водопоглинальною здатністю даного борошна. Виріб із спельтового має приємний смак і аромат.

В результаті проведених досліджень встановлено, що, хоча спельтове борошно цільнозернове є цінною функціональною сировиною, хліб з борошняної суміші зі спельтовим борошном має низьку якість, а саме менший питомій об'єм виробу та пористість м'якушки. Отже, хліб із суміші пшеничного борошна першого сорту і спельтового цільнозернового потребує подальшого застосування певних технологічних заходів для покращення якості готових виробів.

Список використаних джерел:

1. Bonafaccia G., Galli V., Francisci R., Mair V., Skrabanja V., Kreft I. Characteristics of spelt wheat products and nutritional value of spelt wheat-based bread. *Food Chemistry*. 2000. № 68. P. 437-441.

2. Дробот В.І., Михонік Л.А., Семенова А.Б. Порівняльна характеристика хімічного складу та технологічних властивостей суцільно змеленого пшеничного борошна та борошна спельти. *Хранение и переробка зерна*. 2014. № 4. С. 37–39.

3. Дробот В.І., Михонік Л.А., Семенова А.Б., Фалендиш Н.О. Борошно стародавніх пшениць, продукти переробки круп'яних культур та шроти у технології хліба: монографія. Київ: ПрофКниг, 2018. С. 5–74.

4. Жигунов Д.О., Мардар М.Р., Соц С.М., Барковська Ю.С., Г.Д. Жигунова Г.Д. Дослідження технологічних властивостей пшениці та спельти як сировини для виробництва борошна і крупи. *Наукові праці НУХТ*. 2018. Том 24. № 5. С. 15-20.

Мойсеєнко О.В.
кандидат технічних наук, доцент,
Івано-Франківський державний
технічний університет нафти і газу

МАТЕМАТИЧНЕ ПІДҐРУНТЯ ПОБУДОВИ МАРШРУТУ У НАВІГАЦІЙНОМУ МОБІЛЬНОМУ ДОДАТКУ

Припустимо, що необхідно організувати конференцію, виставку або удосконалити інфраструктуру торгового центру або аеропорту. У будь-якому випадку ми маємо справу з великим потоком відвідувачів. Одна з проблем, з якою доведеться зіткнутись – це навігація всередині

приміщень, а точніше її відсутність. Вирішивши цю проблему, можна підвищити ефективність і доступність заходу, а також поліпшити якість послуг, що надаються. Доведено, що добре продумана і зрозуміла навігація в таких приміщеннях знижує інформаційне навантаження на персонал в середньому на 20%, що сприяє більш ефективному використанню персоналу і, звичайно, дозволяє знизити напруженість, яка може виникнути в разі плутанини і неможливості персоналу оперативно вирішувати проблеми через високе навантаження.

На даний момент на ринку навігаційних сервісів пропонується безліч систем даного призначення, але немає системи, яка б визначала місце розташування пристрою і ефективно прокладала між двома довільними об'єктами всередині будівлі.

Звичайно, що побудова найкоротшого шляху починається із побудови графу. Для зберігання такого графу в пам’яті електронного пристрою використовуються матриця суміжності, або матриця інцидентності.

Прийнявши до уваги специфіку сфери використання алгоритму побудови маршрутів нами запропоновано використати матрицю інцидентності. В такій матриці відображено зв’язки між інцидентними елементами графа.

На наступному етапі були проаналізовані алгоритми пошуку найменшого шляху.

Алгоритм Дейкстри зазвичай використовується для проходу по графам, дуги яких мають різну вагу.

Алгоритм Дейкстри [1]:

– на кожному кроці обирає крок з множини відділених вершин вершину з найменшою відділю до старту і стирає ребра, що виходять з неї;

– завершує свою роботу, коли ціль досягнута (або переглянуті всі вершини).

Швидкість роботи алгоритму Дейкстри залежить від швидкості операцій з пріоритетною чергою.

Оскільки в нашому випадку ми розглядаємо мережу доріг/шляхів, то m= O(n) (граф планарний майже всюди).

Для фібоначчієвих куп час роботи алгоритму складає $O(n \cdot \log n + n)$, для двійкових куп: $O(n \log n)$.

Вдосконалений алгоритм Дейкстри передбачає роботу тільки з цілочисельними ребрами.

Цей алгоритм має кілька переваг у порівнянні з пошуком в ширину: він враховує вагу (довжину) шляху і оновлює вузли, якщо до них знайдений кращий шлях.

Алгоритм Беллмана-Форда – алгоритм пошуку найкоротшого шляху у зваженому графі. За час $O(|V| \times |E|)$ алгоритм знаходить найкоротші шляхи від однієї вершини графа до всіх інших (V – кількість вершин, E – кількість ребер). На відміну від алгоритму Дейкстри, алгоритм Беллмана-Форда допускає ребра з від'ємною вагою. Зауважимо, що найкоротших шляхів може не існувати. Так, в графі, що містить цикл з від'ємною сумарною вагою, існує як завгодно короткий шлях від однієї вершини цього циклу до іншої (кожен обхід циклу зменшує довжину шляху).

Кращий алгоритм для пошуку оптимального шляху в різних просторах є A*-алгоритм. Цей евристичний пошук перебирає всі вузли і по мірі наближення до найкращого маршруту йде через цей вузол [2]. Формула евристики :

$$f(n) = g(n) + h(n),$$

де $f(n)$ – оцінка, призначена вузлу n;

$g(n)$ – найменша вага /вартість досягнення у вузол n з початкової;

$h(n)$ – евристична наближена вартість шляху до мети від n-го вузла.

