УДК 519.7 doi: 10.15421/322214

О.П. Соф'їн, А.Є. Шевельова
Дніпровський національний університет ім. О. Гончара

РОЗРОБКА АЛГОРИТМУ ГЕНЕРАЦІЇ КЕРУЮЧИХ СКІНЧЕННИХ АВТОМАТІВ МІЛІ В ЗАДАЧІ ПРО ШТУЧНОГО МУРАХУ

Запропоновано алгоритм спільного використання генетичних алгоритмів та скінченних автоматів для розв'язання задачі про штучну мураху. Виконано обчислювальні експерименти, що демонструють ефективність цього алгоритму.

Ключові слова: скінченний автомат, генетичний алгоритм, проблема штучної мурахи, роботизоване планування шляху.

O.P. Sofin, A.E. Sheveleva
Oles Honchar Dnipro National University

DEVELOPMENT THE GENERATION ALGORITHM OF CONTROLLING MEALY MACHINE IN THE ARTIFICIAL ANT PROBLEM

Planning in the field of artificial intelligence and robotics requires finding a plan that receives information from sensors about the state of various objects in the system, and then uses this information to select a sequence of actions that change the state of objects in the system. An example in which the automatic construction of the logic of an entity with complex behavior is of crucial importance is one of the classic problems of the joint use of genetic algorithms and finite state machine – the problem of the "Artificial Ant". We have a difficult task for an artificial ant trying to follow a strictly defined, irregular, indirect and discontinuous path. Using a genetic algorithm, you need to build a Mealy machine, which represents the logic of the behavior of an artificial ant.

Two paths specially designed for the artificial ant problem were considered: John Muir Trail, Santa-Fe Trail. The ant is placed in a two-dimensional toroidal grid. The ant control system is built as a finite state machine.

To use a finite state machine as a chromosome, it must be coded. In order to build a genetic algorithm based on finite state machine, the genome must be coded into a string of bits. An encoding used in a genetic system where the finite state machine is an array of tables in which each state, input, and movement has an arbitrarily assigned binary value. To unambiguously define a finite state machine, it is enough to specify the initial state and list in canonical order the columns of the state to which the state machine goes and the output action of the table. Since the actual finite state machine of the genetic system has up to 32 states (requiring 5 bits), the actual length of the genome is 32 * 2 * (5 + 2) = 448 bits. It is worth noting that the presence of 32 states does not mean that all of them will be used. There may also be unreachable or unvisited states by the artificial ant during its simulation.

The important step in preparing to use a genetic algorithm is to define a fitness function. The usefulness of a particular 453-bit string in this problem simply comes down to how many of its ant appear in a reasonable amount of time if its actions are controlled by the finite state machine represented by the 453-bit string. The maximum number of time steps is set because we want to include Mealy machine that search all 1024 squares on the grid

using a random walk or as a mosaic pattern. That is why the limit of 200 hourly steps is set. If an ant has a "timeout", its fitness is simply the amount of food eaten up to that point. So, fitness ranges from 0 to 89.

A software implementation of the algorithm for building Mealy machine controlling the behavior of an artificial ant was developed. The result of the work of the program according to the genetic algorithm is as follows: starting from the 90-th generation, the ant began to complete the path in 200 steps.

Keywords: finite state machine, genetic algorithm, artificial ant problem, robotic path planning.

А.П. Софьин, А.Е. Шевелёва

Днипровский национальный университет имени Олеся Гончара

РАЗРАБОТКА АЛГОРИТМА ГЕНЕРАЦИИ УПРАВЛЯЮЩИХ КОНЕЧНЫХ АВТОМАТОВ МИЛИ В ЗАДАЧЕ ОБ ИСКУССТВЕННОМ МУРАВЬЕ

Предложен алгоритм совместного использования генетических алгоритмов и конечных автоматов для решения задачи искусственного муравья. Выполнены вычислительные эксперименты, демонстрирующие эффективность этого алгоритма.

Ключевые слова: конечный автомат, генетический алгоритм, проблема искусственного муравья, роботизированное планирование пути.

