НАЦІОНАЛЬНА АКАДЕМІЯ УПРАВЛІННЯ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК
КАФЕДРА ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ

МЕТОДИ ТА СИСТЕМИ ШТУЧНОГО ІНТЕЛЕКТУ

НАВЧАЛЬНО-МЕТОДИЧНИЙ КОМПЛЕКС

КИЇВ - 2013

Мета та завдання навчальної дисципліни

Мета дисципліни “Методи та системи штучного інтелекту” – це формування знань, вмінь та навичок, необхідних для розроблення і застосування моделей відображення знань, стратегій логічного виведення, технологій інженерії знань, технологій і інструментальних засобів інтелектуальних систем.

Завдання курсу: вивчення сучасних методів і технологій штучного інтелекту, розроблення та застосування моделей представлення знань для побудови інтелектуальних систем.

У результаті вивчення навчальної дисципліни студент повинен

знати: основні підходи, методи і технології штучного інтелекту та способи розроблення і застосування моделей відображення знань, стратегій логічного виведення, технологій інженерії знань, технологій і інструментальних засобів інтелектуальних систем

вміти:

- проектувати елементи лінгвістичного забезпечення інформаційних систем;
- розробляти семантичні портали знань;
- розробляти та застосовувати моделі представлення знань, стратегії логічного виведення, технологій інженерії знань, технологій і інструмен¬тальних засобів побудови інтелектуальних систем.

Програма навчальної дисципліни

Змістовий модуль 1. ***Основні поняття та означення***

Тема 1. Поняття штучного інтелекту

1. Поняття штучного інтелекту (ШІ).
2. Поняття інтелектуальної системи (ІС).
3. Поняття інте¬лектуальної задачі (ІЗ).

Змістовий модуль 2. ***Представлення знань у СШІ***

Тема 2. Представлення знань у СШІ

1. Знання та моделі представлення знань у СШІ.
2. Логічні моделі. Мережні моделі. Сценарії. Інтелектуальний інтерфейс.
3. Продукційні моделі представлення знань.
4. Управління пошуком рішень у продукційних системах.

Змістовий модуль 3. ***Способи подання інтелектуальної задачі (ІЗ) та методи пошуку рішень***

Тема 3. Способи подання інтелектуальної задачі (ІЗ)

1. Способи подання ІЗ, їхні переваги та недоліки.
2. Методи «сліпого» пошуку. Пошук у глибину та ширину.
3. Методи евристичного пошуку.

Тема 4. Пошук рішень ІЗ у просторі станів

1. Алгоритм А*. Генетичний алгоритм.
2. Методи пошуку рішень ІЗ у разі зведення задач до сукупності підзадач.

Змістовий модуль 4. ***Вирішувачі проблем, засновані на знаннях***

Тема 5. Експертні системи (ЕС)

1. Призначення та прин¬ципи побудови ЕС.
2. Узагальнена архітектура ЕС.
3. Класи задач, які вирішуються за допомогою ЕС.

Тема 6. Розробка ЕС

1. Етапи розробки, придбання знань, пошук та пояснення рішень.
2. Інженерія знань.

Тема 7. Семантичні сітки (СС) та фрейми

1. Основні поняття СС, типи СС, способи опису СС.
2. Лінгвістичні змінні, логічне виведення на СС та гібридні нейронні мережі.
3. Основні поняття та типи штучних нейронних мереж.
4. Основні поняття фреймів, структура фрейма. Фреймові системи.

Змістовий модуль 5. ***Сучасні тенденції та підходи до створення СШІ***

Тема 8. Сучасні засоби створення СШІ

1. Сучасні програмні засоби - мови функціонального та логіч¬ного програмування: Prolog, Allegro CLOS, CLIPS, JESS.
2. Онтологічний підхід до представлення знань та інтеграції знань у розподілених інформа¬цій¬них середовищах типу Internet.

Теми лабораторних занять

1. Інтелектуальні агенти
2. Інформативний та неінформативний пошуки
3. Локальний пошук
4. Складні інтелектуальні агенти. Печера Вія

Лабораторні підвищеної складності

Теоретичні основи та алгоритми для виконання лабораторних підвищеної складності наведені у [2].

1. Задача N ферзів (алгоритм відпалу) ([2], розділ 2)
2. Алгоритм адаптованої теорії резонансу ([2], розділ 3)
3. Алгоритм зворотнього розповсюдження ([2], розділ 4)
4. Генетичний алгоритм ([2], розділ 5)
5. Моделювання штучного життя ([2], розділ, 6)

Рекомендована література

Базова

1. Рассел, Норвіг «Штучний інтелект. Сучасний підхід»
2. Джонс М.Т. Программирование искусственного интеллекта в приложениях. - М.: ДМК Пресс, 2006. - 312 с.
3. Субботін С.О. Подання і обробка знань у системах штучного інтелекту та підтримки прийняття рішень: навч. посібник. - Запоріжжя, ЗНТУ, 2008. - 341 с.
4. Рідкокаша А.А., Голдер К.К. Основи систем штучного інтелекту. Навчальний посібник. - Черкаси: "ВІДЛУННЯ-ПЛЮС", 2002. - 240 с.
5. Глибовець М.М., Олецький О.В. Штучний інтелект. - К.:Академія, 2002. - 366 с.
6. Устенко С.А. Функціональне і логічне програмування. Частина 2. Логічне програмування: Навч. пос. – Миколаїв, УДМТУ, 1998. – 49 с.

Допоміжна

1. Архангельский А.Я. Программирование в C++ Builder, 7 изд. - М.: «Бином», 2010. - 1304 с.
2. В.П. Дьяконов. MATLAB 6.5 SP1/7 + Simulink 5/6. Основы применения. Серия "Библиотека профессионала". - м.: СОЛОН-Пресс, 2005. - 800 с.

Лекція 1. Введення до штучного інтелекту

Інтелектуальність в основному пов'язана з раціональною діяльністю. У ідеальному випадку інтелектуальний агент в будь-якій ситуації робить найкращу можливу дію. У подальшому викладі розглядається проблема створення агентів, які є інтелектуальними саме в цьому сенсі.
Філософи заклали основи штучного інтелекту, сформулювавши ідеї, що мозок в певних стосунках нагадує машину, що він оперує знаннями, закодованими на якійсь внутрішній мові, і що мислення може використовуватися для вибору найкращих дій, що робляться.
Математиків надали інструментальні засоби для маніпулювання висловами, що володіють логічною достовірністю, а також недостовірними ймовірнісними висловами. Крім того, вони заклали основу не лише розуміння того, що є обчисленнями, але і формування міркувань про алгоритми.
Економісти формалізували проблему ухвалення рішень, що максимізували очікуваний виграш для особи, що приймає рішення.
Психологи підтвердили ідею, що люди і тварини можуть розглядатися як машини обробки інформації.
Лінгвісти показали, що процеси використання природної мови укладаються в цю модель.
Комп'ютерні інженери надали артефакти, завдяки яким стало можливим створення додатків штучного інтелекту. Зазвичай програми штучного інтелекту мають великі розміри, і не могли б працювати без тих значних досягнень в підвищенні швидкодії і об'єму пам'яті, які були досягнуті в комп'ютерній індустрії.
Теорія управління присвячена проектуванню пристроїв, які діють оптимально на основі зворотного зв'язку з середовищем. Спочатку математичні інструментальні засоби теорії управління вельми відрізнялися від вживаних в штучному інтелекті, але ці наукові області все більше зближуються.
Історія штучного інтелекту характеризується періодами успіху і невиправданого оптимізму, за якими слідувало зниження інтересу і скорочення фінансування. У ній також були періоди, коли з'являлися нові творчі підходи, а потім кращі з них систематично удосконалювалися.
Штучний інтелект розвивався швидше, ніж зазвичай, в минуле десятиліття, оскільки в цій області стали ширшими застосовуватися наукові методи експериментування і порівняння підходів.
Останні досягнення на дорозі розуміння теоретичних основ інтелектуальності нерозривно пов'язані з розширенням можливостей реальних систем. Окремі підобласті штучного інтелекту стали більшою мірою інтегрованими, а сам штучний інтелект успішно знаходить загальне підґрунтя з іншими науковими дисциплінами.

