ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ. КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

УДК 519.681.2

М.Ф. Єфремов, к.т.н., доц.
В.М. Єфремов, здобувач
Ю.М. Єфремов, к.т.н., доц.
Житомирський державний технологічний університет

СТРУКТУРА ПРОЕКТУВАННЯ ІНТЕЛЕКТУАЛЬНОГО АГЕНТА ТА МОЖЛИВОСТІ ЙОГО ВИКОНАННЯ

Більшість сучасних інформаційних систем не призначені для самостійного прийняття рішення, поведінки таких систем повинні бути закладені на етапі розробки. Попадання системи в умови, не враховані її розробниками, приводити до аварійного завершення. Експоненціальне зростання обчислювальних можливостей сучасних процесорів веде до зростання спектра завдань, піддаються автоматизації, і збільшення їх складності. Частково проблему коректності допомагає тестування, але воно містить недолік - поведінка системи зазвичай перевіряється в умовах, які враховувалися вже на етапі розробки. Одним з підходів, спрямованих на вирішення цієї проблеми, є агентно-орієнтоване програмування та застосування мультиагентних систем.

Метою роботи є висвітлення сучасних підходів: до визначення концепції агента, області застосування мультиагентних систем, математична модель агента, сучасні методи проектування і реалізації інтелектуальних агентів, крім того висвітлені підходи до визначення концепції інтелектуального агента, область застосування інтелектуальних агентів і мультиагентних систем, а також методи теоретичного опису агента і його реалізації. В основі наведених методів абстрактного опису інтелектуального агента лежать роботи М. Вулдрідж і Н. Дженінгса. Запропоновано декілька методів, що дозволяють описувати такі особливості поведінки інтелектуального агента, як самообучаемость, цілевизначення і прогнозування, а також планування. Освітлено концепцію «намірів» і широко поширену ментальну (beliefs-desires-intentions BDI) архітектуру інтелектуального агента, а також запропоновані методи їх опису в рамках розробленого формалізму.

Ключові слова: *теоретичне проектування; концепції агента; інтелектуальний агент; реактивність; соціальність; на що реагує агент; самонавчальний агент; архітектура агента; комунікація агентів.*

Постановка проблеми у загальному вигляді та зв'язок з важливими практичними завданнями. Однією з основних особливостей більшості сучасних інформаційних систем (ІС) є те, що вони не призначені для самостійного прийняття рішення в тих чи інших ситуаціях. Імовірно, щоб всі можливі нюанси поведінки ІС повинні бути створи людиною і закладені на етапі розробки. Попадання ІС в умови, не враховані розробниками, приводити до аварійного завершення або більш тяжких наслідків, аж до пошкодження обладнання та навіть до катастроф. З огляду на, що обчислювальних можливостей сучасних процесорів мають, тенденцію експоненціального зростання це веде до збільшення можливостей повної або часткової автоматизації ІС, зростає їх складність. При цьому, стає все складніше враховувати всі особливості поведінки ІС на етапі їх розробки, при цьому зростає ймовірність помилок, через складність їх обліку.

Частково проблему коректності ІС допомагає вирішити тестування, але воно містить недолік - поведінка ІС перевіряється тільки в тих умовах, які заплановані людиною на етапі розробки. При складних ІС повне тестування, перевірки всіх можливих комбінацій умов, неможливо за розумний час. Вирішення цієї проблеми можливе з використанням агентно-орієнтованого програмування і застосування мультиагентних систем, де використовується програмний агент. Важливим відмітним властивістю концепції агента є наявність зовнішнього середовища, з якою агент здатний взаємодіяти, але не володіє можливістю її контролювати, тому агент завжди повинен бути готовий до того, що вжиті нею дії не призведуть до бажаних результатів. Таким чином, агент є системою, здатною адекватно реагувати на зміни зовнішнього середовища, не передбачені явно його поведінковими механізмами. Саме це властивість і робить концепцію агента привабливим інструментом для вирішення багатьох завдань, що виникають сьогодні в області інформаційних технологій, таких, як, наприклад, розподілене управління і штучний інтелект.

В основі наведених методів абстрактного опису інтелектуального агента лежать роботи М.Вулдрідж [1, 2] і Н.Дженінгса [3], який дозволив запропонувати кілька методів, що дозволяють описувати такі особливості поведінки інтелектуального агента, як самообучаемость, ціле покладання і прогнозування, а також планування .

Постановка завдання. Освітлення концепції «намірів» і широко поширену ментальну (beliefs – desires – intentions BDI) архітектуру інтелектуального агента, а також запропонуємо методи їх опису в рамках розробленого формалізму.

Викладення основного матеріалу. Так згідно визначення агента, сформульованих багатьма сучасними дослідниками: Агент це, обчислювальна система, поміщена в зовнішнє середовище, здатна взаємодіяти з нею, здійснюючи автономні раціональні дії для досягнення певних цілей.

Однак наведене вище визначення не виділяє явно властивості інтелектуального агента, які передбачають гнучкість його поведінки. Зазвичай вважається, що інтелектуальний агент повинен мати таки властивості, як [2]:

• *реактивний* - здатністю відчувати зовнішнє середовище і реагувати на зміни в ній, здійснюючи дії, спрямовані на досягнення цілей.

• п*роактивних* - здатністю показувати кероване цілями поведінку, проявляючи ініціативу, здійснюючи дії, спрямовані на досягнення цілей.

• *соціальний* - здатність взаємодіяти з іншими сутностями зовнішнього середовища (іншими агентами, людьми і т. д.) Для досягнення цілей.

Будь-якого з перших двох властивостей досягти досить легко. Наприклад, властивістю проактивности, в широкому сенсі, володіє будь-який компілятор, основна «мета» якого - сформувати низькорівневий об'єктний код на основі програмного коду на мові програмування високого рівня. По суті будь-яка програмна функція може бути розглянута як система, метою якої є перетворення деяких вхідних даних у вихідні. Однак очевидно, що зміна у вхідних даних функції під час її роботи практично напевно призведе до краху або, як мінімум, невідповідності вихідних даних вхідним. Таким чином, довільна програмна система, взагалі кажучи, не володіє властивістю реактивності.

З іншого боку, тільки реактивні системи (тільки реагують на зміни у зовнішньому середовищі) теж досить прості - навіть найпримітивніша продуктивна система демонструє реактивність. Поєднання ж в системі обох властивостей в потрібних пропорціях є непростим завданням. Якщо агент жорстко слід сценарієм досягнення мети, не реагуючи на зміни у зовнішньому середовищі і не володіючи здатністю помічати необхідність коригування плану, навряд чи він зможе досягти поставленої мети. З іншого боку, поведінка, обмежене лише реакцією на зовнішні стимули, без будь-якого планування, не дозволить агенту досягти цілей, які потребують послідовних і взаємопов'язаних дій.