Вступ. Планування у сфері штучного інтелекту та робототехніки вимагає пошуку плану, який отримує інформацію від датчиків про стан різних об’єктів у системі, а потім використовує цю інформацію для вибору послідовності дій, які змінюють стан об’єктів у системі. Прикладом, в якому автоматична побудова логіки сутності зі складною поведінкою має вирішальне значення є одна з класичних задач спільного використання генетичних алгоритмів та скінченних автоматів – задача про «Штучного мураху».

Скінченні автомати – модель управління прийняттям рішень для систем із скінченною кількістю станів.

Скінченим автоматом Мілі (Mealy machine) називається шістка об'єктів: $\mathcal{A} = \langle S, X, Y, s_0, \delta, \lambda \rangle$, де:

S – скінчена непорожня множина внутрішніх станів;

X – скінчена непорожня множина вхідних символів (вхідний алфавіт);

Y – скінчена непорожня множина вихідних символів (вихідний алфавіт);

$s_0 \in S$ – початковий стан;

$\delta : S \times X \to S$ – функція переходів;

$\lambda : S \times X \to Y$ – функція виходів.

Існують як мінімум два простих способи реалізації скінченного автомата:

- діаграма станів (або граф переходів) – графічне представлення множини станів і функції переходів. Являє собою орієнтований граф. Якщо перехід зі стану s_1 в s_2 може бути здійснений по одному з декількох символів, то всі вони повинні бути над дугою діаграми;
- таблиця переходів – табличне представлення функції δ. Зазвичай в такій таблиці кожному рядку відповідає один стан і один допустимий вхідний

символ. В клітинці на перетині рядка і стовпця записується стан, в який повинен перейти автомат.

У подальшому будемо використовувати табличний спосіб представлення скінченного автомата.

Генетичні алгоритми – це сімейство алгоритмів, які використовують деякі генетичні принципи, що наявні в природі, для розв'язання конкретних обчислювальних задач. Протягом більш ніж чотирьох десятиліть генетичні алгоритми (GA) виявилися дуже потужним і також дуже загальним інструментом для розв'язання задач навчання в нейронних мережах, пошуку найкоротшого шляху, проблеми комівояжера, у стратегії ігор, в задачах, подібних до транспортної задачі, задачі визначення параметрів системи, оптимізації запитів до бази даних, хімії, біології, неврології, теорії автоматів тощо.

У роботі [1] розробили складне завдання для «штучної мурахи», який намагається пройти слідом і отримали примітне рішення у вигляді як кінцевого автомата, так і багатошарової рекурентної нейронної мережі для управління рухами мурахи. Ці дослідження були узагальнені в роботах [2-4]. Розв'язання задачі побудови скінченних автоматів повним перебором є вкрай трудомістке. Тому для побудови скінченних автоматів у задачах такого роду доцільно застосовувати генетичні алгоритми [5–11].

Постановка задачі. Маємо складне завдання для штучного мурахи, що намагається пройти строго визначену, нерегулярну, непряму і розривну стежку. За допомогою генетичного алгоритму потрібно побудувати скінченний автомат Мілі, який представляє логіку поведінки штучного мурахи.

Розглядалися дві спеціально спроектовані для задачі про штучного мураху стежки. Мураха розміщується в двовимірній тороїдальній сітці 32×32.

На рис. 1 наведено приклад «стежки Джона Мюїра» (John Muir Trail) [1]. Вона містить 89 шматочків їжі, 20 поворотів, 4 одинарні проміжки, 7 подвійних проміжків і 7 потрійних проміжків і має загальну довжину 128. Сітка є тороїдальною, тому якщо мураха відходить від краю сітки, вона знову з'являється і продовжує рух на протилежному краю.

Введено обмеження, що мураха має з'їсти всю їжу за 200 або менше ходів.

Чорним кольором зображені комірки із їжею. Порожні комірки на «стежці» зображені сірим кольором. Сірі комірки невидимі для мурахи.

«Стежка Санта-Фе» (Santa Fe Trail) – це нерегулярна звивиста стежка (рис. 2), що складається з 89 харчових гранул. Довжина траси становить 144 клітини та містить 21 поворот. Стежка Санта-Фе, розроблена Крістофером Ленгтоном [2], є дещо складнішою, ніж «стежка Джона Мюїра».