Лекція 2. Інтелектуальні агенти

Агентом є щось сприймає і діє в певному середовищі. Функція агента визначає дію, що робиться агентом у відповідь на будь-яку послідовність актів сприйняття.
Показники продуктивності оцінюють поведінку агента в середовищі. Раціональний агент діє так, щоб максимізувати очікувані значення показників продуктивності, з врахуванням послідовності актів сприйняття, отриманої агентом до даного моменту.
Специфікація проблемного середовища включає визначення показників продуктивності, зовнішнього середовища, виконавчих механізмів і датчиків. Першим етапом проектування агента завжди має бути визначення проблемного середовища з найбільшою можливою повнотою.
Варіанти проблемного середовища класифікуються по декільком важливій розмірності. Вони можуть бути повністю або частково спостережуваними, детермінованими або стохастичними, епізодичними або послідовними, статичними або динамічними, дискретними або безперервними, а також одноагентнимі або мультіагентнимі.
Програма агента реалізує функцію агента. Існує цілий ряд основних проектів програм агента, відповідних характеру явно сприйманої інформації, яка використовується в процесі ухвалення рішення. Різні проекти характеризуються різною ефективністю, компактністю і гнучкістю. Вибір найбільш відповідного проекту програми агента залежить від характеру середовища.
Прості рефлексні агенти відповідають безпосередньо на акти сприйняття, тоді як рефлексні агенти, засновані на моделі, підтримують внутрішній стан, досліджуючи ті аспекти середовища, які не спостерігаються в поточному акті сприйняття. Агенти, що діють на основі мети, організовують свої дії так, щоб досягти своїх цілей, а агенти, що діють з врахуванням корисності, намагаються максимізувати свою власну очікувану "задоволеність". Всі агенти здатні покращувати свою роботу завдяки навчанню.

Лекція 3. Технології пошуку - основа систем розв'язання задач

Перш ніж агент зможе приступити до пошуку рішень, він повинен сформулювати мету, а потім використовувати цю мету для формулювання завдання. Завдання складається з чотирьох частин: початковий стан, безліч дій, функція перевірки мети і функція вартості дороги. Середовище завдання представлене простором станів, а дорога через простір станів від початкового стану до цільового стану є рішенням.

Для вирішення будь-якого завдання може використовуватися єдиний, загальний алгоритм Tree-search; конкретні варіанти цього алгоритму втілюють різні стратегії. Алгоритми пошуку оцінюються на основі повноти, оптимальності, тимчасової і просторової складності. Складність залежить від коефіцієнт галуження в просторі станів, b, і глибини самого поверхневого рішення, d.

При пошуку завширшки для розгортання вибирається самий поверхневий нерозгорнений вузол в дереві пошуку. Цей пошук є повним, оптимальним при одиничних вартостях етапів і характеризується тимчасовою і просторовою складністю O{b^d). У зв'язку з такою просторовою складністю в більшості випадків він стає практично не застосовним. Пошук по критерію вартості аналогічний пошуку завширшки, але передбачає розгортання вузла з найнижчою вартістю дороги, g(n). Він є повним і оптимальним, якщо вартість кожного кроку перевищує деяке позитивне граничне значення e.

При пошуку в глибину для розгортання вибирається найглибший нерозгорнений вузол в дереві пошуку. Цей пошук не є ні повним, ні оптимальним, і характеризується тимчасовою складністю O(b^m) і просторовою складністю O(bm), де m — максимальна глибина будь-якої дороги в просторі станів.

При пошуку з обмеженням глибини на пошук в глибину накладається встановлена межа глибини.

При пошуку з ітеративним поглибленням викликається пошук з обмеженням глибини і кожного разу встановлюються межі, що збільшуються, до тих пір, поки мета не буде знайдена. Цей пошук є повним і оптимальним при одиничних вартостях етапів і характеризується тимчасовою складністюO(b^m) і просторовою складністю O(bm).

Двонаправлений пошук здатний надзвичайно зменшити тимчасову складність, але він не завжди застосовний і може зажадати надто багато простору. Якщо простором станів є граф, а не дерево, то може виявитися доцільною перевірка полягань, що повторюються, в дереві пошуку. Алгоритм Graph-search усуває всі дублюючі стани.

Якщо середовище є частково спостережуваним, то агент може застосувати алгоритми пошуку в просторі довірчих станів, або безлічі можливих станів, в яких може знаходитися агент. В деяких випадках може бути сформована єдина послідовність рішення; у інших випадках агентові потрібний план дій в непередбачених ситуаціях, щоб мати можливість впоратися зі всіма невідомими обставинами, які можуть виникнути.

Лекція 4. Інформований пошук на просторі станів

Пошук по першому найкращому збігу зводиться просто до вживання алгоритму Graph-search, в якому для розгортання вибираються нерозгорнені вузли з мінімальною вартістю (відповідно до деякого критерію). У алгоритмах пошуку по першому найкращому збігу зазвичай використовується евристична функція h(n), яка оцінює вартість досягнення вирішення з вузла n.

При жадібному пошуку по першому найкращому збігу розгортаються вузли з мінімальним значенням h (п). Цей пошук не оптимальний, але часто є ефективним.

При пошуку A* розгортаються вузли з мінімальним значенням f (n) = g{n) + h (п). Алгоритм A* є повним і оптимальним, за умови, що гарантована допустимість (для алгоритму Tree-search) або спадкоємність (для алгоритму Graph-search) функції h{n). Просторова складність алгоритму A* все ще залишається дуже високою.

Продуктивність алгоритмів евристичного пошуку залежить від якості евристичної функції. Інколи хороші евристичні функції можна скласти шляхом ослаблення визначення завдання, попереднього обчислення вартості вирішення підзадач і збереження цієї інформації в базі даних шаблонів або навчання на основі досвіду вирішення даного класу завдань.

Алгоритмами RBFS і SMA* є надійні, оптимальні алгоритми пошуку, в яких використовуються обмежені об'єми пам'яті; за наявності достатньої кількості часу ці алгоритми можуть вирішувати такі завдання, які не дозволяє вирішувати алгоритм A*, оскільки вичерпує доступну пам'ять.

Такі методи локального пошуку, як пошук зі сходженням до вершини, діють на основі формулювань повного стану і передбачають зберігання в пам'яті лише невеликої кількості вузлів. Було розроблено також декілька стохастичних алгоритмів, включаючи пошук з емуляцією відпалу, які повертають оптимальні рішення за наявності відповідного графіка "охолоджування" (тобто графіка поступового

зменшення величини випадкового розкиду). Крім того, багато методів локального пошуку можуть використовуватися для вирішення завдань в безперервних просторах.
Генетичним алгоритмом є стохастичний пошук зі сходженням до вершини, в якому супроводиться велика популяція станів. Нові стани формуються за допомогою мутації і схрещування; у останній операції комбінуються пари станів, узятих з цієї популяції.
Проблеми дослідження виникають, якщо агент не має жодного уявлення про те, які полягання і дії в його середовищі. Для безпечно досліджуваних варіантів середовища агенти, що виконують пошук в оперативному режимі, можуть скласти карту і знайти мету, якщо вона існує. Ефективним методом виходу з локальних мінімумів є оновлення евристичних оцінок на основі досвіду.

Лекція 5. Завдання задоволення обмежень

Завдання задоволення обмежень (Constraint Satisfaction Problems — CSP) складаються із змінних з обмеженнями, що накладаються на них. У вигляді завдань CSP можуть бути описані багато важливих завдань реального світу. Структуру будь-якого завдання CSP можна представити у вигляді її графа обмежень. Для вирішення завдань CSP зазвичай застосовується пошук з поверненнями — одна з форм пошуку в глибину.
Незалежними від області визначення методами виявлення того, яка змінна має бути вибрана наступною в ході пошуку з поверненнями, є евристики, засновані на визначенні мінімальної кількості значень, що залишилися, і статечні евристики. Для впорядкування значень змінною може застосовуватися евристика з найменш обмежувальним значенням.
У алгоритмі пошуку з поверненнями можна набагато скоротити коефіцієнт галуження завдання, поширюючи наслідки часткових привласнень, сформованих в ході роботи цього алгоритму. Простим методом такого поширення є попередня перевірка. Потужнішим методом є забезпечення сумісності дуг, але вживання цього методу може виявитися дорожчим.
Повернення відбувається після того, як для змінної не можна більше знайти допустиме привласнення. При використанні зворотного переходу, керованого конфліктами, повернення відбувається безпосередньо до джерела проблеми, що виникла в процесі пошуку. Для вирішення завдань з обмеженнями вельми успішно використовується локальний пошук на основі евристики з мінімальними конфліктами.
Складність рішення задачі CSP в значній мірі залежить від структури її графа обмежень. Завдання з деревовидною структурою можуть бути вирішені за лінійний час. Метод визначення умов формування безлічі розриву циклу дозволяє перетворити завдання CSP загального вигляду в завдання з деревовидною структурою і є дуже ефективним, якщо існує можливість знайти невелику безліч розриву циклу. Методи деревовидної декомпозиції передбачають перетворення завдання CSP в дерево підзадач і є ефективними, якщо ширина дерева графа обмежень мала.

