УДК 681.3

М.А.Демихов

МЕТОДЫ НЕЧЕТКОГО ПОИСКА В ИНФОРМАЦИОННЫХ СИСТЕМАХ

Воронежский институт высоких технологий

В работе рассмотрены особенности методов нечеткого поиска, которые используются в современных информационных системах. Отмечена роль расстояния Левенштейна, позволяющего оценить характеристики алгоритмов. Обсуждаются возможности методов сэмплирования.

Ключевые слова: нечеткий поиск, информационная система, текст, метод, индексация.

В настоящее время алгоритмы для проведения нечеткого поиска строк имеют большое распространение по системам автоматизации перевода, орфографическим корректорам, программам распознавания печатного текста и даже поисковым системам [1-6].

Для общего случая в нечетком текстовом поиске подразумевается отыскание произвольных компонентов в тексте, но часто задачи могут свести к словарному поиску. Так, например, проведение индексации для многих современных поисковых систем и для электронных каталогов документов базируется на том, что применяют технологию инвертирования**.**

Процессы инвертирования и составления алфавитного указателя имеют много общего и основаны на выделении значимых ключевых слов и составлении списков вхождений ключевых слов в текст. Полученную в результате такого преобразования структуру данных принято называть инвертированным индексом или инвертированным файлом, а список ключевых слов – словарем.

Поиск в инвертированном файле осуществляется в два этапа: сначала происходит выборка слов запроса из словаря, а затем считываются и обрабатываются соответствующие списки вхождений.

Большинство современных электронных библиотек являются коллекцией документов, которые снабжены инвертированными индексами для того, чтобы был быстрый доступ.

Основным компонентом в поисковых модулях может быть назван словарный поиск. Появление ошибок и искажений может быть, как для тех документов, которые вновь добавляют в систему, так и для запросов пользователей, в этой связи задача по эффективному нечеткому словарному поиску появляется, как для этапов создания документов, так и для этапов поиска по той коллекции, которая уже проиндексирована [7-10].

Алгоритм расширения выборки исследователи весьма часто используют для систем, направленных на проверку орфографии. Он базируется на том, что задача о нечетком поиске сводится к задачам о проведении точного поиска. Указанный способ подразумевает, что будут строиться наиболее вероятные «неправильные» варианты в поисковом шаблоне. Другими словами, происходит построение множества различных «ошибочных» слов, которые, например, получаются из исходного после того, как была сделана одна операция редактирования, затем происходит сравнение сформированных терминов по точному соответствию. В качестве основного плюса указанного алгоритма можно указать легкость его модификации для того, чтобы генерировать «ошибочные» варианты в рамках произвольных правил. Но при этом существуют и определенные минусы, основной из которых заключается в большом числе проверок, относящимся к словам, имеющим существенную длину, так как из них есть возможность получения большого числа «ошибочных» слов.

В настоящее время большая часть работ в области нечеткого строкового поиска относится к поиску без предварительного индексирования, который в англоязычных работах часто называется on-line поиском.

Словарный поиск с предварительной индексацией (off-line поиск) является сравнительно малоизученным направлением.

Хотя разработано немало методов и алгоритмов таких, как n-граммная индексация, основанная на индексации фиксированной длины, различные модификации метрических деревьев, алгоритмы поиска в абстрактных метрических пространствах, trie-деревья (лучи) практически невозможно найти работы, посвященные сравнительному анализу алгоритмов нечеткого словарного поиска.

Для функции, которая характеризует похожесть строк, можно сказать, что это основа в нечетком словарном поиске. Проведение выбора подходящей функции похожести оказывает существенное влияние не только на то, какое качество выборки и какая скорость будет поиска, но и еще определяет сложность того, как будет реализован индекс.

Для хорошей функции близости слов есть учет различных типов искажений, в том числе это касается удалений, замен, вставок и транспозиции символов, а для самого хорошего случая говорят о похожести звучания слов.

Среди первых мер близости слов была предложена функция Левенштейна. Расстояние Левенштейна равняется минимальному числу элементарных операций редактирования, которые требуются для того, чтобы преобразовать одну строку в другую.

Алгоритм Вагнера-Фишера дает возможности при рассмотрении двух строк определить расстояние Левенштейна. В указанном алгоритме

можно увидеть ряд достоинств по сравнению с другими алгоритмами, а именно: он характеризуется относительно невысокой сложностью при реализации, есть возможности для того, чтобы качественным образом сравнивать схожесть для числа строк более, чем две, могут быть разные варианты реализации, их можно применять в зависимости от того, какая конфигурация системы, он универсален для различных алфавитов. Среди недостатков можно отметить такие, что при проведении перестановки слов между собой или их частей мы можем получить достаточно большие расстояния. Между достаточно различными короткими словами получаются малые значения, а для похожих и длинных строк — значительные.

Для расстояния Левенштейна такой набор содержит операции по замене, вставке и удалению одного символа. Для модификации расстояния редактирования, которая была предложена Дамерау, в множество элементарных операций включатся транспозиции символов.

Следует сказать, что при этом необходимо, чтобы для транспонированных символов не было применения других операций редактирования. Найденное подобным образом расстояние редактирования может быть определено на основе метода динамического программирования.