Перед першим кроком мураха знаходиться у верхній лівій клітинці і дивиться направо. У нього є сенсор, який дозволяє йому визначити, чи є їжа в клітинці, що знаходиться безпосередньо перед ним. За хід мураха може зробити одну з таких трьох дій – рухатися вперед на один крок, повернутись ліворуч, повернутись праворуч. Щоб з'їсти їжу в клітинці, необхідно потрапити до неї. Після цього клітинка з їжею вважається порожньою і стає для мурашки невідмінною від спочатку порожніх клітинок. Мураха живий протягом гри

– їжа не є необхідним ресурсом для його існування. Жодних інших персонажів, крім мурахи, на полі немає. Мураха може ходити по будь-яких клітинках поля.

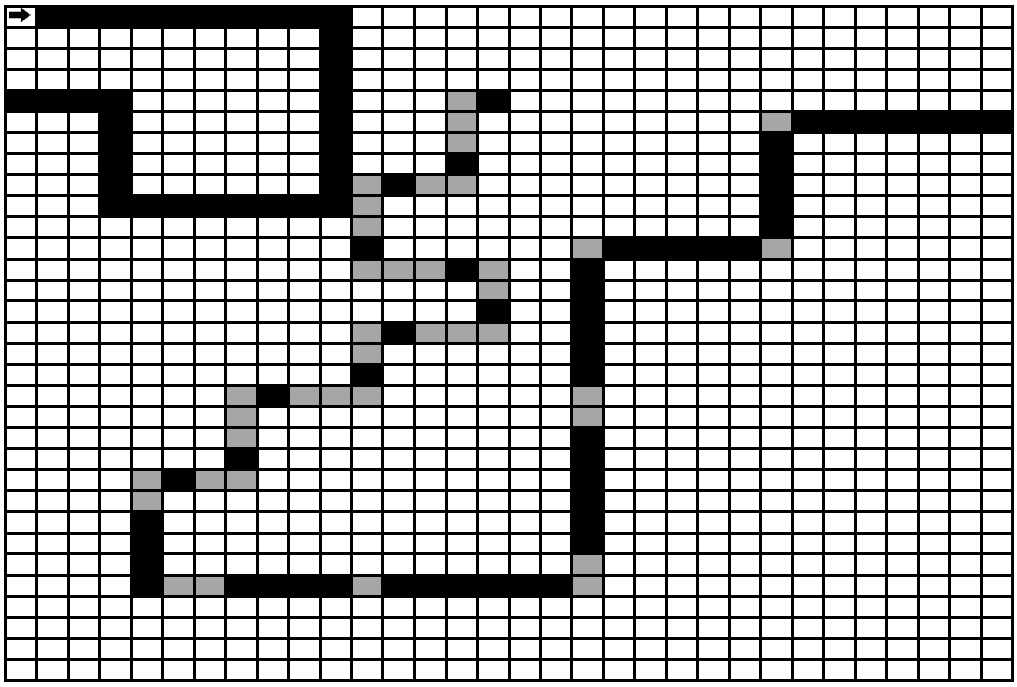

Рис.1

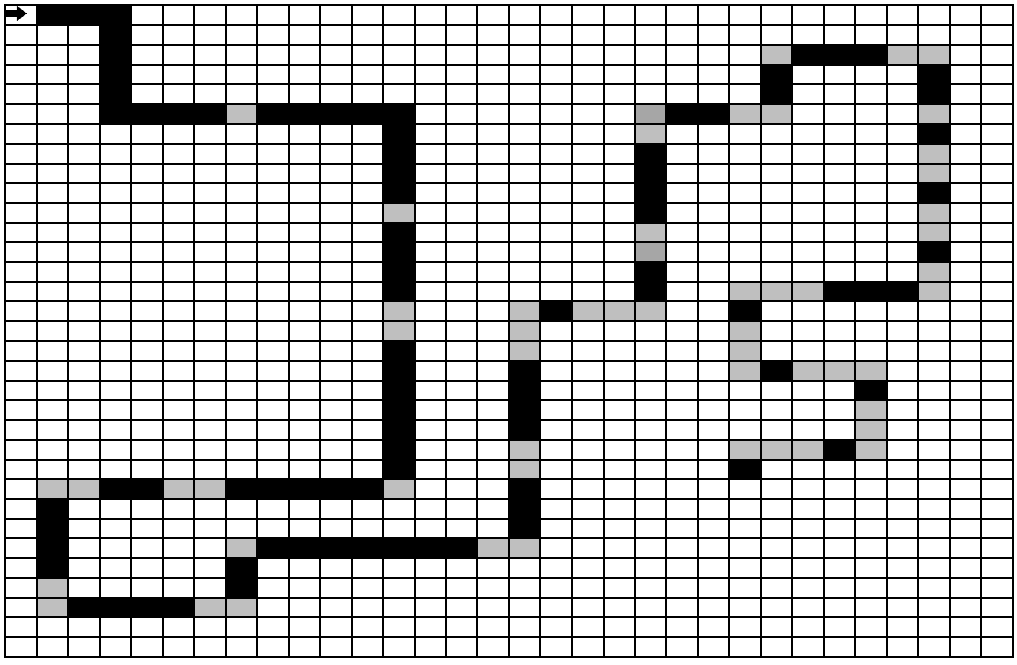

Рис. 2

Система управління мурахою будується як скінченний автомат. Такий автомат має дві вхідні події – 1 (наступна клітинка містить їжу), 0 (наступна клітинка порожня) і три вихідні дії: *L* (повернути ліворуч), *R* (повернути праворуч) і *M* (зробити крок уперед) та переходить у новий стан.

Щоб використовувати скінченний автомат як хромосому, він повинен бути закодований. Для того, щоб побудувати генетичний алгоритм на базі скінченних автоматів, геном повинен бути закодований у рядок бітів. Кодування, використане в генетичній системі, де скінченний автомат є масивом таблиці, в якій кожному стані, вводу та переміщенню є довільно призначене двійкове значення. Щоб однозначно визначити скінченний автомат, достатньо вказати початковий стан та перерахувати у канонічному порядку стовпці стану, в який переходить скінченний автомат та вихідну дію таблиці. Оскільки фактичний скінченний автомат генетичної системи має до 32 станів (для яких потрібно 5 біт), то фактична довжина геному складає 32 * 2 * (5 + 2) = 448 біт. Варто зазначити, що наявність 32 станів не означає, що всі вони будуть використані. Також можуть бути недосяжні або не відвідувані стани штучним мурахою під час його моделювання.