Що ж стосується третього властивості, соціальності, то воно теж не так просто, як може здатися на перший погляд. З одного боку, кожен день мільйони комп'ютерів взаємодіють між собою, обмінюючись пакетами бінарних даних. Але така поведінка ще не можна вважати соціальним. Крім комунікації соціальну поведінку на увазі кооперацію, яка полягає в поділі цілей між окремими сутностями, спільне планування та координацію дій, спрямованих на досягнення загальних цілей. Соціальна поведінка, як мінімум, передбачає наявність у агента уявлень про цілі інших сутностей і тому, як вони планують цих цілей досягти.

Детально область застосування агентів і мультиагентних систем описана в [5] і [7]. Можна виділити три основні класи систем, при реалізації яких зручно використовувати агентно-орієнтований підхід:

• *Відкриті системи* – системи, структура яких може змінюватися в процесі їх функціонування. Найбільший і відкрита на сьогоднішній день система - Інтернет. Соціальність і автономність агента дозволяють ефективно застосовувати його в якості елемента відкритої системи.

• *Складні системи* – найкращими сучасними методами боротьби зі складністю є модульність і трансцендентність. Завдяки автономності агент являє собою приклад трансцендентного модуля.

• *Інтерактивні системи* – більшість існуючих сучасних систем, незважаючи на графічні інтерфейс і потужну систему довідки, вимагають серйозних зусиль з боку потенційного користувача для їх освоєння. За допомогою агентів можна побудувати інтерактивну систему, яка буде не просто приймати і виконувати команди користувача, а активно і інтелектуально взаємодіяти з ним, прагнучи до досягнення загальних цілей.

У промисловості мультиагентні системи найбільш поширені в наступних областях:

• *Автоматизація управління складними системами* – область, в якій давно і ефективно застосовуються інтелектуальні агенти. Як приклади можна привести платформу ARCHON [8], систему управління виробництвом YAMS [9] і систему управління повітряним рухом OASIS [10].

• *Збір та обробка інформації* – агенти часто використовуються для реалізації систем, що збирають і обробляють інформацію з всесвітньої мережі Інтернет. Більшість сучасних пошукових машин реалізовано з використанням агентів.

• *Ігри* – сьогодні в комп'ютерних іграх противниками гравця людини часто стають гравці, реалізовані як інтелектуальні агенти.

Теоретичне проектування агента – це зручний інструмент, що дозволяє проектувати поведінку агентів з використанням чітких формальних методів, а потім перевіряти коректність отриманих систем з

використанням методів автоматичної перевірки коректності. В основу наведеної архітектури покладена модель, описана в [2]

Для початку припустимо, що зовнішнє середовище агента може бути описана за допомогою безлічі S станів середовища (environment states). Можливі дії агента описуються за допомогою безлічі A дій (actions). Теоретично агент може представлятися як функція:

$$action : S^* \rightarrow A \quad (1)$$

Таким чином, вибір конкретного дії з безлічі можливих агент здійснює, ґрунтуючись на поточному стані зовнішнього середовища, а також історії, яка описує всі попередні стані. При цьому, як уже згадувалося вище, дії агента можуть впливати на навколишнє середовище, але не контролювати її повністю. Недетермінірована поведінку зовнішнього середовища в цьому випадку можна описати наступною функцією:

$$env : S \times A \rightarrow 2^S \quad (2)$$

В цьому випадку недетермінізм зовнішнього середовища виражається в тому, що в залежності від свого поточного стану і вибраного агентом дії середовище може перейти в один стан з певної множини. Якщо для будь-якої пари стан-дія безліч можливих станів середовища складається з одного елемента, то таке середовище можна вважати детермінованою.

Взаємодія агента і зовнішнього середовища можна представляти за допомогою історії (history), яка є упорядкованою послідовністю пар стан-дія:

$$h : s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \dots, \quad (3)$$

де s_0 представляє початковий стан зовнішнього середовища, a_i представляє i-е дію, вбрання агентом, а s_i – стан зовнішнього середовища після здійснення агентом (i − 1)-і дії. У цьому випадку в зовнішньому середовищі *env* для агента action історія *h* буде допустимої (possible history), якщо виконані наступні дві умови:

• $\forall n \in N$, $a_n = action((s_0, s_1, s_2, ..., s_n))$ — тобто поведінка агента визначається моделює його функцією;

• $\forall n \in N$, $s_n+1 \in env(s_n, a_n)$ – тобто поведінку середовища визначається моделює функцією і поведінкою агента.

Характеризує поведінкою (characteristic behavior) агента будемо називати безліч всіх допустимих історій агента в зовнішньому середовищі. При цьому якщо якась властивість φ зберігається для всіх історій агента, то його назвемо інваріантним (invariant property) властивістю агента. Такі властивості відіграють особливу роль в проектуванні агентів, так як відображають спочатку закладені в агента мети, тобто саме ті цілі, для досягнення яких і проектується агент. Так, для агента, керуючого атомним реактором, прикладом подібного інваріанта буде властивість реактора не вибухати.

Безліч всіх допустимих історій агента agent в зовнішньому середовищі environment позначаємо як *hist* (*agent, environment*). Два агента ag_1 и ag_2 будуть поведенчески еквівалентні (*behaviorally equivalent*), якщо *hist* (ag_1, *environment*) = *hist* (ag_2, *environment*). В загальному разі взаємодія агента з зовнішнім середовищем ніколи не закінчується, тобто послідовність історії є нескінченною.

Можна виділити певний клас агентів, конкретна дія яких визначається не всією історією змін зовнішнього середовища, а лише її поточним станом. У цьому випадку агент буде представлятися функцією *action* : $S \rightarrow A$, тобто для кожного стану середовища у агента є чітко певним чином впливати. Цей клас агентів називається тільки реагують. Очевидно, що тільки реагує агент є окремим випадком агента. Тільки реагують агенти зазвичай реалізуються за допомогою продуктивних систем, їх поведінку легко передбачувана, але, як правило, не дуже інтелектуально.

Часто для опису агента зручно використовувати модель сприйняття навколишнього середовища. Для цього необхідно ввести безліч P можливих сприйняття і функцію see: S → P, що описує, яким чином певні стану середовища сприймаються агентом. У цьому випадку агент може бути описаний за допомогою функції функцією *see:* $S \rightarrow P$, описує, яким чином певні стану середовища сприймаються агентом. У цьому випадку агент може бути описаний за допомогою функції

$$action: P^* \rightarrow A, \quad (4)$$

тобто дію такого агента визначаються в загальному випадку поточним сприйняттям стану зовнішнього середовища, а також безліччю попередніх сприйнять.

