УДК 004.9 **doi: 10.26906/SUNZ.2020.2.085**

В. В. Міхав, Є. В. Мелешко, М. С. Якименко

Центральноукраїнський національний технічний університет, Кропивницький, Україна

МЕТОД ЗБЕРІГАННЯ ДАНИХ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ НА ОСНОВІ БІНАРНИХ ДІАГРАМ РІШЕНЬ

Анотація. Стаття присвячена дослідженню методів збереження даних рекомендаційних систем. Запропоновано та досліджено використання бінарних діаграм рішень для збереження таких даних. Внаслідок великого розміру рекомендаційних систем суттєвими є обмеження по оперативній пам'яті. Метою роботи є розробка методу зберігання даних рекомендаційної системи у формі бінарних діаграм рішень та порівняння з методами збереження на основі інших структур даних. Дані рекомендаційної системи зберігаються у вигляді графу із вершинами, які представляють користувачів системи та об'єкти системи, а ребра – дії користувачів системи, відношення подібності, зв'язки рекомендацій тощо. Для підвищення ефективності у випадку інтенсивного редагування графу запропоновано збереження даних на основі "гарячого" (хеш-таблиця) та "холодного" (бінарна діаграма рішень) сховищ.Проведено серію експериментів для перевірки ефективності розробленого способу зберігання даних, для чого розроблено програмну модель спрощеної рекомендаційної системи та описано алгоритм роботи такої системи. В чисельному експерименті пропонований спосіб зберігання даних на основі бінарних дерев рішень порівнюється із трьома іншими: на основі бітових масивів, зв'язних списків та хеш-таблиць. Розглянуто переваги та недоліки реалізації кожного із вказаних методів. В ході експерименту для різних значень кількості агентів, предметів, сесій та вподобань досліджено максимальні та мінімальні значення використаної оперативної пам'яті, а також час генерації лайків, сесій та рекомендацій. Встановлено, що у випадку застосування бінарних діаграм рішень обсяг використаної оперативної пам'яті є нижчим за інші способи при меншій швидкодії, що частково може бути компенсовано декількома застосованими оптимізаціями. Завдяки меншому використанню оперативної пам'яті можна зберігати інформацію про більшу кількість вподобань, що може виявитися корисним у випадку великих розмірів графу рекомендаційної системи. Можливість для бінарних діаграм рішень пошуку даних за частковими ключами додатково дозволяє зберігати дані більшої розмірності.

Ключові слова: рекомендаційні системи, бінарні діаграми рішень, зв'язані списки, хеш-таблиці, комп'ютерне моделювання.

Вступ

На сьогоднішній день рекомендаційні системи (РС) мають широке застосування у соціальних мережах, системах Інтернет-торгівлі, поширенні медіаконтенту, реклами тощо [1, 2]. Ефективний спосіб представлення даних, необхідних для роботи такої системи, може зменшити кількість потрібних ресурсів та полегшити розробку і використання більш складних алгоритмів для формування списків рекомендацій.

В наш час існує багато різних систем управління базами даних, крім реляційних баз даних широке застосування отримують бази даних типу NoSQL [3, 4] з різними способами зберігання даних, зокрема, Сховища типу «ключ-значення» (Key-value stores), Масштабовані розподілені сховища (Column Family (Bigtable) stores), графові СУБД (Graph Stores), документо-орієнтовані СУБД (Document Stores) [3-5].

Спосіб збереження даних рекомендаційної системи є важливим з точки зору якості її роботи, швидкості, можливості масштабування, зручності виконання основних операцій з даними для формування рекомендацій.

Все частіше для зберігання даних рекомендаційних систем та інших додатків починають використовувати графові моделі [6-8], також графова форма представлення даних стає поширеною у програмному моделюванні складних систем та мереж [9-12], і це відбувається через ряд переваг графових моделей [8, 13]. Яскравим прикладом такого підходу являється побудова рекомендаційних систем з застосуванням графової СУБД Neo4j [14]. Графові моделі СУБД надають не лише зручний формат зберігання даних, а й зручний формат запитів. В документації до Neo4j є приклади реалізації алгоритмів формування рекомендацій запитами до цієї СУБД, що ілюструє її придатність для використання в РС.

