з нечіткою логікою важливим є забезпечення можливості їх пристосування до змін навколишнього середовища методом навчання бази правил за експериментальними даними. Навчання полягає в адаптивному підборі параметрів нечітких множин та автоматичному генеруванні правил нечіткого логічного виведення. Для цього використовують алгоритми оптимізації та інтелектуального опрацювання даних – градієнтний, генетичний, штучних нейронних мереж, байєсових мереж тощо.

Після визначення індивідуальних виходів правил здійнюється дефазифікація агрегованого виходу. У загальному етап дефазифікації є необов'язковим і використовується у разі необхідності перетворення виведених нечітких лінгвістичних змінних до точного значення.

Існує декілька методів дефазифікації – метод середнього центра, перший максимум, середній максимум, висотна дефазифікація [13]. Наприклад, метод середнього центра, або центроїдний метод, визначається центром ваги вихідної нечіткої множини:

$$y = \frac{\sum_{j=1}^{m} y_j B'(y_j)}{\sum_{j=1}^{m} B'(y_j)}.$$

Для моделі Такагі–Суджено вихідні множини правил задаються у вигляді синглетонів з функціями належності

$$m_{o_k}(y_k) = \begin{cases} 1, & y_k = o_k \\ 0, & y_k <> o_k \end{cases},$$

де o_k – еталонне вихідне значення k-го правила.

Тоді результуюче чітке вихідне значення системи прийняття рішень обчислюється зважуванням значень активованих правил:

$$y = \frac{\sum_{k=1}^{N} a_{R_k} y_k}{\sum_{k=1}^{N} a_{R_k}}.$$

У системах керування отримане чітке вихідне значення використовується у контурі зворотного зв'язку для вироблення керуючих дій.

Методи заохочувального навчання

В основі концепції заохочувального навчання лежить рефлексивна поведінка біологічних організмів з розвиненою нервовою системою. Інформаційною моделлю таких організмів є інтелектуальний агент, який спостерігає стани системи, аналізує поточну ситуацію, приймає рішення, реалізує відповідну дію, змінює стан системи, отримує від середовища поточну винагороду. Діючи цілеспрямовано, агент повинен навчитися приймати такі рішення, які у середньому забезпечують йому найбільший виграш. У роботах [5 – 7] показано, що заохочувальне навчання можна будувати на основі марківських випадкових процесів.

Марківський процес – це кортеж (S, U, p, r), де S – набір усіх станів системи, U – набір можливих дій агента, $p: S \times U \to \Delta(S)$ – функція зміни станів системи, $r: S \times U \to R$ – функція винагороди. Тут $\Delta(S)$ – набір усіх розподілів імовірностей на множині S. Імовірність зміни станів p залежить тільки від поточного стану середовища і поточних дій агентів:

$$p\left(s_{t+1} = s' \mid (s_t, u_t), t = 0,1,2,...,t\right) = p\left(s_{t+1} = s' \mid s_t, u_t\right).$$

У кожен момент часу t середовище перебуває у стані $s \in S$ і агент вибирає та реалізує дію $u \in U(s)$. В окремих випадках, для спрощення, можна прийняти, що $U(s) = U, \forall s \in S$. На основі вибору $u \in U$ середовище змінює свій стан згідно з розподілом імовірностей $p(s,u)$ і агент отримує випадковий виграш r. Агент взаємодіє з недетермінованим середовищем, модель якого, в загальному випадку, йому не відома. Агенту доступні для спостереження лише поточні стани середовища та власні стани і стратегії поведінки.

Функція розподілу станів середовища p набуває значення на відрізку $[0,1]$:

$$\forall s \in S, \forall u \in U \quad \sum_{s' \in S} p(s,u,s') = 1,$$

де s – поточний стан системи, s' – стан у наступний момент часу.

Марківське прийняття рішень полягає у перетворенні станів системи на дії агента на основі функції вибору стратегій:

$$p : S \to U.$$

Функція p визначає імовірності вибору стратегій $u \in U$ агентом у кожному стані середовища:

$$\forall s \in S \quad \sum_{u \in U} p(s,u) = 1.$$

Розподіл p набуває значення на відрізку $[0,1]$. Якщо $p(s,u) \in \{0,1\}$, то агент здійснює детермінований вибір варіантів рішень.

Метою агента є максимізація функції сумарних виграшів Υ за рахунок формування ефективної стратегії p:

$$E_p[\Upsilon] \to \max_p. \tag{3}$$

Цільова функція очікуваного виграшу (3) залежить від станів середовища, дій агента, функції вибору стратегій тощо, які зумовлюють стохастичну природу системи прийняття рішень. Оскільки поточний виграш є випадковою величиною, то функція очікуваного виграшу формулюється у кумулятивному вигляді, наприклад, сумарної, середньої або сумарної дисконтованої винагороди.

З перерахованих цільових функцій найчастіше у самонавчальних системах прийняття рішень застосовується функція з дисконтуванням виграшів, використання якої обумовлено результативністю та легкістю математичних перетворень.

Функція дисконтованої винагороди визначається на нескінченному відрізку часу, причому поточні виграші зважуються додатними коефіцієнтами $g \in (0,1]$, які визначають співвідношення між поточними та прогнозованими (майбутніми) виграшами:

$$\Upsilon_t = \sum_{i=0}^{\infty} g^i r_{t+i}, \tag{4}$$

де r_t – значення поточних виграшів у момент часу t.

Дисконтування поточних виграшів здійснюється за законом геометричної прогресії і при $g < 1$ забезпечує швидку стабілізацію функції очікуваного виграшу. Крім того, значення коефіцієнта g визначає характер прийняття рішень агентами. Близьке до 0 значення коефіцієнта надає перевагу виграшам на короткому відрізку часу. Навпаки, близьке до 1 значення цього коефіцієнта надає перевагу перспективним виграшам на довгому відрізку часу.