Лекція 6. Пошук рішення в умовах протидії

Будь-яка гра може бути визначена за допомогою початкового полягання (яке показує, як здійснюється підготовка дошки до гри), допустимих дій в кожному стані, перевірки термінального стану (що дозволяє визначити, коли гра закінчена) і функції корисності, яка застосовується до термінальних станів.
У іграх з двома гравцями і нульовою сумою, що характеризуються повною інформацією, для вибору оптимальних ходів за допомогою перебору вузлів в глибину в дереві гри може використовуватися алгоритм мінімаксного пошуку.
Алгоритм альфи-бети-пошуку обчислює такі ж оптимальні ходи, як і алгоритм мінімаксного пошуку, але дозволяє досягти набагато більшої ефективності, видаляючи піддерева, які, ймовірно, не потрібні для пошуку рішення.
Зазвичай не представляється можливим розглядати все дерево гри (навіть за допомогою альфи-бети-пошуку), тому необхідно в якійсь крапці зупиняти пошук і застосовувати функцію оцінки, що дозволяє визначити наближене значення корисності деякого стану.
Ведення ігор з елементами випадковості можна здійснити за допомогою розширення алгоритму мінімаксного пошуку, в якому оцінюються вузли жеребкування шляхом визначення середньої корисності всіх їх дочірніх вузлів з врахуванням вірогідності кожного дочірнього вузла.
Для визначення оптимальних ходів в іграх з неповною інформацією, таких як бридж, необхідно формувати міркування про поточний і майбутній довірчих станах для кожного гравця. Одна з простих апроксимацій може бути отримана шляхом усереднювання значення даної дії зі всіх можливих

конфігурацій бракуючої інформації.

Лекція 7. Логічні агенти

Інтелектуальним агентам потрібні знання про світ для того, щоб вони чи міг виробляти хороші рішення. Знання містяться в агентах у формі висловів на мові представлення знань, які зберігаються в базі знань.
Агент на основі знань складається з бази знань і механізму логічного виводу. Він діє шляхом збереження висловів про світ в своїй базі знань, використання механізму логічного виводу для здобуття нових висловів і вживання цих висловів для ухвалення рішення про те, яку дію слід виконати.
Мова подання визначається за допомогою його синтаксису, який задає структуру висловів, і його семантики, яка визначає істинність кожного вислову в кожному з можливих світів, або модель цього вислову. Для розуміння процесу формування логічних міркувань украй поважно визначити, як зв'язані логічні следствія різних висловів.
З вислову a слідує інший вислів β , якщо β є достеменним у всіх світах, де істинно a. Еквівалентні визначення засновані на понятті допустимості вислову a => β і нездійсненності вислову $\alpha \wedge \neg \beta$.
Логічний вивід — це процес здобуття нових висловів із старих. Несуперечливі алгоритми логічного виводу забезпечують здобуття лише таких висловів, які є логічними следствіями; повні алгоритми забезпечують здобуття всіх висловів, що є логічними следствіями.
Пропозіциональная логіка — це дуже простий мова, що складається з пропозіциональних символів і логічних в'язок. Ця мова дозволяє розглядати вислови, відносно яких відомо, чи є вони достеменними, помилковими або мають повністю невідоме логічне значення.
За наявності постійного словника пропозіциональних символів безліч можливих моделей залишається кінцевою, тому логічне слідство можна перевірити, перебираючи моделі. Ефективні алгоритми логічного виводу на основі перевірки по моделі для пропозіциональной логіки включають методи пошуку з поверненнями і локального пошуку і часто дозволяють дуже швидко вирішувати крупні завдання.
Правилами логічного виводу є шаблони несуперечливого логічного виводу, які можуть використовуватися для пошуку доказів. Правило резолюції дозволяє створити повний алгоритм логічного виводу для баз знань, які представлені в кон'юнктивній нормальній формі. Прямий логічний вивід і зворотний логічний вивід — це алгоритми логічного виводу, які надзвичайно природно личать для баз знань, представлених у формі хорновських виразів.
На основі пропозіционально логіки можуть бути створені агенти двох типів: у агентах на основі логічного виводу для стеження за світом і виявлення його прихованих властивостей використовуються алгоритми логічного виводу, тоді як в агентах на основі логічних схем вислову представлені у вигляді бітів в регістрах, а оновлення вмісту регістрів відбувається в результаті поширення сигналів за логічними схемами.
Пропозіциональна логіка є досить ефективною для вирішення деяких завдань в окремо взятому агентові, але не дозволяє розповсюдити свою дію в масштабах таких варіантів середовища, які володіють необмеженими розмірами, оскільки не має достатньої виразної потужності, яка дозволила б коротко виразити тимчасові, просторові і інші універсальні шаблони зв'язків між об'єктами.

Лекція 8. Логіка першого порядку

Мови представлення знань мають бути декларативними, композиційними, виразними, незалежними від контексту і несуперечливими. Різні варіанти логіки відрізняються один від одного по своєму онтологічному вкладу і епістемологическому вкладу.
Пропозіциональная логіка вносить вклад лише до загального об'єму відомостей, що стосуються існування фактів, а логіка першого порядку дозволяє нарощувати об'єм відомостей, що стосуються існування об'єктів і стосунків, тому набуває значної виразної потужності.
Можливий світ, або модель, для логіки першого порядку визначається безліччю об'єктів, стосунками між ними і функціями, які можуть до них застосовуватися.
Константні символи іменують об'єкти, предикативні символи іменують стосунки, а функціональні символи іменують функції. Інтерпретація задає відображення між символами і моделлю. У складних термах функціональні символи застосовуються до термів для іменування об'єкту. Істинність вислову визначається за наявності інтерпретації і моделі.
Атомарний вислів складається з предиката, вживаного до одному або декількох термів; воно стає

достеменним тоді і лише тоді, коли має місце відношення, позначене предикатом, між об'єктами, позначеними його термами. У складних висловах, так само як і в пропозиціональной логіці, використовуються в'язки, а вислови з кванторами дозволяють виражати загальні правила.
Для розробки бази знань в логіці першого порядку потрібно ретельно провести процес аналізу проблемної області, вибору словника і складання аксіом, необхідних для підтримки бажаних процедур логічного виводу.

Лекція 9. Логічний вивід в логіці першого порядку

У першому підході використовується правило логічного виводу для конкретізациі кванторів в цілях перетворення завдання логічного виводу у форму пропозиціональной логіки. Як правило, цей підхід характеризується дуже низькою швидкодією.
Використання уніфікації для виявлення відповідних підстановок для змінних дозволяє усунути етап конкретизації в доказах першого порядку, внаслідок чого цей процес стає набагато ефективнішим.
У піднятій версії правила відділення уніфікація застосовується для здобуття природного і потужного правила логічного виводу — узагальненого правила відділення. У алгоритмах прямого логічного виводу і зворотного логічного виводу це правило застосовується до безлічі певних виразів.
Узагальнене правило відділення є повним стосовно певних виразів, але проблема логічного слідства залишається напіввирішуваною. Для програм Datalog, що складаються з певних виразів без функцій, проблема логічного слідства вирішувана.
Прямий логічний вивід використовується в дедуктивних базах даних, де він може поєднуватися з реляційними операціями баз даних. Він також застосовується в продукційних системах, які забезпечують ефективне оновлення за наявності дуже великих наборів правив. Прямий логічний вивід для програм Datalog є повним і виконується за час, визначуваний поліноміальною залежністю.
Зворотний логічний вивід використовується в системах логічного програмування, таких як Prolog, в яких реалізована складна технологія компіляції для забезпечення дуже швидкого логічного виводу.
Недоліками зворотного логічного виводу є зайві етапи логічного виводу і безконечні цикли; ці недоліки можна усунути шляхом запам'ятовування.
Узагальнене правило логічного виводу на основі резолюції дозволяє створити повну систему доказу для логіки першого порядку з використанням баз знань в кон'юнктивній нормальній формі.
Існує декілька стратегій скорочення простору пошуку для систем з резолюцією без втрати повноти. Ефективні засоби автоматичного доказу теорем на основі резолюції використовувалися для доказу цікавих математичних теорем, а також для перевірки і синтезу програмних і апаратних засобів.

