ЛЕКЦІЯ 1. БУЛЕВИЙ ПОШУК

Глибовець А.М.

INFORMATION RETRIEVAL (IR)

- Інформаційний пошук – що це таке?
- Інформаційний пошук (IR) – це процес пошуку в великій колекції (що зберігається як правило, в пам’яті комп'ютера) деякого неструктурованого матеріла (зазвичай – документу), що задовольняє інформаційній потребі.

INFORMATION RETRIEVAL (IR)

- Додаткові задачі IR :
  - навігація по колекції документів
  - фільтрація документів
  - кластеризація
  - класифікація

INFORMATION RETRIEVAL (IR)

- Класифікація систем інформаційного пошуку за масштабом:
  - WEB – пошук
  - Системи корпоративного, відомчого і орієнтованого на предметну область пошуку
  - Персональний інформаційний пошук

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Як приклад візьмемо зібрання творів Шекспіра.
- Припустимо, ми хочемо визначити в якому творі використовуються слова Brutus AND Caesar AND NOT Calpurnia.
- Як ми можемо здійснити такий пошук?

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Самий простий спосіб – «в лоб», послідовний перегляд всіх документів (linear scanning)
- Часто називають – прямим пошуком або англійською мовою grepping.
- Доволі часто така обробка доповнюється пошуком по шаблону з джокерами (wildcard pattern matching)
- На сучасних комп'ютерах виконання простих запитів на колекціях даних середніх розмірів (загальний обсяг зібрань творів Шекспіра трохи менше одного мілліонна слів) відбудеться за прийнятний час для пересічного користувача.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Але дуже часто необхідно дещо більше:
  - Інколи необхідно швидко обробити велику колекцію документів (мілліарди і триліони слів)
  - Інколи необхідна більша гнучкість при виконанні операції порівняння (Romans NEAR countrymen, може означати «не далі ніж 5 слів», або «в межах одного речення»)
  - Інколи необхідно виконати ранжований пошук

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Для того, що б уникнути послідовного перебору текстів при виконанні кожного запиту, необхідно завчасно скласти *індекс документів*.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ МАТРИЦЯ ІНЦИДЕНТНОСТІ «ТЕРМІН-ДОКУМЕНТ»

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	…
Antony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
…							

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Для обробки запиту «Brutus AND Caesar AND NOT Calpurnia» ми беремо вектор для термінів Brutus, Caesar і Calpurnia, обраховуємо двійкове доповнення до останнього вектору і виконуємо порозрядну операцію AND.
- 110100 AND 110111 AND 101111 = 100100
- Результат запиту: Antony and Cleopatra і Hamlet.

МОДЕЛЬ БУЛЕВОГО ПОШУКУ

- Модель булевого пошуку – це модель інформаційного пошуку, в ході якої можна оброблювати будь-який запит, що має вигляд булевого виразу, тобто виразу, в якого терміни використовуються в поєднанні з операціями AND, OR і NOT.
- В рамках цієї моделі документ розглядається просто як множина слів.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Розглянемо більш реальний приклад.
- Ми маємо N=1 мільйон документів.
- Документ – будь, який об’єкт, на основі яких вирішено будувати систему інформаційного пошуку.
- Групу документів по яким здійснюється пошук, називають колекцією (collection). Інколи називають корпусом (corpus) або масивом текстів (body of texts).
- Припустимо, що кожний документ складається приблизно з 1000 слів (2-3 книжкові сторінки).
- Якщо припустити, що в середньому на зберігання слова (з урахуванням пробілів і знаків пунктуації) відводиться 6 байт, то така колекція займе коло 6 Гбайт (GB).
- Як правило, в таких документах може міститися близько M = 500000 різних термінів.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Наша мета – розробити систему, що виконує пошук за довільним запитом (ad hoc retrieval).
- Мета системи – знайти в колекції документи, які являються найбільш релевантними, по відношенню до довільних інформаційних потреб, що передаються системі за допомогою одноразових, ініційованих користувачем запитів.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Тепер ми вже не можемо скласти матрицю «термін-документ» наївним чином.
- Матриця розміром 500К x 1М містить пів трильйона нулів і одиничок – надто багато.
- Але ви ж розумієте, що ця матриця надто розріджена.
- Кожний документ складається приблизно з 1000 слів, матриця буде містити не більше 1 мільярда одиничок, тому як мінімум 99,8% комірок будуть містити 0.
- НАБАГАТО КРАЩЕ ЗБЕРІГАТИ ЛИШЕ ОДИНИЧКИ.