Очевидно, що модель агента зі сприйняттям еквівалентна базової, і на перший погляд може здатися, що подібна модифікація не дає ніяких переваг і тільки ускладнює уявлення. Але насправді модель сприйняття має наступну важливу властивість:

$$(see(s_1) = see(s_2) \Rightarrow s_1 = s_2),$$

тобто різні стану середовища можуть однаково сприйматися агентом. Таким чином, модель сприйняття дозволяє явно відобразити такі два аспекти діяльності агента:

• *Неповнота інформації* - тільки в рідкісних випадках агент може володіти повною інформацією про зовнішнє середовище, достатньою для того, щоб однозначно ідентифікувати стан. Найчастіше агенту доступна лише часткова інформація, що дозволяє припустити, що зовнішнє середовище знаходиться в одному з станів класу з деяким набором інваріантних властивостей. Модель сприйняття дозволяє явно описати ці класи.

• *Надмірність інформації* - з іншого боку, найчастіше для прийняття рішення повна інформація про стан зовнішнього середовища і не потрібно, а цікава лише інформація про тих частинах системи, які можуть вплинути на виконання дії і його наслідки. В цьому випадку ми знову приходимо до класів станів, що володіють інваріантними властивостями.

Нескладно помітити, що функція сприйняття задає відношення еквівалентності «≈» на безлічі S таке, що

$s \approx s' \Leftrightarrow see(s) = see(s')$.

Це відношення розбиває безліч S на класи еквівалентності. Потужність відповідного безлічі класів еквівалентності (позначимо її $| \approx |$) дозволяє судити про сенсорних можливості агента - чим більше ця потужність, тим чіткіше агент здатний сприймати зовнішнє середовище. При $| \approx | = | S |$ агент має набутої сенсорними здібностями і може відрізнити будь-два різних стану зовнішнього середовища. В іншому граничному випадку, коли $| \approx | = 1$, сенсорні здібності у агента відсутні - він не здатний відрізнити жодне стан навколишнього середовища від іншого.

Оскільки виразна потужність моделі агента зі сприйняттям еквівалентна базової моделі агента, але має ряд корисних властивостей, далі в міркуваннях ми будемо розвивати саме її. Однак, при бажанні, адаптувати подальші міркування до базової архітектурі агента не складе труднощів.

Ще однією цікавою модифікацією теоретичного проектування агентів є агент зі станом. Вважається, що такий агент містить деякі внутрішні структури даних (що цілком природно для будь-якої інформаційної системи), які він модифікує в залежності від сприйняття поточного стану зовнішнього середовища, і на основі отриманих результатів вибирає дію. Для формалізації цього процесу введемо безліч I внутрішніх станів агента і функцію

$$refine : I \times P \rightarrow I, \quad (5)$$

що відповідає за оновлення внутрішнього стану відповідно до поточного сприйняттям середовища. Агент же в цьому випадку буде описуватися за допомогою функції

$$action : I \rightarrow A, \quad (6)$$

тобто дію буде вибиратися на основі поточного внутрішнього стану. При цьому вибір дії насправді здійснюється за допомогою суперпозиції функцій action (refine (i, see (s))). Для коректного опису поведінки агента зі станом необхідно визначити початковий стан i0. Тоді зміна внутрішнього стану агента і вибір дії буде відбуватися таким чином:

$\forall n \in N,\ i_n+1 = refine(i_n, see(s_n)) \wedge a_n = action(refine(i_n, see(s_n)))$.

Нескладно довести, що модель агента зі станом еквівалентна базової моделі. З одного боку, стан агента визначається виключно його історією, тобто базова модель по виразною потужністю не менше моделі агента зі станом. З іншого боку, в стані агента може зберігатися вся його історія, тобто модель зі станом за виразною потужністю не менше базової моделі.

Цінна перевага моделі агента зі станом полягає в зручності її практичної реалізації. Стан дозволяє засновувати вибір дії не на послідовності вхідних даних змінної (і потенційно безкінечною) довжини, а на одному конкретному елементі стану. Виявляється, що найчастіше вся історія агента і не потрібна для прийняття конкретного рішення, а досить лише якийсь отриманої з неї конкретної інформації.

Описана вище теоретична архітектура має один суттєвий недолік - певний таким чином агент не отримує інформації про вчинені ним самим діях, що різко обмежує його можливості в накопиченні досвіду і аналізі потенційних наслідків дій.

Можна вважати, що інформація про дії агента є частиною інформації про навколишнє середовище, але такий підхід не є наочним і інтуїтивно зрозумілим. Більш правильним вирішенням виниклої проблеми представляється включення інформації про здійснюваних діях явно під вхідні дані функції вибору дії:

$$action : (P \times A)^* \rightarrow A \quad (7)$$

У такому вигляді агент явно отримує інформацію про всю історію взаємодії з навколишнім середовищем, а не тільки про послідовність станів, в які навколишнє середовище переходила.

Для агента зі станом аналіз і накопичення досвіду здійснюються функцією оновлення стану (5), отже, інформацію про дії агента логічніше обробляти саме з її допомогою:

$$refine : I \times P \times A \rightarrow I \quad (8)$$

Зауважимо, що параметром функції перетворення стану передається не послідовність всіх дій агента, а тільки інформація про останньому зробленому їм дії. При цьому передбачається, що інформацію про попередні дії агент вже проаналізував і відбив результати аналізу в своєму внутрішньому стані, зробивши їх доступними для всіх наступних ітерацій [14] (рис.1).