Лекція 10. Логічні моделі представлення знань

Для великомасштабного представлення знань потрібна онтологія загального призначення, що дозволяє організувати і зв'язати воєдино різні спеціалізовані галузі знань. Онтологія загального призначення повинна охоплювати широкий круг знань і бути здатною, в принципі, представити будь-яку проблемну область.
У лекції дається верхня онтологія, заснована на категоріях і численні подій. Тут розглядалися структуровані об'єкти, час і простір, зміни, процеси, речовини і переконання.
Дії, події і час можуть бути представлені або за допомогою ситуаційного числення, або із застосуванням виразніших засобів вистави, таких як числення подій і числення флюєнтних висловів. Такі вистави дають можливість агентові складати плани за допомогою логічного виводу.
Розумові стани агентів можуть бути представлені рядками, які описують їх переконання.
У лекції представлений детальний аналіз проблемної області здійснення покупок в Internet, досліджена загальна онтологія і показано, як знання проблемної області можуть використовуватися торгівельним агентом.
Описані системи представлення загального призначення, такі як семантичні мережі і описові логіки, що дозволяють організовувати ієрархії категорій. З цими системами пов'язана важлива форма логічного виводу, звана спадкоємством, дозволяюча визначати логічним шляхом властивості об'єктів на підставі даних про їх приналежність до категорій.
Припущення про замкнутий світ, будучи реалізованим в логічних програмах, надає простий спосіб уникнути необхідності задавати великі об'єми негативної інформації. Таку інформацію найпростіше інтерпретувати як задану за умовчанням, яка може бути пері визначена за допомогою додаткової інформації.

цілому для надання можливості здійснювати формування міркувань за умовчанням призначені немонотонні логіки, такі як логіка непрямого опису і логіка умовчання. Вживання програмування безлічі відповідей дозволяє прискорити немонотонний логічний вивід, багато в чому аналогічно тому, як використання алгоритму WALKSAT прискорює пропозіциональний логічний вивід.
Системи підтримки істинності дозволяють ефективно здійснювати оновлення і передивляються баз знань.

Лекція 11. Основи планування агентів

Системи планування засновані на використанні алгоритмів вирішення завдань, які застосовуються до явних представлень станів і дій в пропозіциональной логіці (або логіці першого порядку). Такі вистави забезпечують можливість здобуття ефективних евристик, а також розробки потужних і гнучких алгоритмів для вирішення завдань планування.
У мові Strips застосовуються описи дій в термінах їх передумов і результатів, а також опису початкових і цільових станів у вигляді кон'юнкцій позитивних літералів. У мові ADL деякі обмеження мови Strips ослаблені і допускається використання диз'юнкції, заперечення і кванторів.
Пошук в просторі станів може діяти в прямому (прогресивному) напрямі або у зворотному напрямі. Ефективні евристики можуть бути отримані шляхом прийняття припущення про незалежність подцелей, а також за допомогою різних ослабінь завдання планування.
У алгоритмах планування з частковим впорядкуванням (Partial-order Planning ~ POP) простір планів досліджується без прагнення до створення повністю впорядкованої послідовності дій. Такі алгоритми діють у зворотному напрямі від мети, додаючи в план дії, потрібні для досягнення кожної подцелі. Такий підхід стає особливо ефективним при вирішенні завдань, прийнятних для використання підходу за принципом "розділяй і володарюй".
Граф планування може формуватися інкрементно, починаючи з початкового стану. Кожен його рівень містить надмножина всіх літералів або дій, які можуть виявлятися на даному тимчасовому етапі. Крім того, на кожному рівні умовно задаються стосунки взаємного виключення, або стосунки, що взаємно виключають, між літералами або діями, які не можуть відбуватися одночасно. Графи планування дозволяють отримувати корисні евристики для планувальників в просторі станів і планувальників з частковим впорядкуванням і можуть використовуватися безпосередньо в алгоритмі Graphplan.
Алгоритм Graphplan обробляє граф планування, використовуючи зворотний пошук для витягання плану. Цей алгоритм допускає певне часткове впорядкування між діями.
У алгоритмі Satplan завдання планування перетвориться в пропозіциональниє аксіоми і після цього до них застосовується алгоритм перевірки здійснимості для пошуку моделі, відповідної дійсному плану. Було розроблено декілька різних пропозіциональних вистав, що володіють різною мірою компактності і ефективності.
Кожен з основних підходів до планування має свої прибічники, і ще не досягнута повна згода в тому, який з цих підходів є найкращим. Конкуренція між цими підходами і їх взаємне збагачення привели до істотного підвищення ефективності систем планування.

Лекція 12. Планування і здійснення дій на реальному світі

У багатьох діях споживаються ресурси, такі як гроші, паливо або сировина. Ці ресурси зручно в цілому розглядати як числові величини, а не намагатися міркувати, скажімо, про кожну окрему монету або купюру у всьому світі. Дії здатні виробляти і споживати ресурси, тому зазвичай дешевше і ефективніше перевіряти часткові плани на предмет задоволення в них ресурсних обмежень, перш ніж робити спроби їх подальшого уточнення.
Час — це один з найбільш важливих ресурсів. За його витрачанням можна стежити, застосовуючи спеціалізовані алгоритми складання розкладів або об'єднуючи складання розкладів з плануванням.
планування за допомогою ієрархічної мережі завдань (Hierarchical Task Network — HTN) дозволяє агентові отримати пораду від проектувальника проблемної області у формі правил декомпозиції. Такий підхід забезпечує створення дуже великих планів, потрібних для багатьох реальних застосувань. У стандартних алгоритмах планування передбачається наявність повної і правильної інформації, а також детермінованого, повністю спостережуваного середовища. Це припущення є недійсним в багатьох проблемних областях.
З проблемою неповної інформації в плануванні можна впоратися, використовуючи дії з вживання датчиків для здобуття необхідної інформації. Умовні плани дозволяють агентові отримувати за допомогою датчиків інформацію про світ під час виконання плану для визначення того, по якій гілці

плану він повинен слідувати далі. В деяких випадках може використовуватися планування без використання датчиків або погоджене планування для формування плану, який під час свого виконання не вимагає вживання результатів сприйняття. І плани без використання датчиків, і умовні плани можуть бути сформовані по методу пошуку в просторі довірчих станів.
Неправильна інформація приводить до того, що передумови дій і планів залишаються невиконаними. Контроль виконання дозволяє виявляти порушення передумов, створюючи передумови успішного здійснення плану.
У перепланірующем агентові використовується контроль виконання і в міру необхідності здійснюються відновні дії для повернення до первинного плану. Безперервно плануючий агент створює нові цілі в процесі своєї діяч ності і реагує на зміни ситуації в реальному часі.
Мультіагентноє планування необхідне, якщо в середовищі є інші агенти, з якими доводиться кооперувати, конкурувати або координувати свої дії. У багатотільному плануванні формуються спільні плани з використанням ефективної декомпозиції описів спільних дій, але це планування повинне доповнюватися певною формою координації, якщо два кооперативні агенти повинні погоджувати один з одним, яким із спільних планів слід виконати.

Лекція 13. Невизначеність

Невизначеність виникає і унаслідок економії зусиль, і через відсутність знань. Невизначеності не можна уникнути в складних, динамічних або важкодоступних світах. Наявність невизначеності означає, що багато спрощень, можливих в дедуктивному логічному виводі, стають більше не допустимими.
У оцінках вірогідності виражається нездатність агента прийти до певного рішення, що стосується істинності вислову. Вірогідність є вираженням мірі упевненості агента. До основних типів імовірнісних тверджень відносяться твердження, що стосуються апріорної вірогідності і умовної вірогідності простих і складних висловів.
У оцінках вірогідності виражається нездатність агента прийти до певного рішення, що стосується істинності вислову. Вірогідність є вираженням мірі упевненості агента. До основних типів імовірнісних тверджень відносяться твердження, що стосуються апріорної вірогідності і умовної вірогідності простих і складних висловів.
Якщо повний спільний розподіл доступний, воно може використовуватися для здобуття відповідей на запити шляхом підсумовування елементів з даними про атомарні події, відповідні висловам запиту. Наявність властивості абсолютної незалежності між підмножинами випадкових змінних дозволяє факторізовать повний спільний розподіл на менші спільні розподіли. Це дає можливість значно зменшити складність, але рідко зустрічається на практиці.
Правило Байеса дозволяє обчислювати невідому вірогідність з відомої умовної вірогідності, зазвичай в причинному напрямі. За наявності багаточисельних свідоцтв вживання правила Байеса, як правило, приводить до виникнення таких же проблем масштабування, які виникають при використанні повного спільного розподілу.
Властивість умовної незалежності, викликана наявністю прямих причинних зв'язків в даної проблемної області, може забезпечити факторизацію повного спільного розподілу на менші умовні розподіли. У наївній байесовській моделі передбачається наявність умовної незалежності всіх змінних дії, якщо задана одна змінна причини; розміри цієї моделі збільшуються лінійно, залежно від кількості результатів.
Агент в печері Вія може розраховувати вірогідність спостережених об'єктів світу і використовувати їх для ухвалення кращих рішень в порівнянні з тими, які приймає простий логічний агент.