ПРИКЛАД ІНФОРМАЦІЙНОГО ПОШУКУ

- Ця ідея є основною по відношенню до першої важливої концепції інформаційного пошуку – інвертований індекс.

ІНВЕРТОВАНИЙ ІНДЕКС (ІНВЕРТОВАНИЙ ФАЙЛ)

- Основна ідея:
  - Спочатку ми записуємо в пам’ять словник термінів (vocabulary або lexicon).
  - Для кожного терміну ми створюємо список, в якому вказані документи, що містять даний термін.
    - Кожний елемент списку містить інформацію про те, що термін знаходиться в документі, а також (доволі часто) координату терміна в документі (posting).
    - Список називається – списком словопозицій (posting list) або інвертованим списком.
  - Словник впорядковується за алфавітом.
  - Кожний список словопозицій впорядковується за ідентифікаторами документів.

ПЕРША СПРОБА СТВОРИТИ ІНВЕРТОВАНИЙ ІНДЕКС

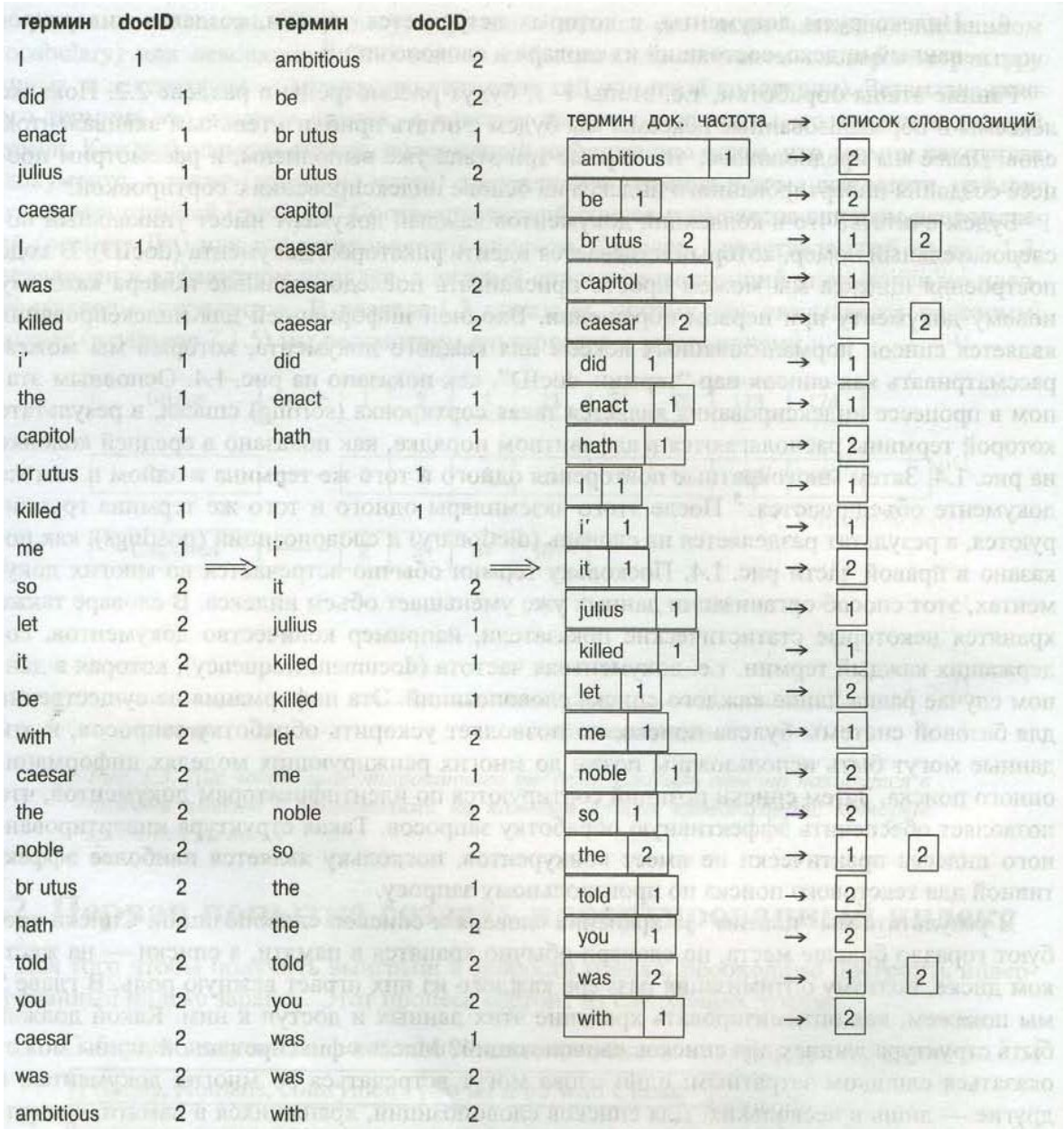
- Для того, що б отримати виграш в швидкості необхідно побудувати інвертований індекс завчасно.
- Процес складається з наступних етапів:
  - Збираємо документи, що підлягають індексації
  - Робимо розмітку тексту, перетворюючи кожний документ в список лексем (tokens)
  - Проводимо попередню лінгвістичну обробку, створюємо список нормалізованих лексем, що являють собою терміни, що індексуються
  - Індексуємо документи, в яких зустрічається термін, створюючи інвертований індекс
- Ранні етапи обробки 1-3 будуть розглянуті в наступній лекції, тому поки, що на них не зупиняємося. Поки, що лексеми і нормалізовані лексеми, будемо вважати майже еквівалентами слів.