Другим важливим кроком у підготовці до використання генетичного алгоритму є визначення функції пристосованості (показника придатності), який визначає, наскільки добре певний рядок бере участь у розв’язанні проблеми. Придатність конкретного 453-бітного рядка в цій задачі просто полягає в тому, скільки їжі мураха з’їдає за розумний проміжок часу, якщо його діями керує скінченний автомат, представлений 453-бітним рядком. Максимальна кількість часових кроків встановлюється тому, що ми хочемо виключити автомати, які шукають усі 1024 квадрати на сітці за допомогою випадкового блукання або як мозаїчний візерунок. Тому і встановлений ліміт у 200 часових кроків. Якщо мурашка має «тайм-аут», її придатність – це просто кількість їжі, з’їденої до цього моменту. Отже, придатність коливається від 0 до 89.

Функція пристосованості визначається формулою:

$$f\left(x_i\right) = n,$$

де n – кількість знайденої їжі.

Чисельні результати та їх обговорення. Чисельні експерименти були виконані для популяції 65536 бітових рядків випадкових бітів довжиною 448 одиниць. Фактично розмір популяції є довільним, тому варто звернути увагу на баланс кількості геномів та обчислювальних можливостей конкретного комп’ютеру.

Кожен бітовий рядок декодується у скінченний автомат, і всі 65536 геному виконуються за 200 дій на окремих копіях шляху. В кінці кожного покоління всі мурахи обираються за успіхом. Мурахи, які набрали найвищі бали у своєму поколінні (від 1% до 10% відбірної фракції), відбираються для розведення (мутацій та схрещень), а всі інші відкидаються. Рядки розвиваються довільно. Тоді нове покоління виробляється за наступною процедурою.