Лекція 14. Формування міркувань в умовах невизначеності

Байесовская мережа — це орієнтований ациклічний граф, вершини якого відповідають випадковим змінним; з кожною вершиною пов'язаний розподіл умовної вірогідності для цієї вершини, якщо дані її батьківські вершини. Байесовские мережі лежать в основі зручного способу представлення стосунків умовної незалежності в даної проблемної області.
Будь-яка байесовская мережа задає повний спільний розподіл; кожен елемент спільного розподілу визначається як твір відповідних елементів в локальних умовних розподілах. Байесовская мережа часто дозволяє експоненціально зменшити розміри імовірнісної вистави в порівнянні з повним спільним розподілом.
Багато умовних розподілів можуть бути представлені компактно за допомогою канонічних сімейств

розподілів. Цілий ряд канонічних розподілів використовується в гібридних байесовських мережах, які включають і дискретні, і безперервні змінні.
Під імовірнісним виводом в байесовських мережах мається на увазі обчислення розподілу вірогідності безлічі змінних запиту, якщо дана безліч змінних свідоцтва. Алгоритми точного імовірнісного виводу, такі як алгоритм усунення змінної, дозволяють обчислювати суми творів умовної вірогідності настільки ефективно, наскільки це можливо.
У полідеревах (одинзв'язних мережах) точний імовірнісний вивід вимагає часу, лінійно залежного від розміру мережі. А в загальному випадку проблема такого виводу нерозв'язна. Методи стохастичної апроксимації, такі як оцінка ваги з врахуванням правдоподібності і метод Монте-Карло на основі ланцюга Марков, дозволяють отримати прийнятні оцінки дійсної апостеріорної вірогідності в мережі і здатні впоратися з набагато крупнішими мережами в порівнянні з точними алгоритмами.
Теорія вірогідності може застосовуватися у поєднанні з ідеями представлення знань, запозиченими з логіки першого порядку, для створення дуже потужних систем формування міркувань в умовах невизначеності. Реляційні моделі вірогідності (Relational Probability Model — RPM) включають обмеження на засоби вистави, які гарантують здобуття повністю певного розподілу вірогідності, який може бути виражене у вигляді еквівалентної байесовської мережі.
Були запропоновані цілий ряд альтернативних систем для формування міркувань в умовах невизначеності. Взагалі кажучи, істиннісно-функціональні системи не дуже добре личать для таких міркувань.

Лекція 15. Імовірнісні міркування в часі

Зміну стану світу можна врахувати, використовуючи безліч випадкових змінних для представлення цього стану в кожен момент часу. Такі вистави можуть бути спроектовані так, щоб вони задовольняли властивості марковості, згідно з яким майбутнє не залежить від минулого, якщо дано сьогодення. У поєднанні з припущенням про те, що даний процес є стаціонарним (тобто таким, що його закони не змінюються в часом), це дозволяє набагато спростити представлення.
Тимчасова імовірнісна модель може розглядатися як модель переходу, що містить, яка описує процес розвитку, і модель сприйняття, що описує процес спостереження. Основними завданнями імовірнісного виводу в тимчасових моделях є фільтрація, передбачення, згладжування і визначення за допомогою обчислень найбільш вірогідного пояснення. Кожне з цих завдань може бути вирішена з використанням простих, рекурсивних алгоритмів, час виконання яких лінійно залежить від довжини даної послідовності.
Трохи більш детально було описано три сімейства тимчасових моделей: приховані марківські моделі, фільтри Кальмана і динамічні байесовськие мережі (остання модель включає дві перших як окремі випадки). Двома важливими застосуваннями для тимчасових імовірнісних моделей є розпізнавання мови і стеження.
Якщо не прийняті особливі припущення, як при використанні фільтрів Кальмана, точний імовірнісний вивід за наявності багатьох змінних стану, мабуть, стає нездійсненним. Створюється враження, що на практиці ефективним алгоритмом апроксимації є алгоритм фільтрації часток.

Лекція 16. Ухвалення простих рішень агентом

Теория вероятностей описывает, в чем должен быть уверен агент согласно полученному свидетельству, теория полезности показывает, к чему должен стремиться агент, а теория принятия решений позволяет объединить подходы этих двух теорий для определения того, что должен делать агент. Теория принятия решений может использоваться для создания систем, которые принимают решения, рассматривая все возможные действия и выбирая из них именно то, которое приводит к наилучшему ожидаемому результату. Така система відома під назвою раціонального агента.
Теорія полезностей показує, що агент, що керується стосунками переваги між лотереями, сумісними з безліччю простих аксіом, може бути описаний як що володіє функцією корисності; крім того, агент вибирає дії так, щоб можна було максимізувати його очікувану корисність.
Теорія багатоатрибутної корисності присвячена винавчанню корисності, яка залежить від декількох різних атрибутів станів. Стохастичним домінуванням є особливо зручний метод ухвалення несуперечливих рішень навіть за відсутності точних значень корисності для атрибутів.
Мережі ухвалення рішень є простою формальною системою для опису і вирішення завдань ухвалення рішень. Вони є природним розширенням байесовських мереж і, окрім вузлів жеребкування, містять вузли рішення і вузли корисності.

Інколи для вирішення завдання доводиться займатися пошуком додаткової інформації, перш ніж приймати рішення. Вартість інформації визначена як очікуване підвищення корисності в порівнянні з ухваленням рішень без цієї інформації. Експертні системи, в яких передбачається використання інформації про корисність, володіють додатковими можливостями в порівнянні з системами, в яких застосовується виключно імовірнісний вивід.
Вони не лише володіють здатністю виробляти рішення, але і можуть використовувати вартість інформації для визначення того, чи слід прагнути до її здобуття, а також здатні розрахувати чутливість своїх рішень до невеликих змін в оцінках вірогідності і корисності.

Лекція 17. Ухвалення складних рішень агентом

Завдання послідовного ухвалення рішень в невизначених варіантах середовища, звані також марківськими процесами ухвалення рішень (Markov Decision Process — MDP), визначаються за допомогою моделей переходу, задаючих імовірнісні результати дій, і функції винагороди, яка показує, яка винагорода відповідає кожному стану.
Корисність послідовності станів є сумою всіх винагород уздовж цієї послідовності, яка, можливо, з часом піддається знеціненню. Рішенням задачі MDP є стратегія, в якій з кожним станом, досяжним для агента, зв'язано деяке рішення. Оптимальна стратегія максимізувала корисність послідовності станів, що зустрічається, при її здійсненні.
Корисністю стану є очікувана корисність послідовностей станів, що зустрічаються при здійсненні оптимальної стратегії, починаючи з цього стану. Алгоритм ітерації по значеннях для вирішення завдань MDP діє за принципом ітеративного вирішення рівнянь, що зв'язують корисності кожного стану з полезностями його сусідніх станів.
У алгоритмі ітерації по стратегіях чергуються етап обчислення полезностей станів згідно поточної стратегії і етап удосконалення поточної стратегії по відношенню до поточних полезностям.
Завдання MDP в частково спостережуваному середовищі, або завдання POMDP, є набагато важчими для вирішення, чим завдання MDP. Вони можуть бути вирішені шляхом перетворення в завдання MDP в безперервному просторі довірчих станів. Оптимальна поведінка при вирішенні завдань POMDP повинна передбачати збір інформації для зменшення невизначеності і тому ухвалення кращих рішень в майбутньому.
Для варіантів середовища POMDP може бути створений агент, що діє на основі теорії рішень. У такому агентові для представлення моделі переходу і моделі спостереження для оновлення його довірчого стану і проектування можливих послідовностей дій в прямому напрямі використовується динамічна мережа ухвалення рішень.
Теорія ігор описує раціональну поведінку для агентів в тих ситуаціях, в яких одночасно взаємодіють безліч агентів. Рішеннями для ігор є рівноваги Неша — профілі стратегій, в яких жоден з агентів не має стимул-реакцій, під впливом яких він міг би відхилитися від визначеної для нього стратегії.
Проектування механізму може використовуватися для визначення правив, по яких має бути організоване взаємодія агентів в цілях максимізації деякої глобальної корисності завдяки функціонуванню окремих раціональних агентів. Інколи удається знайти механізми, що дозволяють досягти цієї мети, не вимагаючи від кожного агента, щоб він враховував те, які варіанти вибрані іншими агентами.