ПЕРША СПРОБА СТВОРИТИ ІНВЕРТОВАНИЙ ІНДЕКС

- Уявимо, що перших три етапи виконані.
- Розглянемо процес створення інвертованого індексу на основі індексування з сортуванням.
- Будемо вважати, що в колекції кожен документ має унікальний номер, який часто називається ідентифікатором документа (docID).
- При побудові індексу ми можемо надавати новий номер кожному новому документу послідовно.

ПЕРША СПРОБА СТВОРИТИ ІНВЕРТОВАНИЙ ІНДЕКС

- Вихідною інформацією для індексування є список нормалізованих лексем для кожного документу, який ми можемо розглядати як список пар «термін – docID».
- Основним етапом в процесі індексування є таке сортування списку, в результаті якої терміни розташовуються в алфавітному порядку. Після чого, багаторазові повторення терміну в одному документі об’єднуються.
- Після чого екземпляри одного і того ж терміну групуються, а результат розділяється на словник і словопозиції.
- Після чого списки позицій сортуються по ідентифікаторам документів.


термин docID
I 1
did 1
enact 1
julius 1
caesar 1
I 1
was 1
killed 1
i' 1
the 1
capitol 1
br utus 1
killed 1
me 1
so 2
let 2
it 2
be 2
with 2
caesar 2
the 2
noble 2
br utus 2
hath 2
told 2
you 2
caesar 2
was 2
ambitious 2
термин docID
ambitious 2
be 2
br utus 1
br utus 2
capitol 1
caesar 1
caesar 2
caesar 2
did 1
enact 1
hath 1
I 1
I 1
i' 1
it 2
julius 1
killed 1
killed 1
let 2
me 1
noble 2
so 2
the 1
the 2
told 2
you 2
was 1
was 2
with 2
термин док. частота → список словопозиций
ambitious 1 → 2
be 1 → 2
br utus 2 → 1 2
capitol 1 → 1
caesar 2 → 1 2
did 1 → 1
enact 1 → 1
hath 1 → 2
I 1 → 1
i' 1 → 1
it 1 → 2
julius 1 → 1
killed 1 → 1
let 1 → 2
me 1 → 1
noble 1 → 2
so 1 → 2
the 2 → 1 2
told 1 → 2
you 1 → 2
was 2 → 1 2
with 1 → 2

ПЕРША СПРОБА СТВОРИТИ ІНВЕРТОВАНИЙ ІНДЕКС

- Така структура інвертованого індексу майже не має конкурентів, оскільки є найбільш ефективною для текстового пошуку по довільному запиту.

ПЕРША СПРОБА СТВОРИТИ ІНВЕРТОВАНИЙ ІНДЕКС

- В результаті ми «платимо» за зберігання словника і списків словопозицій.
- Списки вимагають багато місця тому зазвичай зберігаються на жорсткому диску, а словники в пам’яті.
- Тому тут дуже важлива оптимізація. Ми спробуємо поговорити про це пізніше.