Ефективність (вартість) станів системи визначається значенням функції

$$V_p(s) = E_p[\Upsilon \mid s_0 = s], \tag{5}$$

де E_p – очікувана винагорода агента за реалізацію стратегії p, починаючи зі стану середовища s.

В основі марківських навчальних систем, які використовуються, якщо невідома функція виграшів або функції зміни станів системи, лежить рівняння Беллмана.

Обчислення (5) може бути виконано у рекурсивній формі. Враховуючи (4), після нескладних перетворень отримаємо:

$$V_p(s \mid s_t = s) = E(r_t) + g\sum_{k=0}^{\infty} g^k E(r_{t+k+1}) = E(r_t) + gV_p(s_{t+1}) = r(s,p(s)) + g\sum_{s' \in S} p(s' \mid s,p(s))V_p(s'). \quad (6)$$

де s' – можливі майбутні стани системи.

Метою агента є знаходження функції вибору стратегій p^*, яка максимізує функцію (5) для всіх станів середовища:

$$\forall p \; \forall s \in S \quad V^{p^*}(s) \geq V^p(s).$$

Для оптимальної функції вибору стратегій p^* для кожного стану $s \in S$ одержимо:

$$V_{p^*}(s) = \max_{u \in U}\left[r(s,u) + g\sum_{s' \in S} p(s' \mid s,u)V_{p^*}(s') \right]. \quad (7)$$

Якщо агенту відомі значення функції виграшів та функції переходів, то обчислення оптимального значення функції вибору стратегій p^* може бути виконане методами динамічного програмування [27].

Проблема ускладнюється, коли значення функцій виграшів або переходів невідоме. Тоді агент потребує взаємодії з середовищем для визначення оптимальної стратегії поведінки. Агент повинен ідентифікувати функції виграшів та переходів, взаємодіючи з середовищем, а потім використати вираз (7) для визначення оптимальної стратегії поведінки. Такий підхід називається заохочувальним навчанням, основаним на моделі середовища.

З іншого боку, агент може навчитися оптимальній стратегії поведінки без знання або попередньої ідентифікації функції виграшів або функції переходів. Такий підхід називається заохочувальним навчанням без моделі середовища.

В умовах невизначеності функція $V(s)$ може бути обчислена методом її поновлення на основі нагромадження даних у кожній ітерації взаємодії агента з середовищем:

$$V_{t+1}(s_t) = (1 - b_t)V_t(s_t) + b_t\left[r_{t+1} + gV_t(s_{t+1}) \right], \quad (8)$$

де $b_t \in [0,1]$ – параметр, який визначає швидкість навчання.

Метод (8) відомий у літературі під назвою *Temporal Difference Learning* або *TD*(0) [7]. Головною ідеєю цього методу є наближення функції $V(s_t)$ у напрямку бажаного значення $r_{t+1} + gV_t(s_{t+1})$. Інакше кажучи, метод враховує реакцію середовища на один крок вперед.

Метод *TD*(l) будується на основі врахування декількох попередніх кроків:

$$\forall s \in S \quad V_{t+1}(s) = V_t(s) + b_t\left[r_{t+1} + gV_t(s_{t+1}) - V_t(s_t) \right]e(s), \quad (9)$$

де $e(s)$ – коефіцієнт відповідності (важливості) стану.

Значення коефіцієнта $e(s)$ пропорційне до кількості відвідувань стану s в минулому:

$$e(s) = \sum_{k=1}^{t} (lg)^{t-k} c[s_t = s_k],$$

де $l \in [0,1]$; $c[s_t = s_k] \in \{0,1\}$ – індикаторна функція події.

Коефіцієнт $e(s)$ можна обчислити у реальному часі:

$$e(s) = lg\, e(s) + c(s_t = s_t).$$

Іншим підходом до заохочувального навчання є метод *Q*-навчання (reinforcemant Q-learning) [5 – 7]. Для цього використовується спеціально побудована Q-функція середніх виграшів, яка визначає ціну дії – сумарний виграш агента, який у стані s вибрав дію u:

$$Q_p(s,u)=E_p\left[\Upsilon\middle|s_0=s,u_0=u\right], \tag{10}$$

де $Q_p(s,u)$ є табличною функцією значень варіантів дій u у станах s. На відміну від функції $V(s)$, яка визначена у просторі станів середовища, функція $Q(s,u)$ визначена у просторі станів та дій агентів. Це робить зручним її використання у системах керування, коли критеріальна функція визначається для кожної пари стан-дія.

Аналогічно до (6) отримаємо:

$$Q_p(s,u)=r(s,u)+g\sum_{s'\in S}p(s'\mid s,u)V_p(s'). \tag{11}$$

Для оптимальної функції вибору стратегій p^* для кожного стану $s\in S$ одержимо:

$$Q_{p^*}(s,a)=r(s,u)+g\sum_{s'\in S}p(s'\mid s,u)V_{p^*}(s'), \tag{12}$$

де

$$p^*(u\mid s)=\arg\max_{u\in U}Q_{p^*}(s,u). \tag{13}$$

Дотримання принципу оптимальності Беллмана забезпечує оптимальний виграш агента з досягнутого поточного стану $s\in S$ в усі майбутні моменти часу. Застосування цього принципу для усіх станів забезпечує досягнення глобального оптимального розв'язку.

Оптимальні стратегії у вигляді (13) визначаються для детермінованих середовищ. В умовах невизначеності розвідування простору стан-дія системи прийняття рішень може бути виконано на основі випадкового розподілу, пропорційно до значень функцій $Q(s,u)$.

Одним із варіантів імовірнісного обчислення стратегій є використання "e-жадібного" алгоритму випадкового вибору:

$$\forall u'<>u\ \ p(u'\mid s)=e<<1;\ p(u\mid s)=1-e,\ u=arg\max_a Q(x,a).$$

Імовірність вибору агентом дії u, якщо середовище перебуває у стані s, інакше може бути визначена так:

$$p(u\mid s)=\frac{Q(s,u)+e}{\sum_a Q(s,a)+|U_s|e}, \tag{14}$$

де $Q(s,a)\ge 0$; $|U_s|$ – потужність множини (кількість) варіантів дій агента у стані x; $0<e<<1$ – зміщення, необхідне для невиродженості відношення (14) при $Q(s,a)=0\ \ \forall a\in U_s$.