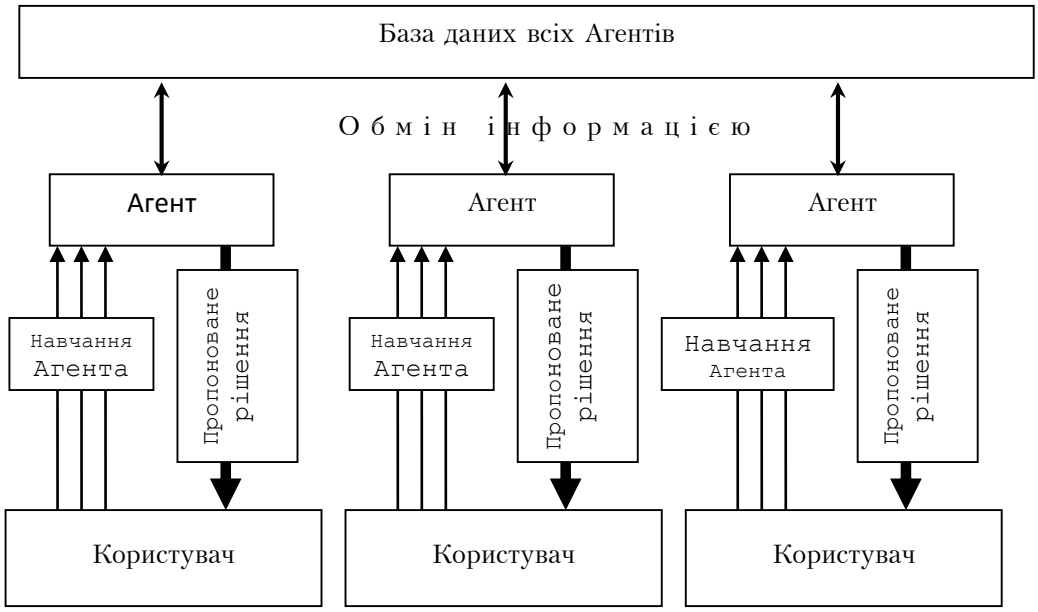

Рис. 1. Комунікація агентів між собою і користувачем

Звернемо увагу на таке важливе відмітна властивість інтелектуального агента, як кероване цілями поведінку. Одним з можливих способів опису мети агента є визначення оціночної функції:

$$goal : P \rightarrow [0, 1]. \quad (9)$$

Ця функція дозволяє агенту для кожного сприйняття стану середовища визначити, наскільки воно відповідає поставленим перед ним цілі. Часто можна зустріти ситуацію, коли мета агента є «неподільної», тобто агент або досягає мети повністю, або не досягає її взагалі. У цьому випадку значення оціночної функції логічніше розглядати не на відрізку [0, 1], а на безлічі з двох елементів {0, 1}, тобто Goal: P → {0, 1}. Таким чином, мета агента вважається досягнутою тільки в тому випадку, якщо зовнішнє середовище перейшла в один зі станів s, так їх, що see (s) ∈ goal-1 (1), і не досягнутої інакше. Існує інша поширена ситуація, коли агенту слід не прагнути до певних станів середовища, а, навпаки, уникати їх. Таку поведінку можна змоделювати за допомогою симетричною функції 1 - goal.

Зазвичай агент переслідує не одну конкретну мету, а деякий їх набір, при цьому цілі агента є частиною його внутрішнього стану: I = I'× 2Goals, де Goals = {g | g: P → [0, 1]} є кінцеве безліч всіх оціночних функцій, а I є інша частина стану агента, що не відноситься до ціле покладання. Тоді загальна оціночна функція агента для стану (i, G) може бути визначена, наприклад, так:

$$goal(s) = 1/|G| * \sum_{g \in G} g(p), \quad (10)$$

де p є оцінювана сприйняття стану середовища. У разі, коли агент не має жодної мети (G = ∅), значення оціночної функції визначається рівним нулю (goal = 0).

На практиці найчастіше цілі агента мають свої пріоритети. У цьому випадку структура безлічі цілей ускладнюється:

$$\boldsymbol{Goals = \{(g, w) \mid g : P \rightarrow [0, 1], w \in [0, +\infty)\},}$$

де w є невід'ємне число, яке визначає пріоритет конкретної мети для агента. Загальна оціночна функція агента для стану (i, G) тоді може мати вигляд:

$$\boldsymbol{goal(s) = \frac{1}{\sum_{(g,w) \in G} w} \sum_{(g,w) \in G} (w \cdot g(p))} \quad (11)$$

У разі, коли агент не має жодної мети з ненульовим пріоритетом значення оціночної функції визначається рівним нулю (goal = 0).

$$\boldsymbol{(G = \emptyset \text{ или } \sum_{(g,w) \in G} w = 0)}$$

Однак для ефективного керованого цілями поведінки агенту недостатньо просто визначити свої цілі, в більшості випадків необхідною умовою для ефективності є здатність передбачати наслідки своїх дій для зовнішнього середовища. Як частину своїх статків агент повинен включати що прогнозує функцію

$$prog : P \times A \rightarrow 2^P \quad (12)$$

приймаючу в якості параметрів поточне сприйняття стану зовнішнього середовища і дію агента, а повертає безліч можливих сприйняття станів середовища, в одне з яких вона перейде після виконання дії.

Більш гнучкий варіант прогнозуючої функції включає ймовірність переходу зовнішнього середовища в той чи інший стан:

$$prog : P \times A \rightarrow 2^{P \times [0,1]}, \quad (13)$$

тобто в парі (p2, θ) ∈ prog (p1, a) число θ є ймовірність того, що при виконанні агентом дії a зовнішнє середовище зі стану, яка сприймається як p1, перейде в стан, сприймається як p2. Очевидно, що просту що прогнозує функцію можна виразити за допомогою імовірнісної, використовуючи граничні значення ймовірності 0 і 1.

Прогнозуюча функція є важливою складовою внутрішнього стану агента і будується на основі його акумульованого досвіду. Позначимо множину всіх прогнозують функцій через

$Progs = \{prog \mid prog : P \times A \rightarrow 2^{P \times [0,1]}\}$

В этом случае структура внутреннего состояния агента примет вид

$I = I' \times 2^{Goals} \times Progs.$

Использование прогнозирующих функций позволяет представить задачу принятия решения агентом как задачу нахождения точки максимального значения функции потенциального эффекта действия $\upsilon : A \rightarrow [0, +\infty)$, выражаемой через функции *goal* и *prog*. Для простой прогнозирующей функции это $\upsilon(a) =$

$$\sum_{p \in prog(p_{cur}, a)} goal(p),$$

де p_{cur} є сприйняття поточного стану зовнішнього середовища. Для ймовірнісної прогнозуючої функції цей вислів складніше:

$$v(a) = \sum_{(p,\theta) \in prog(p_{cur}, a)} (\theta \cdot goal(s))$$

Якщо ж звернутися до обчислення функції goal (див. Рівняння 11), то можна отримати наступне підсумкове вираз функції υ для агента зі станом (i, G, prog) і зовнішнього середовища в стані, що сприймається як p_{cur}:

$$v(a) = \sum_{(p,\theta) \in prog(p_{cur}, a)} \left(\frac{\theta}{\sum_{(g,w) \in G} w} \cdot \sum_{(g,w) \in G} (w \cdot g(p)) \right) \quad (14)$$

Зауважимо, що для підвищення ефективності обчислень нормувальний коефіцієнт $\left(\sum_{(g,w) \in G} w \right)^{-1}$ можна винести за знак суми або взагалі відкинути, так як він не впливає на розташування точки максимуму функції. У цьому випадку функція v набуде вигляду:

$$v(a) = \sum_{(p,\theta) \in prog(p_{cur}, a)} \left(\theta \cdot \sum_{(g,w) \in G} (w \cdot g(s)) \right) \quad (15)$$

Слід зауважити, що в разі обмеженого ресурсами агента точно вирішити задачу пошуку точки максимуму функції для досить складних проблем може виявитися неможливим за розумний час. У таких випадках агент може використовувати наближені методи рішення, а функція υ дозволить оцінити ефективність поведінки агента [15] (рис. 2).