Лекція 18. Навчання на основі спостережень

Навчання приймає багато різних форм залежно від характеру продуктивного елементу, компонента, підмета удосконаленню, і доступному зворотному зв'язку. Якщо доступний зворотний зв'язок або від вчителя, або від середовища, що дозволяє набувати правильних значень, що відносяться до прикладів, то завдання навчання відноситься до типа контрольованого навчання.
Таке завдання, зване також індуктивним навчанням, зводиться до винавчання деякої функції на прикладах її вхідних і вихідних даних. Винавчання функції з дискретними значеннями називається класифікацією; винавчання безперервної функції називається регресією.
Індуктивне навчання зводиться до пошуку сумісної гіпотези, яка узгоджується з прикладами. При цьому повинен дотримуватися принцип бритви Оккама, згідно з яким слід завжди вибирати найбільш просту сумісну гіпотезу. Складність рішення цієї задачі залежить від вибраного способу вистави.
Дерева рішень дозволяють представити будь-які булеві функції. Евристика, що визначає приріст інформації, може стати основою ефективного методу пошуку простого, сумісного дерева рішень.
Продуктивність повчального алгоритму вимірюється кривою навчання, яка показує прогностичну

точність вживання алгоритму до перевірочної безлічі як функцію від розміру повчальної безлічі.
Методи навчання ансамблю дерев рішень, такі як посилення, часто показують вищу продуктивність в порівнянні з методами навчання окремих дерев рішень. Для аналізу вибіркової і обчислювальної складності індуктивного навчання застосовується теорія обчислювального навчання. Ця теорія дозволяє знайти компроміс між виразністю мови гіпотез і легкістю навчання.

Лекція 19. Вживання знань у навчанні

Успішне вживання апріорних знань у навчанні показує, що ще перспективнішим є кумулятивне навчання, в якому агенти, що виучуються, підвищують свою здібність до навчання у міру того, як набувають все більше і більше знань. Апріорні знання сприяють навчанню, оскільки дозволяють усувати помилкові гіпотези, які в іншому випадку вважалися б сумісними, а також "доповнювати" пояснення прикладами, що приводить до створення коротших гіпотез. Такі можливості часто дозволяють добитися прискореного навчання за допомогою меншої кількості прикладів.
Апріорні знання можуть виконувати різні функції в логічному виводі, а описи цих функцій, виражені у формі обмежень логічного слідства, дозволяють визначити цілий ряд всіляких методів навчання.
У методі навчання на основі пояснення (Explanation-based Learning — EBL) передбачається витягання загальних правил з окремих прикладів шляхом формування пояснень для цих прикладів і узагальнення пояснень. Методом EBL є дедуктивний метод, за допомогою якого знання основних принципів перетворяться в корисні, ефективні експертні знання спеціального призначення.
У методі навчання з врахуванням рельовантності (Relevance-based Learning — RBL) використовуються апріорні знання у формі визначень для виявлення релевантних атрибутів, що приводить до створення зменшеного простору гіпотез і прискорення навчання. Метод RBL забезпечує також створення дедуктивних узагальнень на основі окремих прикладів.
У методі індуктивного навчання на основі знань (Knowledge-based Inductive Learning — KBIL) передбачається пошук індуктивних гіпотез, що дозволяють пояснювати безліч результатів спостережень за допомогою фонових знань.
У методах індуктивного логічного програмування (Inductive Logic Programming — ILP) метод KBIL застосовується до знань, виражених в логіці першого порядку. Методи ILP дозволяють отримувати в процесі навчання такі реляційні знання, яке не можуть бути виражені в системах на основі атрибутів.
Індуктивне логічне програмування може здійснюватися в рамках низхідного підходу, в якому здійснюється уточнення дуже загального правила, або висхідного підходу, зворотного по відношенню до процесу дедуктивного логічного виводу.
За допомогою методів ILP природним чином виробляються нові предикати, за допомогою яких можуть бути виражені нові ємкі теорії, тому вони відкривають можливість створення систем формування наукових теорій загального призначення.

Лекція 20. Статистичні методи навчання

У методах байесовського навчання завдання навчання формулюється як один з видів імовірнісного виводу, в якому спостереження використовуються для оновлення розподілів апріорної вірогідності по гіпотезах. Таким підходом є хороший спосіб реалізації принципу бритви Оккама, але швидко стає важкоздійсненним при зростанні складності простору гіпотез.
У навчанні на основі максимальної апостеріорної вірогідності (Maximum A Posteriori — MAP) вибирається єдина гіпотеза, найбільш вірогідна згідно з наявними даними. При цьому все ще використовується розподіл апріорної вірогідності гіпотези і сам цей метод часто є легше здійсненним, ніж повне байесовськоє навчання.
У навчанні з врахуванням максимальної правдоподібності вибирається гіпотеза, яка максимізувала правдоподібність даних; цей метод еквівалентний методу навчання MAP з рівномірним розподілом апріорної вірогідності.
У простих випадках, таких як лінійна регресія і повністю спостережувані байесовськие мережі, рішення з врахуванням максимальної правдоподібності можна легко знайти в замкнутій формі. Особливо ефективним методом, який добре масштабується, є наївне байесовськоє навчання.
Якщо деякі змінні приховані, то рішення з локальною максимальною правдоподібністю можна знайти з використанням алгоритму ЇМ. До числа додатків відповідного методу входить кластеризація за допомогою змішаних розподілів гаусів, навчання байесовських мереж і прихованих марківських моделей.
Визначенням в процесі навчання структур байесовських мереж є приклад методу вибору моделі. У

цьому методі зазвичай передбачається дискретний пошук в просторі структур. При цьому необхідно якимсь чином забезпечити пошук компромісу між складністю моделі і мірою її відповідності даним.
У моделях на основі екземпляра розподіл представлений з використанням колекції повчальних екземплярів. Таким чином, кількість параметрів зростає із збільшенням розмірів повчальної безлічі. У методах найближчої сусідньої крапки здійснюється пошук екземплярів, найближчих до даної крапки, а в ядерних методах формується комбінація всіх екземплярів, зважена по відстані.

ЛАБОРАТОРНІ РОБОТИ

Лабораторна робота No1 . Інтелектуальні агенти

Ціль.

Розглянути та дослідити різні типи раціональних агентів. Порівняти ефективність їх поведінки.

В лабораторній роботі йдеться про «світ пилососу» - віртуальний світ, в якому існує один штучний агент – пилосос (див. тему 1 для опису поведінки агента-пилососа). Його задача – мінімізувати кількість бруду в заданому просторі.

В якості середовища моделювання використати проект «Vacuum Cleaner World» (http://web.ntnu.edu.tw/~tcchiang/ai/Vacuum%20Cleaner%20World.htm). Цей проект реалізований на мові C++ та реалізує середовище задачі і поведінку простого агенту.

Завдання.

1) Необхідно розробити програми раціональних агентів двох типів: простого рефлексного агента та рефлексного агента, якій заснований на моделі. Для створення власного агенту потрібно модифікувати файли agent.cpp та agent.h оригінального проекту. Агент з моделлю середовища може отримувати інформацію про стан середовища через клас Environment (файли environment.cpp/.h)

2) Створити декілька власних карт середовища (на основі прикладу agent.map)

3) Провести серію експериментів для визначення ефективності роботи програм агентів.

Порівняти два створених агенти між собою, а також з оригінальним агентом (який йде в дистрибутиві проекту «Vacuum Cleaner World») та діє випадковим чином. Порівняння зробити для всіх створених карт, а також оригінальної карти agent.map. Вхідним параметром для експериментів вважати час моделювання (наприклад: 100, 500, 1000, 2000, 5000, 10000). За проведеними серіями необхідно визначити:

• середній ступінь бруду (average dirty degree);

• середню використану енергію (average consumed energy).

Результати експериментів представити у вигляді таблиць та графіків.

Звіт

За результатом виконання лабораторної роботи необхідно підготувати звіт. Звіт повинен містити:

1. Постановку задачі

2. Опис принципу роботи розроблених агентів

3. Результати проведення серії експериментів у вигляді таблиць та графіків

4. Аналіз отриманих результатів.

5. Висхідні коди програм агентів.

Звіт подається після здачі роботи в електронному вигляді.