Цей алгоритм враховує довжину попереднього шляху з алгоритму Дейкстри з евристикою з алгоритму «кращий-перший». Доти поки еврістичне наближення h(n) є допустимим, він однозначно знаходить найкоротший шлях, тобто не перевищує відстані, що залишилась до кінцевої мети. Цей алгоритм найкращим чином використовує евристику: жоден алгоритм не пройде меншу кількість вузлів, не враховуючи при цьому вузли з однаковою вагою.

В реальних задачах A*-алгоритм є дуже гнучким. Оцінкою вузла часто є комірка або позиція, яку займає об'єкт. У нашому випадку це точка локації в приміщенні. Сусідні стани іноді можуть змінюватися за ситуацією (наприклад, перепланування будівлі). Суміжні стани можуть бути виключені, тому що вони непрохідні. Вартість/вага переходу з однієї позиції в іншу можуть бути різними характеристиками: звичайною віддаллю між позиціями або «вартість» за часом.

Найпростіший спосіб скорочення потреб в пам'яті для пошуку A* алгоритму полягає в застосуванні ідеї ітеративного поглиблення в контексті еврістичного пошуку. Реалізація цієї ідеї призвела до створення алгоритму A* з ітеративним поглибленням (Iterative-Deepening A* – IDA*). Основна

відмінність між алгоритмом IDA* і стандартним алгоритмом ітеративного поглиблення полягає в тому, що умовою зупинки розгортання є f-вартість (g + h), а не глибина; на кожній ітерації цим зупинним значенням є мінімальна f-вартість будь-якого вузла, що перевищує зупинне значення, досягнуте в попередній ітерації. Алгоритм IDA* є оптимальним для вирішення багатьох завдань з одиничними вартостями етапів і дозволяє уникнути істотних витрат, пов'язаних з підтримкою відсортованої черги вузлів. На жаль, цей алгоритм характеризується такими ж труднощами, пов'язаними з використанням вартостей з дійсними значеннями, як і ітеративна версія пошуку за критерієм вартості.

Нами пропонується застосувати рекурсивний пошук за першим найкращим збігом. В такому рекурсивному алгоритмі виконуються спроби імітувати роботу стандартного пошуку за першим найкращим збігом, але з використанням тільки лінійного простору. Цей алгоритм має структуру, аналогічну структурі рекурсивного пошуку в глибину, але на відміну від нескінченного проходження вниз вздовж поточного шляху даний алгоритм контролює f-значення найкращого альтернативного шляху, доступного з будь-якого «предка» поточного вузла. Якщо поточний вузол перевищує цю межу, то поточний етап рекурсії скасовується і рекурсія триває з альтернативного шляху. Після скасування даного етапу рекурсії в алгоритмі реалізується заміна f-значення кожного вузла уздовж даного шляху найкращим f-значенням його дочірнього вузла. Завдяки цьому в алгоритмі запам'ятовується f-значення найкращого листового вузла з забутого поддерева і тому в певний час після цього часу може бути прийнято рішення про те, чи варто знову розгортати це піддерево.

В змодельованій ситуації прокладання маршруту, алгоритм A* з рекурсивним пошуком є найкращим алгоритмом для визначення оптимальних шляхів у різних просторах. Він гарантовано знаходить оптимальний шлях і, з врахуванням обмеження значення ***f***, ніколи не перевищує дійсної відстані, що залишилась до цілі.

Список використаних джерел:

1. Вороновский Г. К., Махотенко К. В., Петрашев С. Н., Сергеев С. А. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. – Х.: ОСНОВА, 1997. – 112 с.

2. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ / Пер. с англ. под ред. Шеня А. – М.: МЦНМО: БИНОМ. Лаборатория знаний, 2004. – 2-е изд., стереотип. – 960 с.

3. Зак Ю. А., Математические модели и алгоритмы построения эффективных маршрутов доставки грузов, РУСАЙНС. М., 2015, 306 с.

4. Domschke W., Logistik: Transport. Grundlagen lineare Transport- und Umladeprobleme, 5 Auflage, R. Oldenburg Verlag, München–Wien, 2007, 234 pp.

5. Grünert T., Imich St., *Optimierung in Transport*, v. 1, Grundlagen, Shaker Verlag, 2005; v. 2, Wege und Touren, Shaker Verlag, 2005.

6. Bianchessi N., Righini G., «Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery», Computers & Operations Research, 34:2 (2007), 578–594.

7. Montano F. A. T., Calvao R. D., «Vehicle Routing Problem with Simultaneous Pick-ap Up and Delivery Service», Operational Research of India, 39:1 (2002), 19–33.

8. Montano F. A. T., Calvao R. D., «A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service», Computers & Operations Research, 33 (2006), 595–619.

Польшакова О.М.
старший викладач;
Юдов А.М.
студент,
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

ІНТЕГРАЦІЯ ОНЛАЙН БАЗИ ДАНИХ В СИСТЕМУ OCR

Оптичне розпізнавання символів – це дослідницька область, яка має за ціль розробку комп'ютерної системи з можливістю автоматично розпізнавання та обробки тексту із зображень.

У наші дні існує величезний попит на зберігання інформації на диску даних, що містяться у друкованих чи рукописних документах або зображеннях, щоб надалі мати змогу повторно використовувати цю інформацію за допомогою комп'ютерів. В багатьох випадках ці тексти вкрай об’ємні та потребують багато місця на пристрої де вони обробляються, що може спричинити до подальшого уповільнення роботи пристрою. Для запобігання цього, можливо реалізувати зберігання текстів у базі даних, яка буде знаходитись на іншому пристрої, відмінному від того, де відбувається розпізнання тексту.