Замість (14) можна використати розподіл Больцмана:

$$p(u\mid s)=\frac{e^{Q(s,u)/T}}{\sum_a e^{Q(s,a)/T}},$$

де T – температурний параметр системи. Для великих значень T реалізується близький до рівномірного випадковий розподіл. Для малих значень T реалізується розподіл, близький до "жадібного" вибору дій агентів.

Метод заохочувального навчання здійснює оптимізацію стратегій прийняття рішень у реальному масштабі часу [7]. Метод Q-навчання не вимагає наявності моделі середовища прийняття рішень, оскільки здійснює числову ідентифікацію середовища у просторі стан-дія:

$$Q_{t+1}(s_t,u_t)=(1-b_t)Q_t(s_t,u_t)+b_t\left[r_{t+1}+g\max_{a\in U}Q_t(s_{t+1},a)\right]. \tag{15}$$

Метод (15) виконує додаткову операцію максимізації Q-функції у стані s_{t+1}. Оскільки Q-функція (10) залежить від стратегій p, то її оптимізація без значних труднощів можлива для стаціонарних стратегій. На відміну від (15), наступний метод SARSA-навчання (State – Action – Reward – new State – new Action) оперує з поточними значеннями стратегій і може використовуватися для нестаціонарних стратегій:

$$Q_{t+1}(s_t,u_t)=(1-b_t)Q_t(s_t,u_t)+b_t\left[r_{t+1}+g\,Q_t(s_{t+1},u_{t+1})\right]. \qquad (16)$$

Застосування методу (16) накладає додаткові обмеження на шукані стратегії. Так, цей метод забезпечує оптимальність Q-функції, якщо стратегії p в асимптотиці часу забезпечують оптимальний вибір дій. Однак застосування методу (16) забезпечує переваги у задачах прогнозування та оптимального керування.

На початковому відрізку часу агент є ненавченим і його стратегії повинні забезпечувати можливість максимально глибокого розвідування пошукового простору, який визначається станами середовища та можливими діями агента. Це забезпечує можливість отримання в ході навчання глобального оптимального значення Q-функції. Навчений агент реалізує "жадібний" алгоритм поведінки, вибираючи дії, які відповідають максимальним значенням Q-функції, що зменшує час розв'язування оптимізаційної задачі.

Аналогічно до (9) можна побудувати $Q(l)$-метод:

$$\forall s\in S,\forall u\in U\quad Q_{t+1}(s,u)=Q_t(s,u)+b_t\left[r_{t+1}+g\,Q_t(s_{t+1},u_{t+1})-Q_t(s_t,u_t)\right]e(s,u), \qquad (17)$$

$$e(s,u)=l\,g\,e(s_{t-1},u_{t-1})+c(s=s_t\wedge u=u_t).$$

За методами (8), (9), (15) – (17) виконують обчислення оптимальних середніх виграшів на основі поточних стимулів і фактично вони є методами теорії стохастичної апроксимації. Умови їх збіжності визначаються положеннями цієї теорії [29]. Так, при $t\to\infty$ функція $Q_{t+1}(s_t,u_t)$ збігається до оптимального значення

$$Q^*(s,u)=V(s)=\sum_u p(u\mid s)\max_{a\in U}Q(s,a)$$

у разі дотримання умов стохастичної апроксимації для спадних невід'ємних послідовностей величин $b_t=t^{-l}$ $(l>0)$:

$$\sum_{t=0}^{\infty}b_t=\infty,\qquad \sum_{t=0}^{\infty}b_t^2<\infty.$$

Методи (8), (9), (15) – (17) є базовими для побудови інших методів навчання у системах прийняття рішень.

Алгоритми заохочувального навчання

Відомі два основні алгоритми для обчислення оптимальних стратегій поведінки агентів – ітерація за стратегіями (метод послідовного наближення у просторі стратегій) та ітерація за критеріями (метод послідовних наближень у просторі функцій) [5 – 7].

Ітерації за стратегіями ґрунтуються на оцінюванні поточного значення стратегії, яке потім вдосконалюють, використовуючи алгоритм "жадібної" оптимізації [28]. Суть цього алгоритму полягає у прийнятті на кожному етапі локально-оптимальних рішень, вважаючи, що результуюче рішення буде глобально-оптимальним. Ітерації за критеріями ґрунтуються на послідовних наближеннях значення функції і при цьому немає необхідності у повторному обчисленні її точного значення.

Алгоритм методу ітерації за стратегіями зводиться до такої послідовності кроків:

1) задати $t=0$; ініціалізувати функції виграшів V_t, наприклад, $V_t(s)=0$; ініціалізувати функцію вибору стратегій $p(s)\in U(s)\ \ \forall s\in S$;

2) для всіх $s\in S$ виконати ітерації

$$V_{t+1}(s)=r(s,p(s))+g\sum_{s'\in S}p(s'\mid s,p(s))V_t(s')$$

у моменти часу $t:=t+1$, поки не виконається умова точності $|V_{t+1}(s)-V_t(s)|\le e$;

3) запам'ятати поточне значення функції $p_{old}(s)=p(s),\ \forall s\in S$;

4) для всіх $s\in S$ знайти уточнене значення

$$p(s)=\arg\max_{u\in U}\left[r(s,u)+g\sum_{s'\in S}p(s'\mid s,u)V_{t+1}(s')\right];$$

5) якщо $p_{old}<>p$, то перейти на крок 2.