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Як відбувається обробка запитів за допомогою інвертованого індексу і базової моделі булевого пошуку?
- Розглянемо обробку простого кон'юнктивного запиту (simple conjunctive query)
  - Brutus and Calpumia
- по інвертованому індексу.

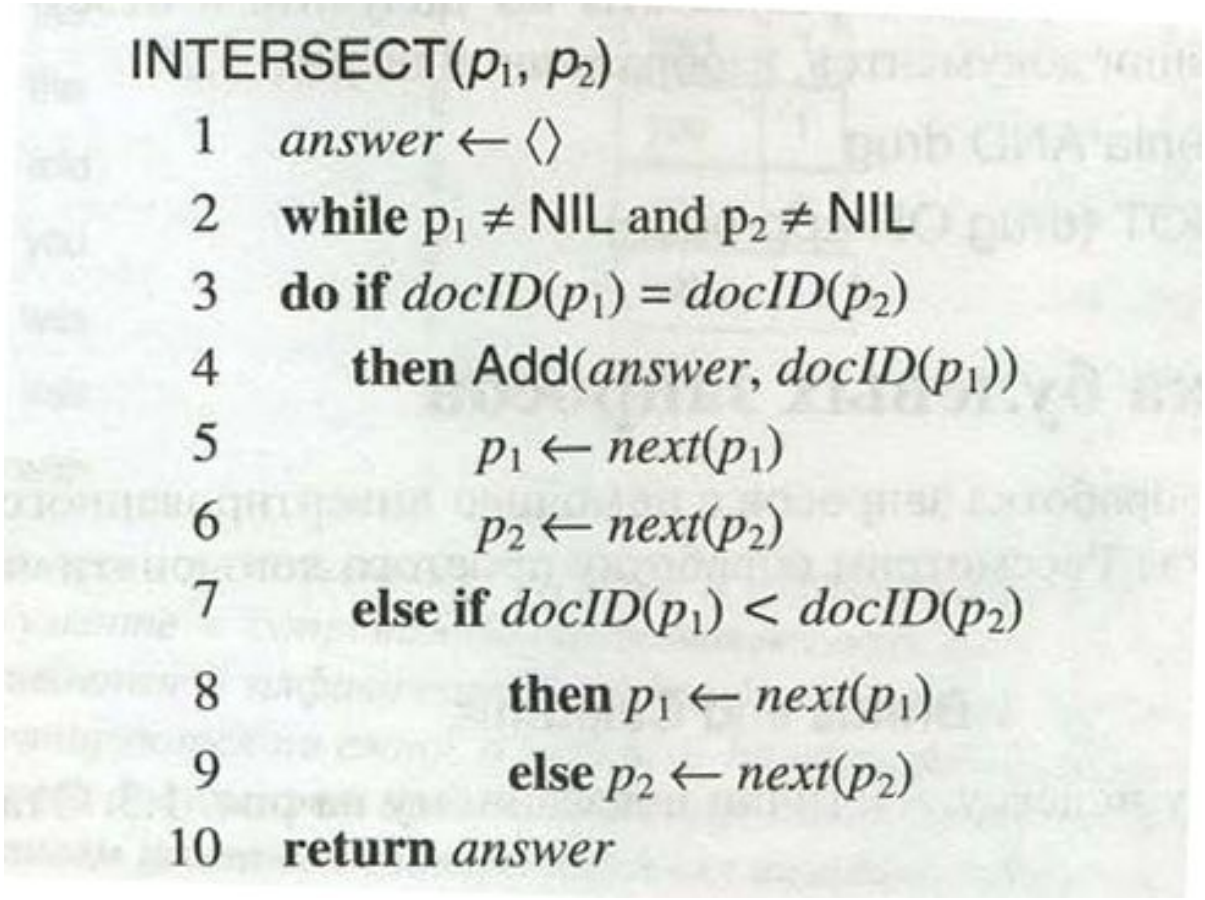
ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Ця обробка зводиться до наступного.
  - 1. Виявляємо термін Brutus в словнику.
  - 2. Знаходимо список його словопозицій.
  - 3. Виявляємо термін Calpumia в словнику.
  - 4. Знаходимо список його словопозицій.
  - 5. Знаходимо перетин цих двох списків.

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Операція перетину (intersection) є надзвичайно важливою.
- Існує простий і ефективний метод перетину списку словопозицій за допомогою алгоритму злиття.