Існують різні способи представлення графів в комп'ютерних системах, зокрема, таблиці суміжності, списки ребер, матриці інцидентності тощо [15]. Також графи можна зберігати у вигляді бінарних діаграм рішень, які по суті являються економною формою представлення булевих функцій у вигляді орієнтованого ациклічного графа [16-18].

На даний момент практично не досліджено можливість використовувати бінарні діаграми рішень для зберігання даних рекомендаційної системи.

Метою даної роботи була розробка методу зберігання даних рекомендаційної системи у формі бінарних діаграм рішень, а також дослідження ефективності даного методу шляхом експериментів на програмній моделі рекомендаційної системи.

Основна частина

Дані рекомендаційної системи зручно представляти у вигляді графу. Один з найбільш поширених способів зберігання графів у комп'ютерних програмах – використання матриці суміжності [15]. Матриця суміжності графу A містить значення булевого типу, тобто вона є матричним представленням булевої функції F, яка описує наявність дуги між двома вузлами. Якщо орграф G помічений, то необхідно також зберегти його вагу.

Представлення булевої функції як таблиці істинності чи досконалої кон'юнктивної/диз'юнктивної нормальної форми потребує $\Omega(2^{2m+l})$ пам'яті та вимагає значних обчислень для визначення значення фун-

кції. Тому у даній роботі пропонується дослідити використання бінарних діаграм рішень для зберігання графу даних рекомендаційної системи.

Бінарні діаграми рішень (БДР) – це економна форма представлення булевих функцій у вигляді орієнтованого ациклічного графу (рис. 1). Вершини графу представляють аргументи функції, листки – її двійкові значення [16-18]. Для додавання і вилучення ребер та зміни ваги ребер необхідно мати можливість редагувати дані графу. БДР дають можливість зберігати дані у стисненому вигляді та швидко отримувати значення функції за її параметрами, але редагування БДР вимагає складних обчислень. При представленні булевих функцій у формі БДР стало можливим розв'язувати багато проблем, які при традиційних представленнях структур нерозв'язні через значну розмірність таких представлень і складність операцій над ними. БДР можуть успішно застосовуватися фактично в кожній галузі, де потрібно обробляти дискретні структури даних.

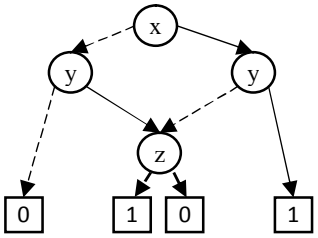

Рис. 1. Приклад бінарної діаграми рішень

Адаптація БДР для представлення орієнтованих графів з даними рекомендаціійної системи. Коли дані рекомендаційної системи зберігаються у вигляді графу, то як правило вони мають такий формат:

– вершини графу – користувачі системи, об'єкти системи (контент, товари, тощо).

– ребра графу – дії користувачів по відношенню до об'єктів (перегляди, оцінки, тощо), відношення подоби, зв'язки рекомендацій типу користувачу-рекомендовано-об'єкт, тощо.

Була досліджена можливість зберігати граф рекомендаційної системи у пам'яті у вигляді бінарної діаграми рішень. Для проведення експерименту по вивченню потреб в оперативній пам'яті було розроблено програмну модель спрощеної рекомендаційної системи, в якій було виділено три основні сутності – агент, сесія та предмет.

Рекомендаційна система отримує такі параметри: n_a – кількість агентів, n_s – кількість сесій, n_i – кількість предметів, n_{al} – максимальна кількість вподобань агента, n_{sl} – максимальний розмір сесії.

РС у розробленій програмній моделі працює за таким алгоритмом:

1. Для кожного з n_a агентів випадковим чином генерується від 1 до n_{al} вподобань (рис. 2). При цьому унікальність вподобань не перевіряється, тому реальна кількість вподобань може виявитися менше.

2. Створюється n_s сесій. До кожної сесії закріплюється випадковим чином обраний агент. Потім серед вподобань цього агента випадковим чином обирається від 1 до $min(n_{al}, n_{sl})$ вподобань, які копіюються до сесії (рис. 3).