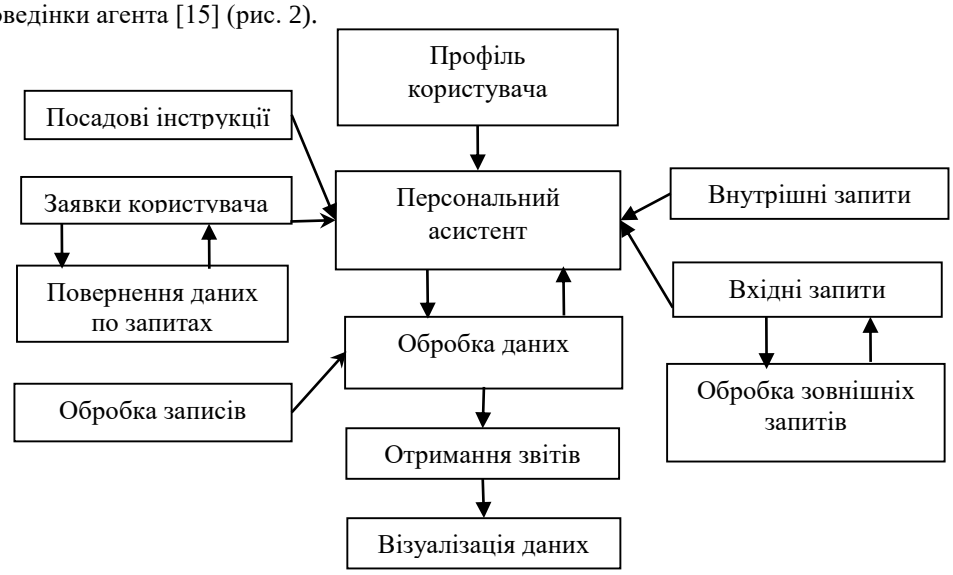

Рис. 2. Дія агента при вирішенні складаний проблем з використання наближення методів рішень

*Як зазначалося вище, інтелектуальний агент імовірно повинен демонструвати проактивное пов*едінку, що фактично означає будувати деякі плани і розраховувати свої дії на кілька ходів вперед. Таким чином, крім оперативних цілей, що досягаються на поточному дії, у агента з'являються перспективні цілі, для досягнення яких йому буде потрібно виконати послідовність з декількох дій. Насправді саме перспективні цілі є визначальними, і на їх основі агент формує цілі оперативні за допомогою планування, що використовує наявний у нього досвід.

У більшості випадків процес планування, крім формування оперативних цілей, включає в себе підтримку допоміжної структури даних, що є частиною загального стану агента. Позначимо множину всіх таких структур як Plans, тоді стан агента буде включати безліч перспективних цілей, що прогнозує функцію, інформацію про план і решту допоміжну інформацію: I = I '× 2Goals × Progs × Plans.

В результаті процес планування можна змоделювати за допомогою планувальної функції

plan: 2Goals × P rogs → Plans,

яка на основі даних перспективних цілей формує структуру з описом плану, використовуючи для цього прогнозуючу функцію, і функцію формування оперативних цілей

oper: Plans × P → 2Goals,

яка на основі плану і сприйняття поточного стану зовнішнього середовища здійснює формування безлічі оперативних цілей.

Можна відзначити, що в більшості випадків процес планування має велику обчислювальну складність, ніж процес прийняття рішення про конкретний оперативному дії. Для оптимізації цього процесу можна запам'ятовувати основу одного разу складеного плану і адаптувати її до виникаючих завдань, що може виявитися значно ефективніше створення нового плану «з нуля».

Коріння ментальної (beliefs-desires-intentions) архітектури інтелектуальних агентів лежать в філософських підходах до аналізу розумової діяльності людини, того, як люди на практиці приймають рішення про те, що їм слід робити. Можна виділити наступні окремі етапи в ухваленні рішення ментальним агентом: спочатку агент повинен зрозуміти, чого він хоче, потім визначити, які цілі з бажаних, він буде намагатися реалізувати, а потім зрозуміти, як він буде реалізовувати обрані цілі.

При цьому в стані агента чітко розмежовуються наступні компоненти:

• Уявлення (beliefs) - деяка інформація про закономірності і поточний стан зовнішнього середовища, яка є у розпорядженні агент. При цьому передбачається, що ця інформація може бути помилковою і неповною, тому її можна розглядати тільки як уявлення, але не як достовірні знання. Зауважимо, що прогнозуюче функція є частиною уявлень агента. Таким чином, безліч уявлень має структуру Beliefs = I '× Progs.

• Бажання (desires) - безліч всіх цілей, яких агент хотів би домогтися. При цьому безліч може бути великим і суперечливим. Малоймовірно, що агент, обмежений ресурсами, зможе реалізувати всі свої бажання. За своєю структурою безліч бажань збігається з безліччю цілей, зваженому чи ні (Desires = 2Goals).

• Наміри (intentions) - безліч тих цілей, яких агент вирішив домогтися. Сформований безліч цілей повинно бути здійснимо за поданнями агента, тобто все наміри агента повинні бути досяжні в сукупності. Для уявлення намірів також можна використовувати зважене безліч цілей, при цьому ваги намірів будуть залежати від ваг відповідних бажань, але, можливо, не збігатися з ними.

Під здійсненне намірів у разі який планує агента можна розуміти наявність у агента плану, що веде до здійснення всіх намірів. Більш того, цей план стає частиною структури намірів агента Intentions = 2^{Goals} × Plans, а також впливає на привласнюються намірам ваги (наприклад, намір, на реалізацію якого було витрачено багато зусиль і яке близьке до завершення, буде мати більший пріоритет). У разі, якщо для подання бажань агент користується зваженими цілями, можна запропонувати механізм оцінки сформованого безлічі намірів, заснований на критерії загальної корисності

$$utility(G) = \sum_{(g,w)\in Des \wedge g\in G} w,$$

де Des є безліч бажань, а G є оцінювана безліч намірів. Таким чином, при формуванні безлічі намірів агент вирішує завдання оптимального вибору, намагаючись максимізувати загальну корисність, зберігши здійсненність безлічі намірів.

Новим ключовим поняттям процесу прийняття рішення в ментальної архітектурі є саме намір. В цілому можна виділити наступні властивості намірів:

• Наміри задають напрямок діяльності - агент намагається знайти дії, здатні здійснити наміри і виконати їх.

• Наміри обмежують майбутній вибір - агент не може формувати нові наміри, несумісні з уже прийнятими, тобто що ведуть до нездійсненності безлічі намірів.