Алгоритм ітерації за критеріями складається з таких кроків:

1) для всіх $s\in S$ виконати кроки 2 – 6;

2) задати $\Delta=0$, $t=0$; ініціалізувати функції виграшів $V_t(s)$ довільними значеннями, наприклад, $V_t(s)=0$;

3) присвоїти $v=V_t(s)$ і для всіх $u\in U$ виконати

$$Q(s,u)=r(s,u)+g\sum_{s'\in S}p(s'\mid s,u)V_t(s');$$

4) визначити $V_{t+1}(s)=\max_{u\in U}Q(s,u)$;

5) обчислити $\Delta=\max(\Delta,|v-V_{t+1}(s)|)$;

6) якщо $\Delta\ge e$ (поки функція вибору стратегій є не добре визначеною), то задати наступний момент часу $t:=t+1$ і перейти на крок 3.

7) Вивести значення функції вибору стратегій p для кожного $s\in S$:

$$p(s)=\arg\max_{u\in U}Q(s,u).$$

У реальних задачах великий розмір пошукового простору призводить до значного зростання часу навчання. Якщо цей час є критичним, то, із раціональних міркувань, виконують редукцію пошукового простору, обмежуючись пошуком у локальних областях. Загалом це приводить до отримання субоптимальних розв'язків задачі.

Заохочувальне навчання з нечіткою логікою

Заохочувальне *Q*-навчання з нечіткою логікою (fuzzy *Q*-learning) є розширенням традиційного методу *Q*-навчання. Метод навчається на прикладах і реалізує модель експертної апроксимації знань конкретної предметної області. Знання експертів можуть бути закладені у параметри моделі навчання. Завдяки застосуванню лінгвістичних змінних та багатократному повторенню продукційних правил з нечіткою логікою моделюється процес послідовного "роздумування" експертів для досягнення сформульованої мети.

Структура системи прийняття рішень на основі навчання з нечіткою логікою зображена на рис. 2.

Згідно з [20, 24] виходи правил задають значення керуючих дій:

$$R_i: if\ x_1\ is\ A_{i,1}\ and\ ...and\ x_n\ is\ A_{i,n}\ then\ y=U^i(k), \tag{18}$$

де $U^i(k)$ – керуюча дія з номером k.

У (18) if-частина виконується блоком нечіткого логічного виведення, а then-частина виконується блоками навчання та вироблення рішень.

Модель *Q*-навчання з нечіткою логікою будується на основі таких базових положень.

На вхід системи надходить вектор числових параметрів $X = (x_1, x_2, ..., x_n)$, які перетворюються (фазифікуються) на нечіткі значення згідно із їхніми функціями належності.

Фазифіковані вхідні значення перетворюються на вихідні дії за допомогою бази правил $R = (R_1, R_2, ..., R_N)$. Правила, функція належності яких набуває ненульового значення для вхідного значення X, утворюють множину активованих правил $A(X)$.

Кожному виходу правила R_i відповідає набір дискретних дій $U^i = (u_1^i, u_2^i, ..., u_K^i)$.

У кожному правилі дія $u_k^i = U^i(k)$ з номером k характеризується вагою $w^i(k)$, значення якої корегується в процесі навчання. У результаті кожному правилу R_i відповідає ваговий вектор w^i.

Нехай s_t – стан системи у момент часу t; $A(X_t(s_t))$ – множина активованих правил; $a_{R_i}(X_t(s_t))$ – рівень істинності правила R_i після спостереження і застосування X_t.

Агент вибирає дискретні дії на основі визначених для кожного правила ваг набору можливих дій U^i. Процедура вибору дій може бути детермінованою або стохастичною.

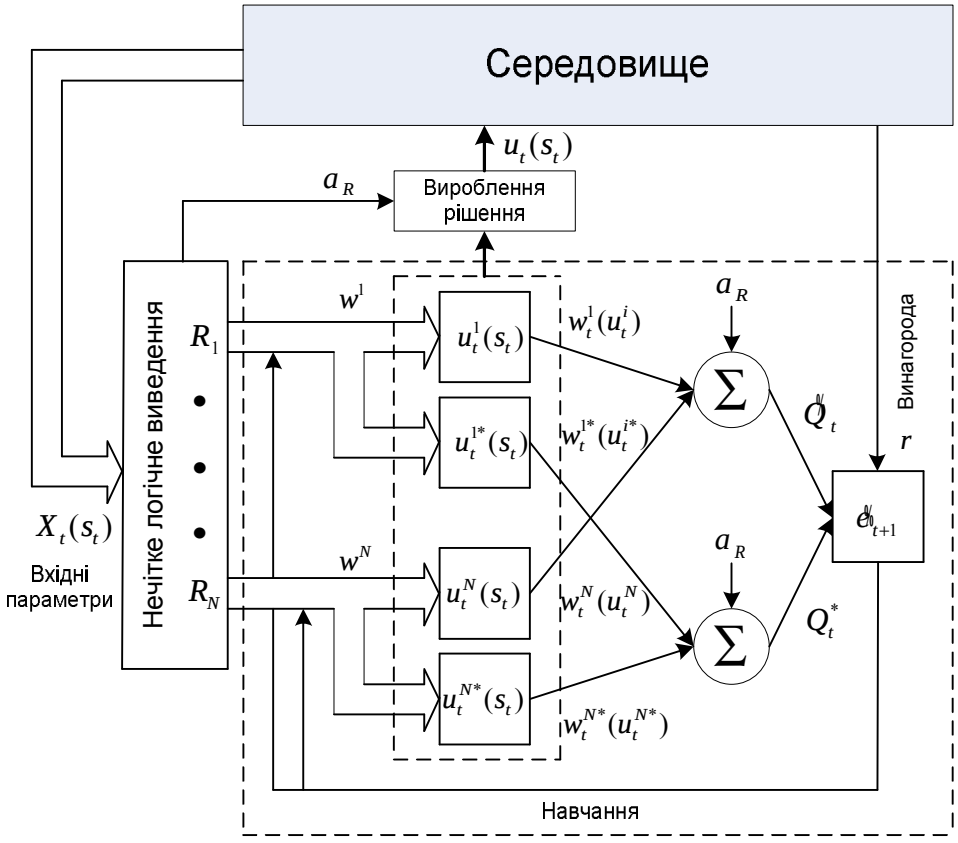

Рис. 2. Структура системи Q-навчання з нечіткою логікою

Детермінований вибір здійснюється за допомогою конкурентного механізму розвідування–експлуатації (Exploration–Exploitation, EE) [20]. В основу механізму покладено двоетапний метод e -жадібного вибору дій.