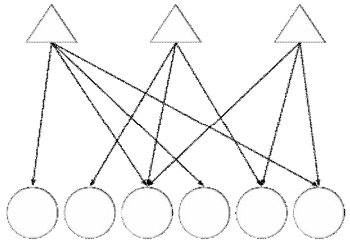

Рис. 2. Створення вподобань для агентів (трикутники – агенти, круги – предмети)

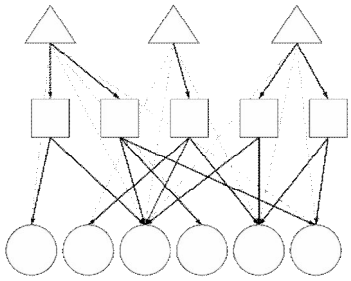

Рис. 3. Створення сесій та вподобань для них (трикутники – агенти, квадрати – сесії, круги – предмети; сесія містить вподобання агента, які він зробив за одне відвідування ресурсу)

3. Випадковим чином обирається контрольна сесія, для якої буде сформовано рекомендацію.

4. Визначаються усі предмети, які належать до контрольної сесії (рис. 4).

5. Здійснюється пошук усіх сесій, вподобання яких мають перетин із вподобаннями контрольної сесії (рис. 5). На цьому етапі є можливість відфільтрувати сесії за розміром перетину.

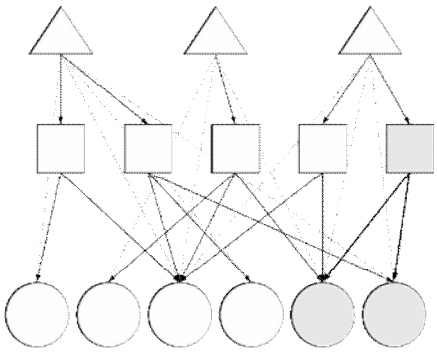

Рис. 4. Визначення контрольної сесії (трикутники – агенти, квадрати – сесії, круги – предмети)

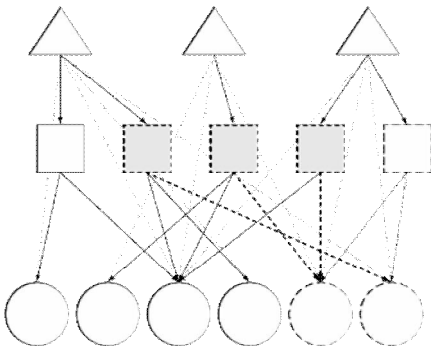

Рис. 5. Пошук сесій, які мають перетин із контрольною сесією (трикутники – агенти, квадрати – сесії, круги – предмети)

6. Визначаються предмети, які буде рекомендовано. Здійснюється пошук усіх предметів, які належать хоча б одній з відібраних сесій, але не належать до контрольної сесії (рис. 6). На цьому етапі є можливість відфільтрувати предмети за кількістю закріплених сесій.

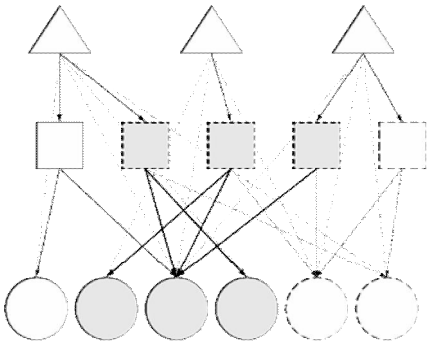

Рис. 6. Вибірка предметів для формування рекомендації (трикутники – агенти, квадрати – сесії, круги – предмети)

Накопичення змін до БДР. Якщо алгоритм передбачає періоди інтенсивного редагування графу, можна тимчасово зберігати зміни у структурі даних, більш пристосованій до внесення змін. Можна застосувати ідею "гарячого" (хеш-таблиця) та "холодного" (БДР) сховищ. У хеш-таблицю заносяться параметри функції, на яких необхідно змінити значення, та саме значення. При досягненні певного розміру або по завершенню логічного блоку операцій із хеш-таблиці формується коригуюча БДР, яка потім зливається із основною. При читанні значень необхідно спочатку перевірити існування значення у хеш-таблиці, і у випадку його відсутності читати із БДР. Складність використання цього підходу полягає у виборі булевої операції, яка буде використовуватися при злитті БДР. Одним з найкращих підходів є використання операції XOR стає, особливо коли необхідно зберігати інформацію не лише про факт наявності дуги між вузлами, а й зберігати вагу цієї дуги. Для цього можна виділити додаткову групу змінних, яка відповідатиме за представлення ваг у БДР.

Було проведено серію експериментів для перевірки ефективності запропонованого способу зберігання даних на основі бінарних діаграм рішень, та порівняння його з іншими способами зберігання даних в інших структурах даних.