• Наміри мають довгий час життя - якщо агент сформував план реалізації наміру, але він провалився, то агент буде формувати нові плани і намагатися реалізувати намір іншим способом. Намір може бути відкинуто лише при здійсненні певного ментального зусилля у випадках, якщо агент прийшов до

висновку, що реалізувати намір неможливо (не вдається сформувати план, що веде до досягнення наміри) або воно вже не актуальне для агента.

• Наміри впливають на міркування про майбутнє і, відповідно, плани - якщо агент виробив намір, то він може будувати плани на майбутнє з припущенням, що цей намір реалізовано.

Сам же процес прийняття рішень ментального агента складається з послідовних етапів, на кожному з яких використовуються результати попередніх. Поетапне перетворення виконується за допомогою наступних функцій:

1. *Актуалізація уявлень (brf: Beliefs × P × A → Beliefs) - на цьому етапі модифікується уявлення агента про поточний стан середовища на основі сприйняття, а також проводиться аналіз і узагальнення отриманого досвіду.*

2. Формування бажань (option: Beliefs × Desires → Desires) - на цьому етапі агент формує безліч всіх своїх бажань. При цьому використовується безліч вже існуючих бажань і актуалізовані уявлення агента.

3. Фільтрація бажань (filter: Beliefs × Desires × Intentions → Intentions) - на цьому етапі визначається, які саме цілі з безлічі бажаних агент буде намагатися реалізувати. Здійснюється вибір з безлічі всіх бажань, з урахуванням уявлень і вже обраних намірів, створюється план реалізації намірів і формуються ваги намірів з урахуванням плану. Важливою умовою роботи фільтра є те, що сформований безліч намірів може містити тільки ті елементи, які спочатку були присутні в одному з вхідних множин:

$filter(Bel, Des, Int) \subseteq Des \cup Int$.

4. Формування оперативних цілей (*oper : Beliefs × Intentions → 2^{Goals}*) — на цьому етапі агент, використовуючи безліч намірів в якості перспективних цілей, формує безліч оперативних цілей і ваги для них. У цьому процесі агент також використовує уявлення і сформований для реалізації намірів план.

5. Вибір дії (*select : Beliefs × 2^{Goals} → A*) — на цьому етапі агент вибирає оптимальну дію, на основі своїх уявлень, а також оперативних цілей і їх ваг.

В раніше висвітленим матеріалі виділили три основні аспекти діяльності агента:

а) формування безлічі цілей (іноді багатоступінчате), б) прогнозування поведінки навколишнього середовища в) планування дій. Таким чином, висвітлимо відомі підходи, що дозволяють реалізувати ці аспекти діяльності агента на практиці.

Можна запропонувати такі підходи до подання мети агента і формуванню безлічі цілей:

• Логічний – в цьому випадку мета агента описується деяким виразом формальної логіки, інтерпретація якого залежить від стану зовнішнього середовища. При цьому клас станів середовища, в яких мета вважається досягнутою, складають ті стану, на яких цей вислів інтерпретується як справжнє. У разі використання нечіткої логіки можна отримати мети, здійсненні частково. Формальна логіка є потужним і привабливим інструментом з багатою теоретичною базою, але має і низку недоліків, пов'язаних в основному з обчислювальною складністю і можливої неповнотою формального апарату.

• Перерахування – в цьому випадку агент явно перераховує стану зовнішнього середовища і ступінь їх відповідності даної мети. Очевидним плюсом даного методу є простота і низька обчислювальна складність для невеликих завдань. Однак в разі складних завдань явне перерахування може зажадати занадто багато пам'яті і призведе до зниження ефективності.

• І-АБО граф – потужний інструмент декомпозиції задач на під задачі. Ідея методу полягає в формуванні орієнтованого графа, на одній "стороні" якого знаходяться високо рівневі, складні цілі агента, а на іншій – елементарні цілі, представлені відповідним чином. Внутрішню частину графа становлять вершини двох типів: "І" та "АБО". У разі вершини "І", для виконання відповідного вершині умови, слід дотримуватися умов для всіх вершин, з'єднаних з вихідними ребрами. Для вершин "АБО" досить виконання умови хоча б в одній з вершин, з'єднаних вихідної дугою. Можна помітити, що І-АБО граф досить легко моделюється засобами інших методів за допомогою відповідних логічних або теоретико-численних операцій, проте явне завдання графа дає цілий ряд вельми істотних для подальшого планування переваг. Наприклад, граф дозволяє явно ідентифікувати найбільш критичні ділянки, задіяні відразу в декількох цілях.

Можна запропонувати такі підходи до формування прогнозуючої функції агента:

• Логічний – в цьому випадку агент підтримує базу знань, що містить безліч тверджень формальної логіки, що описують причинно-наслідкові зв'язки зовнішнього середовища. Для визначення можливих переходів зовнішнього середовища при здійсненні агентом певної дії розглядається безліч логічних наслідків з безлічі тверджень бази знань, інтерпретованих на поточний стан зовнішнього середовища як справжні, в припущенні, що агент виконав дію. Безліч станів середовища, на яких ці логічні слідства інтерпретуються як справжні, і буде визначати можливі переходи середовища. Впровадження нечіткої логіки дозволить отримати вірогідну що прогнозує функцію. Плюсом логічного підходу є висока точність і формальність, до мінусів же можна віднести високу обчислювальну складність.

• Фактографічний – агент підтримує базу знань, що містить інформацію про всі мали місце переходах зовнішнього середовища. При цьому прогнозовані переходи визначаються на основі такої бази знань дуже просто – можливі лише ті переходи, які мали місце раніше, і ймовірність переходу тим вище, чим

частіше такий перехід зустрічався раніше. До плюсів методу можна віднести простоту реалізації і швидкодія, до мінусів же - відсутність узагальнення отриманого досвіду.

• Заснований на застосуванні нейронних мереж – широко поширений останнім часом метод наближеного моделювання. На відміну від повної фактографии в даному підході виявлений перехід не запам'ятовується явно, а використовується як елемент навчальної послідовності для моделює нейронної мережі. Плюсом методу є невисока обчислювальна складність, яка не залежить від обсягу отриманого агентом досвіду, до мінусів же можна віднести повну відсутність наочності у отриманої моделі.

Реалізація ефективного планує механізму є, безумовно, найбільш складною із завдань агента. Основний принцип, що лежить в основі будь-якого планує механізму, - це рекурсивне розбиття задачі на під задачі, проте в кожному конкретному методі декомпозиція може відрізнятись по глибині проведеної деталізації і по її ширині. Можна запропонувати наступні способи декомпозиції:

• Послідовність завдань – в цьому випадку план є плоскою ланцюгом завдань, які слід виконати одну за одною. При цьому кожен елемент ланцюга є простою метою, досяжною за допомогою єдиного дії. Особлива увага тут приділяється організації механізму контролю, що дозволяє визначити, чи завершився попередній крок плану успішно чи ні, тому для організації планування зручніше використовувати неподільні цілі, що дозволяють однозначно визначити, досягнуті вони чи ні. Плюсом такої декомпозиції є відносна простота, мінусом ж – відсутність коштів обробки незапланованих ситуацій.