Перший етап полягає в оптимальному локальному виборі дії у кожному активованому правилі:

$$u_t^i = U^i(k) \,|\, EE(U^i(k)) = \max_{a \in U^i} \left\{ w_t^i(a) + \frac{q}{e^{n_t(a)}} \right\}, \qquad (19)$$

де $q > 0$ – коефіцієнт розвідування станів системи; $n_t(a)$ – частота використання дії a до моменту часу t. Складова $\frac{q}{e^{n_t(a)}}$ працюватиме більшою мірою на початковому відрізку часу (фаза розвідування), коли ваги $w_t^i(a)$ варіантів дій є недостатньо сформованими (система не навчена). З часом значення цього параметра зменшується за експоненційним законом і зростає роль сформованих ваг варіантів дій (фаза експлуатації).

Другий етап полягає у конкурентному виборі дії серед усіх активованих правил для формування результуючої дії:

$$u_t(s_t) = u_t^{i^*} \mid EE(u_t^{i^*}) = \max_{R_i \in A(X_t)} \left\{ EE(u_t^i a_{R_i}(X_t(s_t))) \right\}, \tag{20}$$

де $u_t^{i^*}$ – оптимальна результуюча дія у момент часу t.

У разі неперервної керуючої дії виходи системи можуть бути обчислені так:

$$u_t(s_t) = \sum_{R_i \in A(X_t)} a_{R_i}(X_t(s_t)) u_t^i \ .$$

Стохастичний вибір дій може бути рівноімовірним або ефективнішим, адаптивним чи побудованим на основі подібних до (14) раціональних евристик. Застосування випадкових стратегій особливо доцільне у разі стохастичної природи середовища прийняття рішень. Як правило, стохастичний вибір стратегій вимагає більше кроків для прийняття оптимальних рішень порівняно з детермінованим вибором.

На першому етапі стохастичного вибору необхідно для кожного правила $R_i (i = 1..N)$ отримати вектор стратегій p_i^K, здійснивши проектування вектора $w^i = \left(w_t^i(a) + \frac{q}{e^{n_t(a)}}, \forall a \in U \right)$ на одиничний K-вимірний симплекс [30]. На основі стратегій p_i^K виконати рандомізований вибір поточної дії для кожного правила:

$$u_t^i = U^i(k) \mid k = \min_{j=1}^{K} \left(j \left| \sum_{l=1}^{j} p_i^K(l) \ge w \right. \right), \ i = 1..N, \tag{21}$$

де $w \in [0,1)$ – дійсне випадкове число з рівномірним законом розподілу.

На другому етапі стохастичного вибору необхідно обчислити вектор стратегій p_i^N методом проектування вектора $v = \left(w(u_t^i) a_{R_i}(X_t(s_t)), \ i = 1..N \right)$ на одиничний N-вимірний симплекс. Після цього необхідно виконати рандомізований вибір результуючої дії:

$$u_t(s_t) = u_t^i(k) \mid k = \min_{j=1}^{N} \left(j \left| \sum_{l=1}^{j} p_i^N(l) \ge w \right. \right) \tag{22}$$

З кожною парою стан–дія пов'язується функція якості $\overset{N}{Q}_t(s_t, u_t(s_t))$. Поточне значення Q-функції для поточної пари стан–дія $\overset{N}{Q}_t(s_t, u_t(s_t))$ обчислюється так:

$$\overset{N}{Q}_t(s_t, u_t(s_t)) = \frac{\sum_{R_i \in A(X_t)} a_{R_i}(X_t(s_t)) w_t^i(u_t^i)}{\sum_{R_i \in A(X_t)} a_{R_i}(X_t(s_t))}, \tag{23}$$

де u_t^i – e-жадібна дія для правила R_i у момент часу t; $u_t(s_t)$ – загальна подвійна e-жадібна або максимальна e-жадібна дія.

Після реалізації дії $u_t(s_t)$ агент переходить у стан $s' = s_{t+1}$, де спостерігає поточну винагороду r_{t+1}. У стані s_{t+1} агент здійснює локальне розвідування середовища й обчислює поточне оптимальне значення Q-функції:

$$V_t^*(s_{t+1}) = \frac{\sum_{R_i \in A(X_t)} a_{R_i}(X_t(s_{t+1})) \max_{a \in U^i} w_t^i(a)}{\sum_{R_i \in A(X_t)} a_{R_i}(X_t(s_{t+1}))}. \quad (24)$$

На основі $\hat{Q}_t(s_t, u_t(s_t))$, $V_t^*(s_{t+1})$ та r_{t+1} знаходять значення TD-помилки:

$$\hat{e}_{t+1} = r_{t+1} + gV_t^*(s_{t+1}) - \hat{Q}_t(s_t, u_t(s_t)), \quad (25)$$

де $0 \le g \le 1$ – фактор дисконтування виграшів.

Значення помилки $\hat{e}_{t+1}$ використовується для модифікації елементів векторів ваг дій.

Для правила навчання $TD(0)$ ваги дій змінюються так:

$$w_{t+1}^i(u_t^i) = w_t^i(u_t^i) + b_t \hat{e}_{t+1} a_{R_i} \qquad \forall R_i \in A(X_t) \quad (26)$$

Для правила навчання $TD(l)$:

$$w_{t+1}^i = w_t^i + b_t \hat{e}_{t+1} e_t^T, \ \forall R_i,$$

де b_t – вектор, який визначає швидкість навчання для правил; e_t – матриця відповідності ваг дій:

$$e_t^a(U^i(k)) = \begin{cases} l_a e_{t-1}^a(U^i(k)) + a_{R_i}, \ if \ U^i(k) = u_t^i \\ l_a e_{t-1}^a(U^i(k)), \qquad if \ U^i(k) <> u_t^i \end{cases}, \ \forall U^i(k) \ \forall R_i, \quad (27)$$

де $l_a \in [0,1]$ – фактор оновлення.