У проведених експериментах було здійснено порівняння 4-х різних реалізацій – на основі бінарних діаграм рішень, бітових масивів, зв'язних списків та хеш-таблиць. Кожна реалізація дотримується описаного вище алгоритму програмного моделювання рекомендаційної системи, але має певні відмінності.

Реалізація на основі бітових масивів найпростіша, оскільки у цьому випадку є можливість отримати прямий доступ до значення за номером агента/сесії та предмету, але це вимагає значних витрат пам'яті.

Інші структури даних більш складні, тому для прискорення їх роботи для збереження інформації про перетин із контрольною сесією було застосовано хеш-таблиці. Також вони не дають можливості вибирати випадковий елемент, тому на етапі створення сесій відбувається повний перебір вподобань агента, доки вони не закінчяться або доки не буде відібрано необхідну кількість вподобань. Для кожного вподобання агента обирається випадкове число, на основі якого визначається, чи заносити вподобання до сесії.

Зв'язні списки дозволяють зберігати інформацію лише про існуючі вподобання, завдяки чому вони використовують менше пам'яті ніж бітові масиви. Проте через те, що вони не дають доступу до елементів у випадковому прядку, пошук перетину двох сесій займає багато часу.

Хеш-таблиці дозволяють отримати доступ до елементу за ключем, завдяки чому можна покращити результати, отримані за допомогою зв'язних списків. Якщо у випадку зв'язних списків складність пошуку перетину квадратична, то у випадку хеш-таблиці вона лінійна.

Дані рекомендаційної системи під час експериментів зберігалися в оперативній пам'яті з можливістю зберігання у текстових файлах. Для експериментів використовувався персональний комп'ютер з 24 Гб оперативної пам'яті. Прочерки у комірках показують, що не можливо було отримати результат через нестачу оперативної пам'яті.

Результати експериментів наведені у табл. 1, де:

– agentCount – кількість агентів;

– itemsCount – кількість предметів;

– sessionsCount – кількість сесій;

– sessionsSize – розмір сесії;

– maxLikes – максимальна кількість лайків.

Бінарні діаграми рішень мають як переваги, так і недоліки у порівнянні з хеш-таблицями, бітовими масивами та зв'язними списками. БДР дають можливість шукати дані за частковими ключами, що додатково дозволяє зберігати зв'язки більшої розмірності. Бінарні діаграми даних зберігають дані з меншими витратами пам'яті, ніж хеш-таблиці, завдяки чому у оперативній пам'яті одночасно можна розмістити інформацію про більшу кількість вподобань. БДР програють у швидкодії, проте нами було застосовано кілька оптимізацій:

– підтримка унікальності фрагментів БДР обчислюувально дуже дорога, тому вона була відділена від процедури редагування і виконується кілька разів за час генерації тестових даних;

– видалення надлишкових фрагментів з БДР спричиняє розрідженість вузлів у пам'яті, через що зростає час доступу до вузла. Тому після видалення надлишкових вузлів вони ущільняються шляхом перенесення у нову область пам'яті;

– ущільнення вузлів дає можливість звільнити певну кількість пам'яті, проте виявилося, що звільнення пам'яті у цій ситуації неефективне – наступні операції редагування вимагають створення нових вузлів, через що потрібно буде повторно переносити вузли між областями пам'яті;

– для здійснення пошуку у БДР її необхідно проіндексувати відповідно до ключа та параметрів пошуку, а обхід усіх вузлів БДР – це дорога операція. Для прискорення пошуку було покращено процедуру індексування, завдяки чому пошук з вказаною першою частиною ключа не вимагає повторної індексації.