Контрольні питання

Захист лабораторної роботи передбачає вміння давати відповіді на наступні питання:

1. Наведіть означення предметного середовища (PAES)? Які її складові частини?
2. Які є характеристики предметних середовищ?
3. Опишіть принципи роботи простих рефлексних агентів, рефлексних агентів на моделі, агентів на основі цілі та агентів на основі корисності.
4. Опишіть принцип, який покладений в основу агентів, що навчаються. Які складові частини їх архітектури?

Шкала оцінювання

Кожний етап виконання лабораторної роботи

• Відповідь на контрольні питання: 3 бали

• Програмна реалізація алгоритмів: 4 бали

• Проведення серії експериментів та отримані висновки: 2 бали

• Наявність та інформативність звіту: 1 бал

Література

1. Електронний конспект (презентація). Тема 1.

2. Рассел, Норвіг «Штучний інтелект. Сучасний підхід» Глава 2.

Лабораторна робота No2 . Інформативний та неінформативний пошуки

Ціль.

Розглянути та дослідити алгоритми неінформативного та інформативного пошуку. Провести порівняльний аналіз ефективності використання алгоритмів.

Завдання.

1) Реалізувати програму, яка розв'язує поставлену задачу Task за допомогою алгоритму неінформативного пошуку AlgNoInf та алгоритму інформативного пошуку AlgInf, що використовує задану евристичну функцію Func.

Значення параметрів Task, AlgNoInf, AlgInf та Func визначаються за номером варіанту (див. нижче).

Програму реалізувати на довільній мові процедурного програмування (C, C++, C#, Java, ython, Matlab, PHP, ...)

Увага! Алгоритм неінформативного пошуку AlgNoInf реалізовується за принципом «AS IS», тобто так, як є, без додаткових модифікацій (таких як перевірка циклів, наприклад).

2) Провести серію експериментів для вивчення ефективності роботи алгоритмів. Кожний експеримент повинен відрізнятись початковим станом. Серія повинна містити не менше 20 експериментів для кожного алгоритму. За проведеними серіями необхідно визначити:

• середній час пошуку рішення у секундах

• середню кількість згенерованих станів під час пошуку

• середню кількість станів, що зберігаються в пам'яті під час роботи програми

Передбачити можливість обмеження виконання програми за часом (наприклад, 10 хвилин) та використання пам'яті (наприклад, 512 Мб)

Звіт

За результатом виконання лабораторної роботи необхідно підготувати звіт. Звіт повинен містити:

1. Постановку задачі

2. Формулювання проблеми та опис простору станів (див. лекцію 3).

3. Опис евристичної функції оцінки.

4. Результати проведення серії експериментів у вигляді таблиці.

5. Аналіз отриманих результатів (оптимальність, повнота, швидкість, використана пам'ять при роботі алгоритмів).

6. Висхідні коди алгоритмічної частини програм.

Звіт подається після здачі роботи в електронному вигляді.

Контрольні питання

Захист лабораторної роботи передбачає вміння давати відповіді на наступні питання:

1. Яка відмінність між алгоритмами інформативного та неінформативного пошуку?

2. Які алгоритми неінформативного пошуку є повними? Оптимальними?

3. Поясніть необхідність використання евристичних функцій при інформативному пошуку.

4. Якою повинна бути евристична функція для того, щоб алгоритми інформативного пошуку були оптимальними?

5. Які алгоритми інформативного пошуку є повними? Оптимальними? Чому?

6. Поясніть принцип роботи алгоритмів, які реалізовані у лабораторній роботі.

Шкала оцінювання

Кожний етап виконання лабораторної роботи

• Відповідь на контрольні питання: 3 бали

• Програмна реалізація алгоритмів: 4 бали

• Проведення серії експериментів та отримані висновки: 2 бали

• Наявність та інформативність звіту: 1 бал

• Наявність візуалізації роботи алгоритмів: 2 бали.

Література

• Електронний конспект (презентація). Теми 2 та 3.

• Рассел, Норвіг «Штучний інтелект. Сучасний підхід» Глави 3 та 4.

Варіанти

Варіант індивідуального завдання визначається за останніми двома цифрами залікової книжки.

Значення параметрів Task, AlgNoInf, AlgInf та Func для кожного варіанту за таблицею нижче. Використані позначення:

• 8-ферзів – задача розташування 8 ферзів на шахівниці. Зробити реалізацію задачі із повними станами, тобто у кожному стані (початковому включно) міститься всі 8

ферзів. Функція визначення наступника працює наступним чином: обирається один ферзь та переставляється у нову позицію (для неінформативного пошуку – це перша позиція за списком; у інформативному – краща за евристичною функцією).

• 8-puzzle - задача гри у 8. Початковий стан рекомендується отримувати шляхом рухів від цільового стану у зворотній бік (цим буде гарантуватись знаходження розв’язку та глибина дерева пошуку).

• LDFS - Пошук вглиб з обмеженням глибини

• BFS – Пошук вшир

• IDS – Пошук вглиб з ітеративним заглибленням

• A* – Пошук A*

• RBFS – Рекурсивний пошук за першим найкращим збігом

• F1 – кількість пар ферзів, які б’ють один одного з урахуванням видимості (ферзь A може стояти на одній лінії з ферзем B, проте між ними стоїть ферзь C; тому A не б’є B)

• F2 – кількість пар ферзів, які б’ють один одного без урахування видимості

• H1 – кількість фішок, які не стоять на своїх місцях

• H2 – Манхетенська відстань

NoNo	Task	AlgNoInf	AlgInf	Func
1	8-ферзів	LDFS	A*	F1
2	8-puzzle	LDFS	A*	H1
3	8-ферзів	LDFS	RBFS	F1
4	8-puzzle	LDFS	RBFS	H1
5	8-ферзів	BFS	A*	F1
6	8-puzzle	BFS	A*	H1
7	8-ферзів	BFS	RBFS	F1
8	8-puzzle	BFS	RBFS	H1
9	8-ферзів	IDS	A*	F1
10	8-puzzle	IDS	A*	H1
11	8-ферзів	IDS	RBFS	F1
12	8-puzzle	IDS	RBFS	H1
13	8-ферзів	LDFS	A*	F2
14	8-puzzle	LDFS	A*	H2
15	8-ферзів	LDFS	RBFS	F2
16	8-puzzle	LDFS	RBFS	H2
17	8-ферзів	BFS	A*	F2
18	8-puzzle	BFS	A*	H2
19	8-ферзів	BFS	RBFS	F2
20	8-puzzle	BFS	RBFS	H2
21	8-ферзів	IDS	A*	F2
22	8-puzzle	IDS	A*	H2
23	8-ферзів	IDS	RBFS	F2
24	8-puzzle	IDS	RBFS	H2

Лабораторна робота No3 . Локальний пошук

Ціль.

Розглянути та дослідити алгоритми локального пошуку для розв’язання задач оптимізації.

Завдання.

1) Реалізувати програму, яка розв’язує поставлену задачу Task за допомогою алгоритму локального пошуку AlgLocal для повного формулювання. Евристичну функцію обрати довільну для заданої задачі.

Значення параметрів Task, AlgLocal визначаються за номером варіанту (див. нижче).

Програму реалізувати на довільній мові процедурного програмування (C, C++, C#, Java, Python, Matlab, PHP, ...)

2) Провести 20 експериментів роботи алгоритму (для цього вміти генерувати випадковим чином початковий стан), які відрізняються характеристиками роботи алгоритму. За результатами експериментів визначити:

• середню кількість етапів (кроків), які знадобилось для досягнення розв’язку

• середню кількість випадків, коли алгоритм потрапляв в глухий кут (не міг знайти оптимальний розв’язок) – якщо таке можливе

• середній час роботи алгоритму в секундах

3) На додаткові бали. Порівняти результати розв’язання задачі за допомогою алгоритму локального пошуку із результати з лабораторної роботи No2.

Звіт

За результатом виконання лабораторної роботи необхідно підготувати звіт. Звіт повинен містити:

1. Постановку задачі

2. Формулювання проблеми та опис простору станів (див. лекцію 3).

3. Результати роботи алгоритмів (час, кількості етапів та повернень)

4. Висновки по роботі алгоритмів.

5. Висхідні коди алгоритмічної частини програм.

Звіт подається після здачі роботи в електронному вигляді.

Контрольні питання

Захист лабораторної роботи передбачає вміння давати відповіді на наступні питання:

1. Яка відмінність між алгоритмами звичайного пошуку та локального пошуку?

2. Поясніть в чому полягає складність роботи алгоритмів локального пошуку

(пов’язана з ландшафтом станів).

3. Що таке задачі з обмеженнями (CSP)? Як вони формулюються?