1) Об'єднання: 65536 геномів у вигляді скінченних автоматів вибираються випадковим чином (із заміною) з обраної фракції. Не надається перевага тим мурахам, які набрали вищі показники, ніж інші у вибраній фракції.

2) Схрещення: З кожної пари будується по одному скінченному автомату випадковим чином обираючи певний біт з якогось із геномів. Імовірність ви-

бору біту становить від 0,5% до 1,0% на біт, тому середня кількість кросоверів на мурашник за покоління становить 2,25 до 4,5. Схрещування виконується у бітовому рядку.

3) Випадкова мутація: Скінченний автомат змінюється випадковими чином, заміною бітів, знову з ймовірністю від 0,1% до 1% за біт. Таким чином, середня кількість мутацій також становить від 0,225 до 2,25 за мурашку на покоління.

Було побудовано геном на базі скінченного автомату, що закодований бітовим рядком та має 32 стани (для кожного з яких потрібно 5 біт). У табл. 1 наведено кодування переходів скінченного автомату Мілі.

Таблиця переходів стану, яка описує скінченний автомат показує, що для подання хромосом потрібні лише 2 останні колонки, оскільки перша колонка – це відсортований серійний номер скінченного автомату, а значення другої колонки повторюються для кожного стану. Отже, перший стан може бути закодований так: 1001 1101, де перші 4 біти призначені для переходу, якщо значення виходу дорівнює 0, а останні 4 біти для значення виводу 1. Перші 2 біти кожного переходу описують дію, а останні 2 біти - це серійний номер наступного стану. На рис. 3 показаний результат кодованого скінченного автомату.

Таблиця 1

Таблиця переходів двійкового кодованого стану геному для скінченного автомату

Old State	Input	New State	Action
00	0	00	01
00	1	01	11
01	0	01	01
01	1	11	11
10	0	10	10
10	1	11	11
11	0	10	10
11	1	10	11

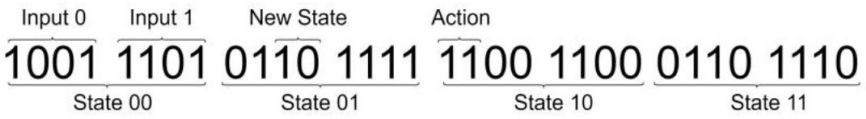

Рис. 3

Було розроблено програмну реалізацію алгоритму побудови скінченних автоматів, керуючих поведінкою штучного мурахи. Результат роботи про-

грами за генетичним алгоритмом такий: починаючи з 90-го покоління мураха почав повністю проходити шлях за 200 кроків.

Вручну порівняно нескладно побудувати скінченний автомат, який описує таку стратегію: «Бачу їжу – йду вперед. Не бачу – повертаю. Зробив коло, але їжі нема – йду вперед». Такий скінченний автомат Мілі має п’ять станів і дозволяє мурахі з'їсти 81 шматочок їжі за 200 ходів, а всі 89 – лише за 314 ходів, тобто обмеження 200 ходів не виконується [2].

У роботі [9] за допомогою генетичних алгоритмів побудовано скінченний автомат Мілі із 13 станів, який дозволяє мурахі з'їсти всю їжу за 200 ходів.

У роботі [2] побудовано скінченний автомат Мілі із 11 станів, який дозволяє з'їсти всю їжу за 193 ходи.

Крім цього, в роботі [11] описується алгоритм перебору, за допомогою якого було встановлено, що скінченні автомати з шістьма і меншою кількістю станів задачу «Штучного мурахи» не розв’язують.

Для стежки Санта-Фе побудовано скінченний автомат із семи станів, який розв’язує задачу за 189 кроків [6].

Отриманий в роботі скінченний автомат повністю розв’язує задачу, тобто штучний мураха за 200 кроків з’їдає всю їжу. Кількість станів скінченного автомата є 32. Можна додатково провести мінімізацію отриманого скінченного автомата за методом Ауфенкампа і Хона.