Параметри збіжності методу можна оцінити за допомогою різницевого варіанта критеріальної функції

$$\Delta_t = |S|^{-1} \sum_{s \in S} |V_{t+1}(s) - V_t(s)|, \quad (28)$$

де $|S|$ – потужність множини станів системи.

Алгоритм навчання з нечіткою логікою

Нехай у момент часу t агент, перебуваючи у стані s_t, виконав дію $u_t(s_t)$ й отримав винагороду r_{t+1} у момент часу $t+1$.

Алгоритм вибору варіантів дій складається з такої послідовності кроків.

1. Ініціалізувати w, Q, b, g, e, t.
2. Задати $t := t+1$.
3. Ініціалізувати $s_t = s_{init}$.
4. Отримати вектор параметрів $X_t(s_t)$ від середовища.
5. Перетворити числові вхідні параметри X_t на нечіткі значення.
6. Для кожного правила розрахувати рівень істинності $a_{R_i}(X_t(s_t))$, $i = 1..N$.
7. Згідно з (23) обчислити та запам'ятати Q-функцію $\hat{Q}_t(s_t, u_t(s_t))$ для поточної пари стан–дія s_t та $u_t(s_t)$.

8. Виконати детермінований або стохастичний вибір дії. Для детермінованого вибору виконати процедуру подвійного e -жадібного вибору дії u_t (19), (20), для стохастичного вибору – процедуру проектування ваг на одиничний симплекс та рандомізований вибір результуючої дії u_t (21), (22).
9. Виконати дію u_t, оцінити стан s_{t+1} та отримати винагороду r_{t+1}.
10. Обчислити рівні істинності $a_{R_i}(X_t(s_{t+1}))$, $i = 1..N$.
11. Оцінити $V^*(s_{t+1})$ – оптимальне значення Q-функції у стані s_{t+1} (24).
12. Обчислити TD-помилку згідно з (25).
13. Знайти параметр швидкості навчання $b_t = t^{-l}$ $(l > 0)$.
14. Обчислити ознаки відповідності дій e_t згідно з (27).
15. Модифікувати вектори ваг дій w_{t+1} згідно з (26).
16. Задати $s_t = s_{t+1}$.
17. Якщо $s_t \neq s_{end}$, перейти на крок 4.
18. Обчислити точність навчання Δ_t (28).
19. Якщо $\Delta_t \geq e$ (або $t \leq t_{end}$), то перейти на крок 2.
20. Вивести значення функцій $Q_t(s,u)$ та $V_t(s)$ $\forall s \in S$, $\forall u \in U$.

Результати моделювання

Для прикладу розглянемо задачу пошуку автономним агентом максимального значення функції X у клітинному просторі S за мінімальну кількість кроків. Клітинний простір задається у вигляді зображеного на рис. 3 графа переходів з вершинами $s = \{i*4+j\}$, $i = \overline{0,3}$, $j = \overline{0,3}$.

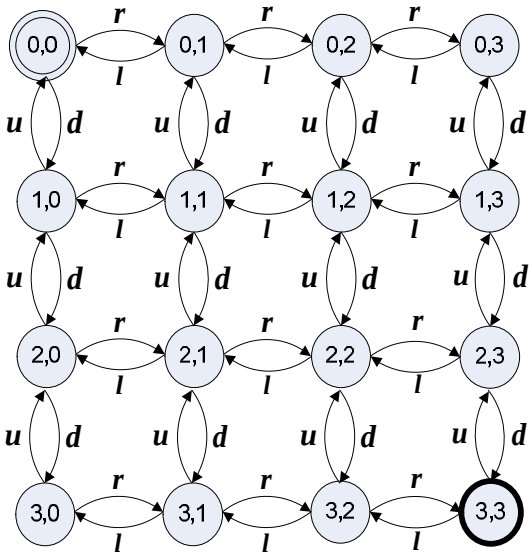

Рис. 3. Простір станів клітинного середовища

Нехай функція $X = (x(s) | \forall s \in S)$ набуває дискретних значень $x(s)$ у кожному стані $s = (i, j)$ простору S:

$$x_{i,j} = (i + j) * h,$$

де $h = 10$ – крок приросту функції. У загальному випадку X може бути векторною функцією.

У моменти часу $t = 1, 2, \ldots$, перебуваючи у стані s, агент вибирає один із варіантів дій $u \in U = \{right, left, down, up\}$, прямуючи від початкового $s_{init} = (0,0)$ до поглинаючого стану $s_{end} = (3,3)$ з максимальним значенням функції X.

Агент здійснює вибір дій на основі моделі Q-навчання з нечіткою логікою. Спостерігаючи у кожному стані $s \in S$ значення функції $x(s)$, агент здійснює її фазифікацію на основі трапецієподібних функцій належності $m = m(x, x1, x2, x3, x4)$:

$$m = \begin{cases} \max(0, (x_4 - x)/(x_4 - x_3)), & x > x_3; \\ \max(0, (x - x_1)/(x_2 - x_1)), & x < x_2; \\ 1, & x \geq x_2 \; and \; x \leq x_3. \end{cases}$$

Якщо входи X складаються тільки з одного елемента, правила логічного виведення $R = (R_i \mid i = 1..3)$ (18) спрощуються до такого визначення рівнів істинності вхідного значення $x = x(s)$:

$$a(R_1) = m(x, -20, 0, 10, 30);$$

$$a(R_2) = m(x, 0, 20, 30, 50);$$

$$a(R_3) = m(x, 20, 40, 50, 70).$$

У кожному стані за вибрану дію агент отримує поточне заохочення, сформоване за правилом:

$$r(s_t, u_t) = \begin{cases} -h, & x(s_{t+1}) < x(s_t) \\ 0, & x(s_{t+1}) = x(s_t) \\ h, & x(s_{t+1}) > x(s_t) \end{cases}.$$

Оцінювання Q-функції виконано методом ітерації за критеріями з точністю $e = 10^{-6}$ для значень параметрів $b = 1$, $g = 0.5$, $q = 1$.