Таблиця 1 - **Результати експериментів**

Структура даних	Використана пам'ять, мін. знач., Gb	Використана пам'ять, макс. знач., Gb	Час генерації лайків, ms	Час генерації сесій, ms	Час генерації рекомендацій, ms
agentCount = 4096, itemsCount = 8192, sessionsCount = 16384, sessionSize = 64, maxLikes = 640					
БДР	0,024	0,037	2562	2209	411
Хеш-таблиця	0,106	0,106	76	103	25
Бітовий масив	0,166	0,166	12	127	111
Зв'язний список	0,108	0,108	78	120	247
agentCount = 16384, itemsCount = 32768, sessionsCount = 65536, sessionSize = 128, maxLikes = 1024					
БДР	0,17	0,29	31408	34551	5240
Хеш-таблиця	0,46	0,46	369	702	179
Бітовий масив	2,5	2,5	179	1327	1623
Зв'язний список	0,79	0,79	516	721	2823
agentCount = 32768, itemsCount = 65536, sessionsCount = 131072, sessionSize = 192, maxLikes = 1536					
БДР	0,452	0,733	95075	106675	23327
Хеш-таблиця	1,1	1,1	1168	2013	640
Бітовий масив	-	-	-	-	-
Зв'язний список	2,2	2,2	1499	2112	9973
agentCount = 65536, itemsCount = 131072, sessionsCount = 262144, sessionSize = 192, maxLikes = 1536					
БДР	0,786	1,3	210945	237214	45766
Хеш-таблиця	2,2	2,2	2312	4034	316
Бітовий масив	-	-	-	-	-
Зв'язний список	4,5	4,5	2989	4323	23223
agentCount = 131072, itemsCount = 262144, sessionsCount = 524288, sessionSize = 256, maxLikes = 2048					
БДР	1,8	3,6	607610	682754	153045
Хеш-таблиця	6,4	6,4	6082	12327	3954
Бітовий масив	-	-	-	-	-
Зв'язний список	12,1	12,1	8052	11195	74346
agentCount = 262144, itemsCount = 524288, sessionsCount = 1048576, sessionSize = 256, maxLikes = 2048					
БДР	3,4	7,2	1366467	1551391	313122
Хеш-таблиця	12,8	12,8	12268	24666	6221
Бітовий масив	-	-	-	-	-
Зв'язний список	24	24	16051	22323	146845
agentCount = 524288, itemsCount = 1048576, *sessionsCount* = 2097152, sessionSize = 256, maxLikes = 2048					
БДР	6,5	14,4	3170877	3623865	643864
Хеш-таблиця	25,5	25,5	24204	116728	44527
Бітовий масив	-	-	-	-	-
Зв'язний список	-	-	-	-	-

Висновки

У роботі розроблено метод зберігання даних рекомендаційної системи на основі бінарних діаграм рішень. Також запропоновано використання двох видів сховищ даних (гарячого та холодного) для підвищення ефективності роботи системи і описано спосіб їх злиття. Розроблено програмну модель спрощеної рекомендаційної системи та описано алгоритм роботи такої системи. На основі розробленої програмної моделі проведено експерименти для перевірки працездатності та ефективності розробленого методу зберігання даних РС на основі БДР та для порівняння його з іншими методами зберігання даних.

Результати експерименту показали, що у випадку застосування бінарних діаграм рішень оперативна па-

м'ять використовуються більш економно в порівнянні із хеш-таблицями, бітовими масивами та зв'язними списками. Для компенсації меншої швидкості роботи з даними в оперативній пам'яті при використанні БДР запропоновано декілька оптимізацій. Завдяки меншому використанню оперативної пам'яті можна зберігати інформацію про більшу кількість вподобань користувачів рекомендаційної системи при виконанні обчислень для формування рекомендацій. Даний факт є суттєвим у випадку великих розмірів графів, що характерно для реальних рекомендаційних систем. Крім того, можливість пошуку даних за частковими ключами в бінарних діаграмах рішень дозволяє додатково збільшити розмірність даних, що зберігаються.