Задача мінімізації кількості внутрішніх станів скінченного автомата полягає в побудові такого автомата, який є еквівалентний заданому і має найменше число внутрішніх станів. Можливість скорочення числа станів визначається тим, що деякі внутрішні стани автомата, визначені як сумісні, можна об'єднати в один стан, не порушений при цьому закони функціонування автомата.

В даний час задача мінімізації скінченного автомата має ефективне і практично прийнятне розв’язання, коли внутрішні стани і виходи скінченних автоматів визначені для будь-якої вхідної послідовності. Відомі алгоритми мінімізації повних автоматів [12-14] зводяться до перебору системи класів сумісності. Основні результати цього підходу закладені в роботах [12, 13].

Висновки. У роботі запропоновано генетичний алгоритм, який здійснює побудову скінченних автоматів Мілі. Ці алгоритми застосовані для задачі про штучну мураху. Алгоритм апробовано на двох спеціально розроблених складних тестових даних, а саме стежці Джона Мюїра та стежці Санта-Фе. Написано власну програму, яка дозволяє проводити числові експерименти. Запропонована в роботі цільова функція дозволила зменшити кількість внутрішніх станів у скінченних автоматів Мілі, які одержують за допомогою генетичного алгоритму. Це також підтверджується експериментальними даними. Насамкінець зазначимо, що задача пошуку штучного мурахи з найменшою кількістю станів не ставилося.

Бібліографічні посилання

1. Jefferson D., Collins R., Cooper C., Dyer M., Flowers M., Korf R., Taylor C., Wang A. The Genesys System: Evolution as a Theme in Artificial Life. *Proceedings of Second Conference on Artificial Life.* MA: Addison-Wesley. 1992. P. 549 – 578.
2. Angeline P. J., Pollack J. Evolutionary Module Acquisition. *Proceedings of the Second Annual Conference on Evolutionary Programming.* 1993. P. 154-163.
3. Maniezzo V., Colorni A. The ant system applied to the quadratic assignment problem. *IEEE Transactions on Knowledge and Data Engineering.* 1999, 11(5). P. 769-778.
4. Koza J.R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA, USA: MIT Press, 1992.
5. Koza J.R. Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*. 1994, 4. P. 87–112.
6. Chambers L. Practical handbook of genetic algorithms. Boca Raton, CRC Press, 1999.
7. Bereza A., Lyashov M., Blanco L. Finite state machine synthesis for evolutionary hardware. *East-West Design & Test Symposium (EWDTS 2013).* IEEE, 2013. P. 1-4.
8. Fabera V., Janes V., Janesova M. Automata Construct with Genetic Algorithm. *9th EUROMICRO Conference on Digital System Design (DSD'06)*. 2006. P. 460-463.
9. Давыдов А.А., Соколов Д.О., Царев Ф.Н. Применение генетических алгоритмов для построения автоматов Мура и систем взаимодействующих автоматов Мили на примере задачи об «Умном муравье». *Научно-технический вестник информационных технологий, механики и оптики.* 2008, 53. С. 108-114.
10. Kim D. Memory analysis and significance test for agent behaviours. *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*. 2006. С. 151 – 158.
11. Царев Ф.Н., Шалыто А.А. Применение генетических алгоритмов для построения автоматов с минимальным числом состояний для задачи об «Умном муравье». *Тезисы научно-технической конференции «Научно-программное обеспечение в образовании и научных исследованиях»*. СПбГУ ПУ. 2008, С. 209 – 215.
12. Aufenkamp D.D., Hohn F.E. Analysis of Sequential Machines. *IRE Trans. Electr. Cornput.* 6. 1957, 6. P. 276-283.
13. Aufenkamp D.D. Analysis of Sequential Machines. *IRE Trans. Electr. Comput.* 7. 1958. 2. P. 299-306.
14. Хопкрофт Д., Мотвани Р., Ульман Д. Введение в теорию автоматов, языков и вычислений. М.: Издательский дом «Вильямс», 2002.

Надійшла до редколегії 16.09.2022.