• Дерево завдань – на відміну від простої послідовності в дереві враховуються різні результати дій агента на кожному кроці, внаслідок чого і виникає розгалуження дерева. Такий план складніше побудувати, проте при його реалізації менша ймовірність того, що зустрінеться незапланована ситуація і доведеться будувати новий план «з нуля».

• Мережа завдань – в цьому випадку допускається подальше злиття розійшлися гілок плану з базовою лінією. План в такому вигляді отримує більш економічне уявлення і може бути швидше побудований, хоча метод його уявлення складніше реалізувати.

Методи виконання декомпозиції слід засновувати на прогнозуючої функції, визначаючи для кожного кроку плану можливі його наслідки. Можна виділити два принципових підходи до побудови плану:

• прямий – в цьому випадку агент на основі поточних умов, використовуючи що прогнозує функцію, намагається визначити дії, здатні привести до досягнення мети;

• зворотний – агент на основі функції, зворотної прогнозуючої, і передбачуваної мети намагається знайти послідовність дій і переходів, що реалізує шлях з поточного вузла до даної мети.

Реалізація агента за допомогою методу повного перерахування та повної фактографии може виявитися значно простіше і ефективніше для не надто об'ємних завдань [16–18]. Розглянемо застосування цих методів для реалізації контролера пристрою-збирача.

Безліч можливих сприйняття агентом стану зовнішнього середовища можна визначити наступним чином:

$P = \{A, B, C\} \times \{-1,0,1\}$

Цілями агента є стану зовнішнього середовища, які сприймаються як

• $\{(x, 1) \mid x \in \{A, B, C\}\}$ - такі стани зовнішнього середовища відрізняє те, що значення оціночної функції збільшилася в порівнянні з попереднім кроком. Цій меті дамо вага $w = 2$;

• $\{(x, y) \mid x \in \{A, B, C\}, y \in \{0,1\}$ - такі стани зовнішнього середовища відрізняє те, що значення оціночної функції не зменшилася в порівнянні з попереднім кроком. Цій меті дамо вага $w = 1$.

База знань Ω агента буде складатися з трійок

$(O, a, \Delta\varphi) \in \{A, B, C\} \times A \times \{-1,0,1\}$, де о - об'єкт зовнішнього середовища, який перебував перед влаштуванням на попередньому кроці, а - розпочате агентом дію, а $\Delta\ \varphi$ - зміна оціночної функції в порівнянні з попереднім кроком.

Що прогнозує функцію можна визначити за допомогою наступного алгоритму.

• Якщо трійка (о, а, Δφ), де о є що знаходиться безпосередньо перед влаштуванням об'єкт зовнішнього середовища, входить в базу знань $((o, a, \Delta\varphi) \in \Omega)$, то ймовірність переходу зовнішнього середовища в стан, сприймається як $(x, \Delta\varphi)$ (де $x \in \{A, B, C\}$), при виконанні агентом дії а дорівнює 1.

• Якщо жодна трійка виду (о, а, z), де о є що знаходиться безпосередньо перед влаштуванням об'єкт зовнішнього середовища, а $z \in \{-1, 0, 1\}$, НЕ входить в базу знань $((o, a, z) \in / \Omega)$, то ймовірність переходу зовнішнього середовища в стан, сприймається як (x, z) (де $x \in \{A, B, C\}$), при виконанні агентом дії а дорівнює 1/2.

Неважко помітити, що поведінка такого агента буде еквівалентно поведінки агента, реалізованого з використанням формальної логіки. При цьому повністю заповнена база знань такого агента буде містити всі необхідні дані для прийняття рішення.

Висновки та перспективи подальших розвідок. Висвітлено сучасні підходи до визначення концепції агента, області застосування мультиагентних систем, математична модель агента, а також сучасні методи проектування і реалізації інтелектуальних агентів. Інтелектуальні агенти та мультиагентні системи на даний момент є однією з найбільш привабливих областей як теоретичної, так і

практичної інформатики. Висвітлені деякі методи побудови і застосування окремого інтелектуального агента, тоді як пов'язані з інтеграцією кількох агентів в єдину систему завдання залишилися за межами розгляду є предметом подальшого дослідження. Висвітлена концепція «намірів» і широко поширену ментальну (BDI) архітектуру інтелектуального агента, а також запропоновані методи їх опису в рамках розробленого формалізму.

Список використаної літератури:

1. *Wooldridge M.J.* The Logical Modeling of Computational Multi-Agent Systems. PhD thesis / *M.J. Wooldridge.* – Manchester, 1992. – 153 p.
2. *Wooldridge M.J.* Intelligent Agents / *M.J. Wooldridge* // Multiagent Systems. – 2001. – Pp. 27–79.
3. *Wooldridge M.J.* Intelligent Agents: Theory and Practice / *M.J. Wooldridge, N.R. Jennings* // The Knowledge Engineering Review. – 1995.
4. *Huhns M.N.* Multiagent Systems and Societies of Agents / *M.N. Huhns, L.M. Stephens* // Multiagent Systems. – 2001. – Pp. 79–121.
5. *Jennings N.R.* Applications of Intelligent Agents / *N.R. Jennings, M.J. Wooldridge.* – London : Queen Mary & Westfield College, University of London. – 2000. – 27 p.
6. *Miraftabi R.* Agents on the Loose: An Overview of Agent Technologies / *R.Miraftabi.* – Joensuu : Department of Computer Sciense, University of Joensuu. – 2000. – 17 p.
7. *Van Dyke Parunak H*. Industrial and Practical Application of DAI / *H.Parunak Van Dyke* // Multiagent Systems. – 2001. – Pp. 27–79.
8. *Jennings N.R.* Using archon to Develop Real-World DAI applications for electricity transportation management and Particle Acceleration Control / *N.R. Jennings, J.Corera, I.Laresgoiti* // IEEE Expert Special Issue on Real World Applications of DAI systems. – 1996.
9. *Van Dyke Parunak H*. Applications of Distributed Artificial Intelligence in Industry / *H.Parunak Van Dyke* // Foundations of Distributed Artificial Intelligence. – 1994.
10. *Ljunberg M.* The OASIS Air Traffic Management System / *M.Ljunberg, A.Lucas* // Proceedings of the Second Pacific Rim International Conference on AI (PRICAI-92). – 1992.
11. *Durfee E.H.* Distributed Problem Solving and Planning / *E.H. Durfee* // Multiagent Systems. – 2001. – Pp. 121–165.
12. *Sen S.* Learning in Multagent Systems / *S.Sen, G.Weiss* // Multiagent Systems. – 2001. – Pp. 259–299.
13. *Єфремов М.Ф.* Комплексна система інтелектуалізації управлінської праці в автогосподарствах на базі інтелектуальних агентів / *М.Ф. Єфремов, В.М. Єфремов, Ю.М. Єфремов* // Вісник ЖДТУ. Серія : Технічні науки. – 2008. – № 4 (47). – С. 139–146.
14. *Єфремов В.М.* Про системний підхід до створення та застосування програмних засобів на базі складального програмування для автоматизованих комп'ютерних систем / *В.М. Єфремов* // Вісник ЖІТІ. Серія : Технічні науки. – 1999. – № 11. – С. 199–206.
15. *Єфремов В.М.* Побудова систем управлення автогосподарством із застосуванням підходу багатоагентного програмування / *В.М. Єфремов* // Вісник ЖІТІ. Серія : Технічні науки. – 2000. – № 12. – С. 234–237.
16. *Єфремов В.М.* Предметна область та основні завдання автоматизації автогосподарства / *В.М. Єфремов* // Вісник ЖДТУ. Серія : Технічні науки. – 2008. – № 3 (46). – С. 113–120.
17. *Єфремов М.Ф.* Агентна система підтримки прийняття рішень для малих і середніх автотранспортних підприємств / *М.Ф. Єфремов, В.М. Єфремов, Ю.М. Єфремов* // Вісник ЖНАЕУ. – 2008. – № 2. – С. 352–360.
18. *Єфремов М.Ф.* Автоматизація управління автогосподарством на базі агентів з урахуванням міжнародних перевезень / *М.Ф. Єфремов, В.М. Єфремов, Ю.М. Єфремов* // Вісник ЖДТУ. Серія : Технічні науки. – 2010. – № 1 (47). – С. 80–85.

References:

1. Wooldridge, M.J. (1992), *The Logical Modeling of Computational Multi-Agent Systems*, PhD thesis, Manchester, 153 p.
2. Wooldridge, M.J. (2001), *Intelligent Agents Multiagent Systems*, pp. 27–79.
3. Wooldridge, M.J. and Jennings, N.R. (1995), «Intelligent Agents: Theory and Practice*», The Knowledge Engineering Review*, pp. 27–79.
4. Huhns, M.N. and Stephens, L.M. (2001), «Multiagent Systems and Societies of Agents*», Multiagent Systems*, pp. 79–121.
5. Jennings, N.R. and Wooldridge, M.J. (2000), *Applications of Intelligent Agents*, Queen Mary & Westfield College, University of London, London, 27 p.
6. Miraftabi, R. (2000), *Agents on the Loose: An Overview of Agent Technologies*, Department of Computer Sciense, University of Joensuu, Joensuu, 17 p.
7. Van Dyke Parunak, H. (2001), «Industrial and Practical Application of DAI*», Multiagent Systems*, pp. 27–79.
8. Jennings, N.R, Corera, J. and Laresgoiti, I. (1996), «Using archon to Develop Real-World DAI applications for electricity transportation management and Particle Acceleration Control», *IEEE Expert Special Issue on Real World Applications of DAI systems*, pp. 27–79.

9. Van Dyke Parunak, H. (1994), «Applications of Distributed Artificial Intelligence in Industry», *Foundations of Distributed Artificial Intelligence*, pp. 27–79.
10. Ljunberg, M. and Lucas, A. (1992), «The OASIS Air Traffic Management System», *Proceedings of the Second Pacific Rim International Conference on AI (PRICAI-92)*, pp. 30–34.
11. Durfee, E.H. (2001), «Distributed Problem Solving and Planning», *Multiagent Systems*, pp. 121–165.
12. Sen, S. and Weiss, G. (2001), «Learning in Multagent Systems», *Multiagent Systems*, pp. 259–299.
13. Єfremov, M.F., Єfremov, V.M. and Єfremov, Yu.M. (2008), «Kompleksna sistema intelektualizatsiї upravlins'koї pratsi v avtogospodarstvakh na bazi intelektual'nikh agentiv», *Visnik ZhDTU, Tekhnichni nauki*, No. 4 (47), pp. 139–146.
14. Єfremov, V.M. (1999), «Pro sistemniy pidkhid do stvorennya ta zastosuvannya programnikh zasobiv na bazi skladal'nogo programuvannya dlya avtomatizovanikh komp'yuternikh system», *Visnik ZhITI, Tekhnichni nauki*, No. 11, pp. 199–206.
15. Єfremov, V.M. (2000), «Pobudova sistem upravlennya avtogospodarstvom iz zastosuvannyam pidkhodu bogatoagentnogo programuvannya», *Visnik ZhITI, Tekhnichni nauki*, No. 12, pp. 234–237.
16. Єfremov, V.M. (2008), «Predmetna oblast' ta osnovni zavdannya avtomatizatsiї avtogospodarstva», *Visnik ZhDTU, Tekhnichni nauki*, No. 3 (46), pp. 113–120.
17. Єfremov, M.F., Єfremov, V.M. and Єfremov, Yu.M. (2008), «Agentna sistema pidtrimki priynyattya rishen' dlya malikh i serednikh avtotransportnikh pidpriєmtv», *Visnik ZhNAEU*, No. 2, pp. 352–360.
18. Єfremov, M.F., Єfremov, V.M. and Єfremov, Yu.M. (2010), «Avtomatizatsiya upravlinnya avtogospodarstvom na bazi agentiv z urakhuvannyam mizhnarodnikh perevezen'», *Visnik ZhDTU, Tekhnichni nauki*, No. 1 (47), pp. 80–85.

ЄФРЕМОВ Микола Федорович – кандидат технічних наук, доцент кафедри ПЗС Житомирського державного технологічного університету.

Наукові інтереси:
– програмування;
– штучний інтелект та розум.

Тел.: (068) 271–26–51.
E-mail: eyuri@list.ru.

ЄФРЕМОВ Віталій Миколайович – здобувач Житомирського державного технологічного університету.

Наукові інтереси:
– програмування;
– штучний інтелект та розум.

Тел.: (068) 271–26–52.
E-mail: eyuri@list.ru

ЄФРЕМОВ Юрій Миколайович – кандидат технічних наук, доцент кафедри ПЗС Житомирського державного технологічного університету.

Наукові інтереси:
– програмування;
– штучний інтелект та розум.

Тел.: (097) 304–68–96.
E-mail: eyuri@list.ru.

Стаття надійшла до редакції 17.02.2017.