Результати розв'язування детермінованої задачі зображено на рис. 4. У вершинах графа подано оптимальне значення функції вартості станів $V_{p^*}(s)$, а біля ребер графа – значення функції ефективності дій $Q_{p^*}(s,u)$ агента.

Щоб знайти оптимальну стратегію p^*, яка забезпечує перехід від стану s_{init} до стану s_{end} по найкоротшому шляху, необхідно для кожного стану $s \in S$ знайти дію $u \in U$, яка максимізує значення критерію $Q(s,u)$. Для детермінованої задачі оптимальну стратегію визначають згідно із (13). Як видно із рис. 4, сформульована задача має декілька оптимальних розв'язків.

Результати розв'язування оптимізаційної задачі зі стохастичним вибором варіантів дій агента подано на рис. 5 у вигляді графіків усередненої у часі різницевої функції:

$$\overline{\Delta}_t = t^{-1} \sum_{t=0}^{t} \Delta_t \; .$$

Як правило, стохастичний вибір стратегій вимагає більше кроків для прийняття оптимальних рішень порівняно з детермінованим вибором.

Графіки рис. 5 зображено у логарифмічному масштабі. Графік з номером $i = 1..3$ відповідає клітинному середовищу розміром $2^{i+1} \times 2^{i+1}$.

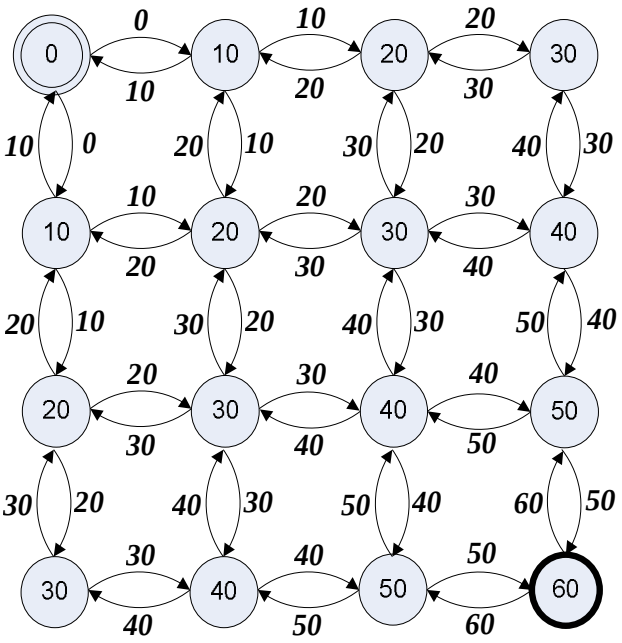

Рис. 4. Розраховані значення функцій $V_{p^*}(s)$ *та* $Q_{p^*}(s,u)$

Зменшення значення різницевої функції у часі свідчить про збіжність методу випадкового вибору стратегій. Швидкість збіжності випадкового процесу $\overline{\Delta}_t$ характеризується порядком q та величиною J, які перебувають у залежності [29]:

$$\overline{\lim_{n\to\infty}} n^q M\{\overline{\Delta}_t\} \le J < \infty .$$

Більшому q та меншому J відповідає більша швидкість збіжності методу навчання. Порядок швидкості збіжності можна оцінити пропорційно до кута нахилу лінійної апроксимації функції Δ_t з віссю часу. Величина швидкості збіжності визначається зміщенням по осі ординат. Як видно з рис. 5, зі збільшенням кількості станів системи швидкість збіжності досліджуваного методу зменшується.

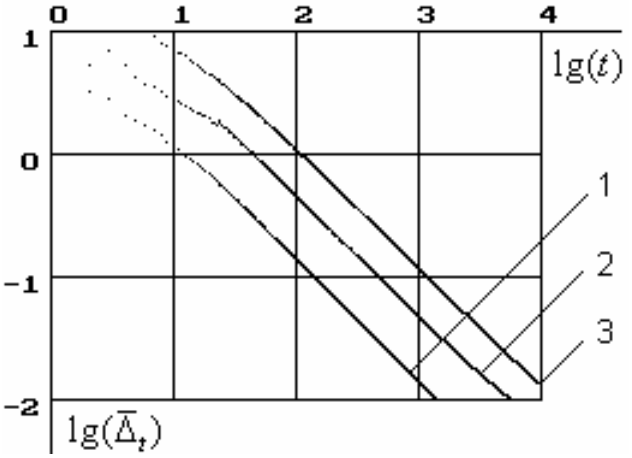

Рис. 5. Збіжність стохастичного методу Q *-навчання*

Кількість ітерацій, необхідних для знаходження оптимальних розв'язків сформульованої задачі, є пропорційною до розмірності клітинного простору. Враховуючи можливу багатоваріантність розв'язків задачі, кількість пошукових ітерацій може бути скорочена за рахунок неповного розвідування станів системи.

Висновки

1. Приймати рішення в умовах невизначеності доцільно з урахуванням проблемно-орієнтованих експертних знань, поданих у вигляді правил нечіткого логічного виведення на основі семантичних моделей штучного інтелекту.
2. У разі відсутності адекватної математичної моделі системи використовується її числова ідентифікація у просторі стан–дія на основі методів динамічного заохочувального навчання.
3. Збіжність методу прийняття рішень залежить від структури і змісту правил нечіткого логічного виведення та від співвідношення і динаміки коефіцієнтів, які входять у модель заохочувального навчання.
4. Досягнутий методом навчання оптимальний розв'язок задачі є стійким до механізму випадкового вибору стратегій, про що свідчить експериментально підтверджене зменшення критеріальної різницевої функції у часі.
5. Отримані результати та розроблене алгоритмічне і програмне забезпечення можуть бути адаптовані для розв'язування практичних задач оптимального прийняття рішень в умовах невизначеності.