ОБРОБКА БУЛЕВИХ ЗАПИТІВ



```
INTERSECT(p1, p2)
 1  answer ← ⟨⟩
 2  while p1 ≠ NIL and p2 ≠ NIL
 3  do if docID(p1) = docID(p2)
 4        then Add(answer, docID(p1))
 5             p1 ← next(p1)
 6             p2 ← next(p2)
 7        else if docID(p1) < docID(p2)
 8             then p1 ← next(p1)
 9             else p2 ← next(p2)
10  return answer
```

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Цей підхід можна узагальнити для обробки і більш складних запитів
- (Brutus OR Caesar) AND NOT Calpurnia

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Оптимізація запиту - це вибір такого способу організації обробки запиту, щоб можна було мінімізувати загальний обсяг роботи, яку повинна виконати система.
- Основним фактором, що впливає на ефективність обробки булевих запитів є порядок, в якому здійснюється доступ до списків словопозицій.
- Який найкращий порядок обробки запиту?

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Розглянемо запит, що складається з t термінів, об'єднаних операцією AND. Наприклад:
  - Brutus AND Caesar AND Calpumia
- Для кожного з t термінів необхідно спочатку знайти списки словопозіцій, а потім застосувати до них операцію AND.
- На практиці використовується стандартний евристичний прийом, який полягає в тому, що терміни обробляються в порядку зростання частоти документів.
- Якщо почати з перетину двох найменших списків словопозицій то всі проміжні результати не повинні перевищувати найменшого списку словопозицій, а значить, ми виконаємо при цьому найменший обсяг роботи.
- Це перший аргумент на користь того, щоб підраховувати частоту термінів у словнику це дозволяє ухвалити рішення про впорядкування на основі даних, що зберігаються в пам'яті комп'ютера, не вимагаючи доступу до списків словопозицій.

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Розглянемо тепер оптимізацію запитів більш загального вигляду.
- (Madding AND crowd) AND (ignoble OR strife) AND (killed OR slain)
- Як і раніше, визначимо частоту кожного терміна, а потім отримаємо оцінки зверху
- розміру для кожного оператора OR підсумовуючи частоти його операндів.
- Після цього запит можна обробляти в порядку зростання розміру кожного диз'юнктного терміна.
- Для довільних булевих запитів ми повинні знайти і тимчасово зберегти відповіді для проміжних виразів, що входять у більш складні висловлювання.

ОБРОБКА БУЛЕВИХ ЗАПИТІВ

- Однак у багатьох простих ситуаціях запити є виключно кон'юнкція термінів.
- Це пояснюється або природою мови запитів, або широкою популярністю такої форми запитів серед користувачів.
- В цьому випадку замість перегляду об'єднаних списків словопозицій за допомогою функції, що має два вхідних аргументу і окремо повертається значення, ефективніше перетнути всі результуючі списки словопозіцій з поточним проміжним результатом у пам'яті.
- При цьому проміжний результат ініціалізується шляхом завантаження списку словопозіцій самого рідко зустрічаємого терміну.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Альтернативою моделі булевого пошуку є моделі пошуку з ранжируванням (ranked retrieval models), наприклад модель векторного простору в рамках яких користувачі в основному застосовують вільні текстові запити (free text queries), тобто набирають одне або кілька слів, а не використовують суворі мовні конструкції з операторами; система ж сама вирішує, які документи краще за інші задовольняють цим запитам.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Незважаючи на десятиліття академічних досліджень переваг ранжованого пошуку, системи, засновані на моделі булевого пошуку, до початку 1990-х років (приблизна дата появи мережі World Wide Web), 30 років залишалися основним або навіть єдиним засобом пошуку у великих комерційних інформаційних службах.
- Однак ці системи використовували не тільки логічні операції (AND, OR і NOT), які ми розглядали до цього часу.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Суворі логічні вирази з термінами з неупорядкованими результатами накладають занадто багато обмежень при пошуку інформації і не дозволяють повністю задовольнити інформаційні запити користувачів, тому в цих системах були реалізовані розширені моделі булевого пошуку з додатковими операторами, такими як оператори близькості термінів.
- Оператор розрахунку відстані між словами запиту (proximity operator) дозволяє вказати, наскільки близько один до одного повинні бути розташовані терміни запиту в документі, причому близькість між термінами може вимірюватися кількістю слів між ними, або ж задаватися структурними одиницями, наприклад реченнями або абзацами.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Приклад Комерційна служба булевого пошуку: Westlaw.
- Westlaw (http://www.westlaw.com/) - найбільша юридична пошукова служба (по кількості передплатників).
- Щоденно півмільйона її передплатників виконують мільйони пошукових запитів в десятках терабайтів текстових даних.
- Вона заснована в 1975 році.
- У 2005 році булевий пошук (який в системі Westlaw називається Terms and Connectors) залишався основним механізмом пошуку і використовувався великою кількістю користувачів, хоча ще в 1992 році у них з'явилася можливість представляти вільні текстові запити (які в системі Westlaw називаються Natural Language).