СПИСОК ЛІТЕРАТУРИ

1. "Recommender Systems Handbook" (2010) Editors F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, New York, NY, USA: Springer-Verlag New York, Inc., 842 p.
2. Jones M. (2013) "Recommender systems, Part 1. Introduction to approaches and algorithms. Learn about the concepts that underlie web recommendation engines", URL: https://www.ibm.com/developerworks/opensource/library/os-recommender1/index.html?s_tact=105agx99&s_cmp=cp
3. Фаулер М., Садаладж П. Дж. (2013) "NoSQL: новая методология разработки нереляционных баз данных", Издательский дом «Вильямс», Москва, 192 с.
4. Meier A., Kaufmann M. (2019) "SQL & NoSQL Databases", Springer Vieweg, Wiesbaden, P. 201-218. – URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.468.7089&rep=rep1&type=pdf
5. Cure O., Blin G (2014) "RDF Database Systems: Triples Storage and SPARQL Query Processing", Elsevier Science, 256 p.
6. Yi N., Li C., Feng X., Shi M. (2017) "Design and implementation of movie recommender system based on graph database", 14th Web Information Systems and Applications Conference (WISA), IEEE, P. 132-135.
7. Angles R. (2012) "A comparison of current graph database models", IEEE 28th International Conference on Data Engineering Workshops, IEEE, 2012, P. 171-177.
8. Засядко Г.Е., Карпов А.В. (2017) "Проблемы разработки графовых баз данных", // ИВД. №1 (44), URL: https://cyberleninka.ru/article/n/problemy-razrabotki-grafovyh-baz-dannyh
9. Мелков С., Мусатов Д., Савватеев А. (2013), "Моделирование социальных сетей", URL: https://kpfu.ru/docs/F117464271/MMS_socnet_cities.pdf
10. Берновски М.М., Кузюрин Н.Н. (2012) "Случайные графы, модели и генераторы безмасштабных графов" // Труды ИСП РАН. URL: https://cyberleninka.ru/article/n/ sluchaynye-grafy-modeli-i-generatory-bezmasshtabnyh-grafov
11. Райгородский А.М. (2012) "Математические модели Интернета", "Квант" №4, С. 12-16, URL: https://elementy.ru/nauchno-populyarnaya_biblioteka/431792
12. Meleshko Ye. (2019) "Computer model of virtual social network with recommendation system", Scientific journal Innovative Technologies and Scientific Solutions for Industries, Kharkiv: NURE, Issue. 2 (8), P. 80-84
13. Робинсон Я., Вебер Д., Эифрем Э. (2016) "Графовые базы данных: новые возможности для работы со связанными данными", М.: ДМК Пресс, 256 с.
14. "Neo4j Documentation" (2020) URL: https://neo4j.com/docs
15. Храбров Д. (2006) "Способы хранения графов в памяти компьютера", URL: https://dexp.in/russian/graph-storage/
16. Кнут Д.Э. (2013) "Искусство программирования, том 4А. Комбинаторные алгоритмы. Часть 1", М.: "Вильямс", 960 с.
17. Minato S. (2001) "Zero-suppressed BDDs and their applications", International Journal on Software Tools for Technology Transfer, №3.2 – pp. 156– 170.
18. Міхав В.В. (2019) "Програмне забезпечення для моделювання мереж репутації користувачів соціальних веб-ресурсів на основі бінарних діаграм рішень", Системи управління, навігації та зв'язку, Т.5., №57, С. 78-83. – URL: http://journals.nupp.edu.ua/sunz/article/download/1720/1408

Received (Надійшла) 25.03.2020
Accepted for publication (Прийнята до друку) 06.05.2020

Method for storing data of a recommender system based on binary decision diagrams

V. Mikhav, Ye. Meleshko, M. Yakymenko

Abstract. The paper is devoted to researching of methods for storing data of recommender systems. Usage of binary decision diagrams for saving such data is proposed and studied. Due to the large size of recommender systems, there are significant limitations on RAM. The purpose of the work is to develop a method of storing data of a recommender system as binary decision diagrams and to compare it with storing methods based on other data structures. The data of a recommender system is stored as a graph with vertices representing users and items of the system, and the edges representing the actions users, similarity relations, relationships of recommendations, et cetera. To increase efficiency in the case of intensive graph editing, data storing based on "hot" (hash table) and "cold" (binary decision diagram) storages is proposed. A series of experiments was made to test the efficiency of the developed method of data storing, for this a software model of a simplified recommender system was developed and the work algorithm of such a system was described. In the numerical experiment the proposed method of storing data based on binary decision trees is compared with three others: based on bit maps, linked lists and hash tables. The advantages and disadvantages of implementing for each of these methods are considered. During the experiment, for various values of the number of agents, items, sessions and preferences, it were investigated the maximum and minimum values of the used RAM, along with time for generating of likes, sessions and recommendations. It has been found that in the case of binary decision diagrams, the amount of used RAM is lower than other methods but at lower speeds, the latter can be partially compensated by several applied optimizations. Due to the less usage of RAM, it is possible to store information about a larger amount of preferences, it may be useful in the cases of large size of the recommender system graph. The ability for binary decision diagrams of data search by partial keys additionally allows to store larger data.

Keywords: recommender systems, binary solution diagrams, linked lists, hash tables, computer simulation.