1. Трухаев Р. И. Модели принятия решений в условиях неопределенности / Р. И. Трухаев. – М.: Наука, 1981. – 258 с. 2. Орловский С. А. Проблемы принятия решений при нечеткой исходной информации / С. А. Орловский. – М.: Наука, 1981. – 208 с. 3. Алтунин А. Е. Модели и алгоритмы принятия решений в нечетких условиях: Монография / А. Е. Алтунин, М. В. Семухин. – Тюмень: ТГУ, 2000.– 352 с. 4. Mitchell T. M. Machine Learning / T. M. Mitchell. – New York: McGraw-Hill, 1997. – 414 pp. 5. Watkins, C.J.C.H. Q-Learning / C.J.C. H. Watkins, P. Dayan // Machine Learning, No. 8. – Kluwer Academic Publishers, Boston. – 1992. – PP. 279–292. 6. Kaelbling L. P. Reinforcement Learning: A Survey / L. P. Kaelbling, M. L. Littman, A. W. Moore // Journal of Artificial Intelligence Research. – 1996. – No. 4. – P. 237 – 285. 7. Sutton R. S. Reinforcement Learning: An Introduction / Richard S. Sutton, Andrew G. Barto. – MIT Press, 1998. – 322 pp. 8. Кравець П. О. Марківські методи навчання у системах прийняття рішень / П. О. Кравець, О. М. Проданюк // Інформаційні системи та мережі: Вісник Нац. ун-ту "Львівська політехніка". – 2008. – № 631. – С. 166 – 177.9. Wooldridge, M. An Introduction to Multiagent Systems / M. Wooldridge. – John Wiley & Sons (Chichester, England), 2002. – 366 pp. 10. Weiss G. Adaptation and Learning in Multiagent Systems / Gerhard Weiss, Sandip Sen, editors. – Berlin: Springer Verlag, 1996. – 585 pp. 11. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений / Л. Заде. – М.: Мир, 1976. – 165 с. 12. Zimmerman H. J. Fuzzy Set Theory and Its Applications / H. J. Zimmerman. – Kluwer, Dordrecht, 1991. – 315 p. 13. Нечеткая логика: алгебраические основы и приложения: Монография / С. Л. Блюмин, И. А. Шуйкова, П. В. Сараев, И. В. Черпаков. – Липецк: ЛЭГИ, 2002. – 113 с. 14. Борисов А. Н. Принятие решений на основе нечетких моделей. Примеры моделей / А. Н. Борисов, О. А. Крумберг, И. П. Федоров. – Рига: Зинатне, 1990.– 184 с. 15. Berenji H. R. Refinement of approximate reasoning-based controllers by reinforcement learning / H. R. Berenji // Proceedings of the Eight International Workshop Machine Learning. – 1991. – PP. 475–479. 16. Асаи К. Прикладные нечеткие системы [пер.с японского] / Под ред. Т. Тэрано, К. Асаи, М. Сугэно. – М.: Мир, 1993. – 368 с. 17. Sugeno M. Industrial applications of fuzzy control / M. Sugeno, ed. – North-Holland,

Amsterdam, 1985. – 269 p. 18. Theoretical aspects of fuzzy control / H. T. Hguen, M. Sugeno, R. Tong, R. R. Yager. – New York, John Wiley & Sons, 1995. – 359 p. 19. Berenji H. R. Fuzzy Q-learning for Generalization of Reinforcement Learning / H. R. Berenji // Proceedings of the fifth IEEE International Conference on Fuzzy Systems. – New Orleans, Louisiana. – September 1996. – PP. 2208 – 2214. 20. Glorennec P. Y. Fuzzy Q-Learning / P. Y. Glorennec, L. Jouffe // Proceedings of the Sixth IEEE International Conference on Fuzzy Systems. – Barcelona, Spain. – 1997. – PP. 659-662. 21. Jouffe L. Fuzzy Inference Systems Learning by Reinforcement Methods / L. Jouffe // IEEE Transactions On Systems, Man and Cybernetics, Part C, Applications and Reviews. – 1998. – Vol.28. – No. 3. – PP. 338 – 355. 22. Ротштейн А. П. Интеллектуальные технологии идентификации: нечеткая логика, генетические алгоритмы, нейронные сети. – Винница: УНІВЕРСУМ –Вінниця, 1999. – 320 с. 23. Mudi R.K. A self-tuning fuzzy PI controller / R. K. Mudi, N. R. Pal // Int. Jo. Fuzzy sets and systems. – № 115. – 2000. – P. 327 – 378. 24. Naeeni A. F. Advanced Multi-Agent Fuzzy Reinforcement Learning. Master Thesis Computer Engineering, Nr: E3098D / Alireza Ferdowsizadeh Naeeni. – Dalarna University, Sweden, 2004. – 99 p. 25. Рутковская Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. – М.: Горячая линия–Телеком, 2004. – 452 с. 26. Штовба С. Д. Проектирование нечетких систем средствами MATLAB. М.: Горячая линия – Телеком, 2007. – 288 с. 27. Puterman M. L. Markov Decision Processes: Discrete Stochastic Dynamic Programming / M. L. Puterman. – John Wiley & Sons, New York, 2005. – 649 pp. 28. Кормен Т.Х. Алгоритмы: построение и анализ. – 2-е изд. / Томас Х. Кормен и др. – М.: Вильямс, 2006. – 1296 с. 29. Вазан М. Стохастическая аппроксимация / М. Вазан. – М.: Мир, 1972. – 295 с. 30. Назин А.В. Адаптивный выбор вариантов: Рекуррентные алгоритмы / А.В. Назин, А.С. Позняк. – М.: Наука, 1986. – 288 с.