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Розглянемо кілька прикладів булевих запитів в системі Westlaw.
- Інформаційні потреби: інформація про юридичних теорії, пов'язані із запобіганням розкритю комерційної таємниці звільненими працівниками, які перейшли на службу в конкуруючі компанії.
  - Запит: "trade secret" /s disclos! /s prevent /s employee!
- Інформаційні потреби: вимоги людей з обмеженими можливостями, що стосуються доступу до робочого місця.
  - Запит: disab! /p access! /s work-site work-place (employment / 3 place)
- Інформаційні потреби: справи, що стосуються відповідальності господарів за поведінку п'яних гостей.
  - Запит: host! /p (responsib! liab!) /p (intoxicat! drunk!) /p guest

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Зверніть увагу на довгий і точний запит і використання операторів близькості не характерних для вебу.
- У середньому запити складаються з десяти слів.
- На відміну від правил, встановлених в Інтернет, пробіли між словами є диз'юнкцію, символ & означає оператор AND, а символи / s, / p і / к встановлюють пошук збігів в одному і тому ж реченні, в абзаці або в околиці k слів відповідно.
- Подвійні лапки означають фразовий пошук (phrase search), тобто пошук послідовних слів.
- Знак оклику (!) являє собою замикаючий джокер; таким чином, слово liab! відповідає всім словами, що починаються літерами liab.
- Слова work-site відповідають будь-якому варіанту: worksite, work-site і work site.
- Типовий запит експерта зазвичай ретельно формулюється і поступово уточнюється, поки не призведе до бажаного результату.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Багато користувачів, особливо професіонали, надають перевагу булевому пошуку.
- Такі запити відрізняються точністю: документ відповідає запиту або ні.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Ми розглянули структуру і процес створення базового інвертованого індексу, включаючи словник і списки словопозицій.
- Ми ввели модель булевого пошуку
- Далі ми більш докладніше розглянемо складніші моделі запитів і складніші структури індексів, необхідні для ефективної обробки таких запитів.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- Перерахуємо наші побажання:
- 1. Ми хотіли б краще визначати набір термінів у словнику і здійснювати пошук, малочутливий до помилок і нечіткому вибору слів.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- 2. Часто хотілося б мати можливість знаходити складені слова чи фрази, що позначають якесь поняття, наприклад "операційна система".
  - Як свідчать приклади, пов'язані з системою Westlaw хотілося б також мати можливість формулювати запити про близькість термінів, наприклад Gates NEAR Microsoft.
  - Для відповіді на такі запити індекс повинен бути істотно доповнений, щоб відображати в якійсь формі близькість між термінами в документі.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- 3. Модель булевого пошуку дозволяє лише визначити наявність або відсутність терміна, але часто хотілося б використовувати більше доступної нам інформації, наприклад, ми могли б присвоювати більшу вагу документом, у яких термін зустрічається кілька разів, і меншу вагу - документам, в яких термін зустрічається лише один раз .
- Для цього в список словопозицій необхідно ввести інформацію про частоту терміна, тобто про кількість появ терміну в документі.

ПОРІВНЯННЯ РОЗШИРЕНОЇ БУЛЕВОЇ МОДЕЛІ І РАНЖОВАНОГО ПОШУКУ

- 4. У відповідь на булеві запити просто повертається (невпорядкована) безліч документів, але зазвичай нам необхідний ефективний метод, що б дозволяв впорядкувати (ранжувати) результати пошуку.
  - Для цього потрібен механізм визначення рангу документа, який включав би в себе ступінь відповідності документа запиту.

- Ваші запитання.
- Дякую за увагу.