4. Які складності виникають при розв’язанні задач у неперервних просторах станів?

5. Поясніть принцип роботи алгоритмів, які реалізовані у лабораторній роботі.

Шкала оцінювання

Кожний етап виконання лабораторної роботи

• Відповідь на контрольні питання: 3 бали

• Програмна реалізація алгоритмів: 4 бали

• Проведення серії експериментів та отримані висновки: 2 бали

• Наявність та інформативність звіту: 1 бал

• порівняння результатів з лабораторною роботою No2: 2 бали

Література

• Електронний конспект (презентація). Тема 4.

• Рассел, Норвіг «Штучний інтелект. Сучасний підхід» Глава 4.

Варіанти

Варіант індивідуального завдання визначається за останніми двома цифрами залікової книжки.

Значення параметрів Task, AlgLocal, Evr1 та Constr для кожного варіанту за таблицею нижче. Використані позначення:

• 8-ферзів – задача розташування 8 ферзів на шахівниці. Зробити реалізацію задачі із повними станами, тобто у кожному стані (початковому включно) міститься всі 8 ферзів. Функція визначення наступника працює наступним чином: обирається один ферзь та переставляється у нову позицію (для неінформативного пошуку – це перша позиція за списком; у інформативному – краща за евристичною функцією).

• 8-puzzle - задача гри у 8. Початковий стан рекомендується отримувати шляхом рухів від цільового стану у зворотній бік (цим буде гарантуватись знаходження розв’язку та глибина дерева пошуку).

• HILL – Стохастичний пошук зі сходженням на вершину із використанням руху вбік (на 100 кроків) та випадковим перезапуском (кількість необхідних разів запуску

визначити самостійно – див. лекцію 5).

• ANNEAL – Локальний пошук із симуляцією відпалу. Робоча характеристика – залежність температури T від часу роботи алгоритму t. Можна розглядати лінійну залежність: T = 1000 - k·t, де k – змінний коефіцієнт.

• BEAM – Локальний променевий пошук. Робоча характеристика – кількість променів k. Експерименти проводи із кількістю променів від 2 до 21.

• TABU – Табу-пошук із короткотерміновою пам’яттю (табу-списком). Розмір списку та час перебування станів у списку визначати експериментальним шляхом

NoNo	Task	AlgLocal
1	8-ферзів	HILL
2	8-puzzle	HILL
3	8-ферзів	ANNEAL
4	8-puzzle	ANNEAL
5	8-ферзів	BEAM
6	8-puzzle	BEAM
7	8-ферзів	TABU
8	8-puzzle	TABU
9	8-ферзів	HILL
10	8-puzzle	HILL
11	8-ферзів	ANNEAL
12	8-puzzle	ANNEAL

ЛАБОРАТОРНА РОБОТА 4. Складні інтелектуальні агенти

Печера Вія

Вій — персонаж української демонології, який найчастіше постає в образі старезного діда з густими й довгими бровами та віями, через які нічого не бачить. Його погляд може бути смертельним для живих істот. Він має згубну, руйнівну силу, яка здатна провалити будинок під землю, а на його місці утворити водоймище. Однак від цієї магічної сили рятує те, що Вій навколо себе нічого не бачить через свої надзвичайно густі та довгі брови й вії

Печера Вія складається із залів, поєднаних проходами. Десь у цій печері бродить Вій, зустріч з яким смерельно небезпечна для будь-якої людини, у тому числі й для Хоми Брута. Але у нього є срібний спис, який можна застосувати тільки раз.

У деяких залах літає паннока на труні. Єдине, що втішає натрапивших на печеру Вія, так те, що можна знайти купку золота.

Ця гра доволі проста у порівнянні з сучасними комп'ютерними іграми, але є достатньо непоганим полігоном для дослідження різноманітних інтелектуальних агентів.

Які ж **показники продуктивності** в цій грі? За те, що Хома знаходить купку золота він отримує +1000 балів. Якщо Хома зіштовхується з панночкою або потрапляє на очі Вєві, то призначається -1000 балів. Крім того Хома віддає -1 бал за кожну свою дію та -10 за використання срібного списа.

Середовище гри. Зали розташовані у вигляді решітки 4X4. Хома завжди починає рух з зали [1,1]. Місцерозташування золота та Вія обираються випадково із рівномірним розподілом з числа квадратів, відмінних від початкового. Крім того, у кожному квадраті, відмінному ві початкового, з ймовірністю 0,2 може літати панночка на труні.

Виконавчі механізми. Хома може рухатися вперед, повертати вліво та вправо на 90°. Хома загине, якщо входить до квадрату, де знаходиться панночка або Вій. Входить до квадрату з мертвийм Вієм безпечно, але при цьому приходиться відчувати неприємний запах.

Рис. 1. Печера Вія.

Спроба пересунення вперед залишається нездійсненною, якщо перед Хомою знаходиться стіна. Щоб взяти об'єкт, що знаходиться у квадраті, де в той же час знаходиться Хома, треба виконати команду ВЗЯТИ. За допомогою дії СТРІЛЯТИ можна пустити стрілу по прямій в тому спрямуванні, куди дивиться Хома. Сріла продовжує рух до тих півр, поки вона не попаде у Вія (й вб'є його), або не вдариться о стіну. А Хоми є тільки одна стріла, тому будь-який ефект можливий від першої дії СТРІЛЯТИ.

Датчики. У Хоми є п'ять датчиків, кожний з яких повідомляє тільки один елемент інформації:

- у квадраті де є Вій, а також в кадратах, безпосередньо (не по діагоналі) сусідніх до цього, Хома відчуває неприємний могільний запах (СМОРІД);
- у квадратах, сусідніх з тими, де літає на труні панночка, Хома буде відчувати рух повітря (ВІТЕРЕЦЬ);
- у квадраті, де є золото, Хома бачить світло (ВІДБЛИСК);
- коли Хома натрапляє на стіну, від відчуває УДАР;
- перед тим, як вражений списом Вій помирає, він випускає страшенний крик, який розноситься по всій печері (КРИК).

Результати актів сприйняття передаються Хомі у вигляді переліку з п'яти елементів. Наприклад, якщо є неприємний запах та рух пвітря, але нема блиску, удару та крику, Хома сприймає результати акту сприйняття - [СМОРІД, ВІТЕРЕЦЬ, НІ, НІ, НІ].

Нижче на рис. 2 наведений приклад ситуації в печері Вія.

Рис. 2. Приклад ситуації у печері Вія.

У першому завданні лабораторної роботи треба визначити середовище печери Вія. Головна складність цього завдання полягає у тому, що наш інтелектуальний агент Хома з самого початку не знає конфігурацію свого середовища. Й бачимо, що для подолання цих перешкод доведеться використовувати логічні міркування.

У більшості варіантів середовища печери Вія для Хоми існує можливість безпечно отримати золото. Але може виникнути ситуація, коли Хомі Бруту придеться обирати: повернутися додому з пустими руками або ризикувати життям, щоб все-таки знайти клад. Близька 21 відсотка варіантів середовища є цілком несприятливими, коли ... (а які саме - це ви повинні зробити у відповідності до пункта 2 завдання лабораторної роботи 1).

Розглянемо деякі особливості роботи нашого інтелектуального агента Хоми в умовах печери Вія. Як ми вже знаємо Хома спочатку знаходиться в квадраті [1,1]. І ми знаємо, що цей квадрат для Хоми безпечний. Надалі ми побачимо, як поширюються знання Хоми по мірі того, як отримуються результати нових актів сприйняття.

Першим сприйняттям є [Ні. Ні, Ні, Ні, Ні], на підставі чого Хома може зробити висновок, що сусідні до нього квадрати є безпечними.

Завдання до лабораторної роботи 4. Побудова логічного агента для "Печери Вія".

1. Розписати середовище печери Вія у відповідності із властивостями проблемного середовища інтелектуального агента використовуючи логічні правила (див. лекція 2).

2. В яких випадках Хома не добереться до золота? Класифікуйте їх.

3. Знайти всі варіанти середовища печери Вія, коли Хома не зможе дістатися до кладу. Графічно відобразіть їх у відповідності до запропонованої вами класифікації.

4. Опишіть логічні вирази, які описують правила гри "Печера Вія". Напишіть логічні правила, які задають умови неможливості добратися Хомі до кладу.

5. Напишіть програму генерації всіх варіантів середовища, коли Хома не зможе взяти золото.

Студент повинен все оформити у вигляді звіту з лабораторної роботи. Тільки після виконання цього завдання студент отримує завдання для 2 лабораторної роботи у викладача.