Алгоритм имеет сложность O(MN), где M и N – длины сравниваемых строк, а для нахождения значения расстояния требуется вычислить MN элементов так называемой матрицы динамического программирования. Сложность вычисления расстояния Левенштейна–Дамерау имеет квадратичный порядок относительно размера строк.

Значительное количество работ посвящено созданию более эффективных алгоритмов [11-15]. Предложенные процедуры вычисления можно условно разделить на две категории.

Первую категорию составляют алгоритмы. В их основе лежит алгоритм динамического программирования, но для определения расстояния редактирования не требуется вычислять все MN элементов матрицы.

Вторую категорию составляют алгоритмы, основанные на эффективном использовании битовых операций.

Другой подход к задаче ускорения вычисления расстояния редактирования заключается в выборе более легко вычислимой функции похожести. Так, например, были предложены и хорошо исследованы различные модификации n-граммных расстояний, основанные на подсчете числа общих подстрок фиксированной длины. Нам также известны примеры функций близости, основанные на вычислении длины наибольшей общей подпоследовательности двух строк.

В настоящее время предложено множество альтернативных функций близости, но расстояние Левенштейна–Дамерау наиболее точно соответствует интуитивному понятию похожести. Кроме того, можно обобщить функцию Левенштейна, чтобы она точнее оценивала фонетическую похожесть слов.

Далее мы будем рассматривать алгоритмы, ищущие лишь относительного элементарного расстояния редактирования (без учета транспозиций и весов символьных преобразований). Такой подход оправдан в случае, когда функция Левенштейна может быть использована в качестве фильтра.

Если объем выборки, полученной с помощью простых алгоритмов, невелик, то для каждой найденной строки можно уточнить расстояние до поискового образца, используя более качественную и ресурсоемкую функцию [16-19].

Цель индексации списка слов – ускорение поиска по сходству. Под поиском по сходству подразумевается отыскание всех слов, для которых расстояние (в нашей работе расстояние Левенштейна) до поискового шаблона не превышает заданную величину.

Алгоритмы, которые позволяют отыскать все строки словаря в заданной окрестности поискового термина, мы будем называть детерминированными. Поскольку понятие меры близости само определено неточно, не всегда имеет смысл выборка всех слов в заданной окрестности поискового шаблона.

Существуют алгоритмы, которые находят большую часть, но не гарантируют нахождение всех строк.

Такие алгоритмы мы будем называть рандомизированными. Типичным примером является поиск слов, имеющих то же значение функции soundex, что и искомое слово.

Довольно распространенным подходом к реализации алгоритмов рандомизированного поиска является индексация относительно значений нескольких хеш-функций. Каждая из хеш-функций преобразует слова в числовые значения.

Например, в алгоритме локально устойчивого хеширования хеш-функции строятся так, что чем меньше расстояние между двумя словами, тем больше вероятность того, что значения хеш- функций на этих словах совпадают. Уменьшая или увеличивая количество хеш-функций, можно достичь желаемого соотношения скорости поиска и полноты выборки.

Рандомизированные алгоритмы весьма разнообразны. С одной стороны, они, как правило, используют индексацию на точное равенство, а потому весьма эффективны, а с другой стороны сильно отличаются по полноте и точности выборки.

Это означает, что сравнивать такие алгоритмы следует в первую очередь относительно качества выборки, что не является целью нашей работы. В дальнейшем мы будем рассматривать только детерминированные относительно функции Левенштейна методы поиска.

В случае тривиального индексирования, когда запрос обрабатывается методом последовательного перебора, не требуется никакого дополнительного преобразования. Недостаток такого подхода – низкая эффективность.

Выделяя в строках общие элементы, можно использовать их для сокращения перебора. Мы будем называть процесс выделения характерных элементов строк сэмплированием. Чаще всего используются следующие методы сэмплирования:

- сэмплирование подстрок: префиксов, суффик– сов или n-грамм
- буквенное сэмплирование
- метрическое сэмплирование

Метрический метод сэмплирования заключается в вычислении вектора расстояний от заданной строки до образующих элементов. Образующие элементы могут быть выбраны из существующих строк словаря или сгенерированы.

В целях поиска сэмплы могут быть преобразованы. Так, например, на основании набора n-грамм могут быть построены, как инвертированные списки, так и частотные вектора. Следующие два вида преобразований сэмплов наиболее распространены:

- построение частотного вектора
- построение сигнатурного вектора

Для построения вектора требуется хеш-функция H(s), отображающая сэмпл в целое число, которое трактуется как индекс элемента вектора. Сигнатурный вектор – это битовый массив, i-ый элемент которого равен единице, если существует сэмпл s такой, что f(s) = i. Частотный вектор – это массив, i-ый элемент которого равен числу сэмплов s таких, что f(s) = i.

Осуществив преобразование строк в сэмплы, можно проиндексировать словарь для быстрого доступа. Для этого можно использовать следующие алгоритмы:

- trie-деревья;
- инвертированный индекс;
- координатные структуры, например, kd- деревья и rd-деревья;
- “точный индекс”, используемый для поиска методом расширения выборки.

Вывод. Комбинируя различные методы сэмплирования и индексирования, можно строить новые алгоритмы. Разнообразие существующих алгоритмов – это основная проблема, возникающая при их

экспериментальной проверке.

ЛИТЕРАТУРА

1. Фомина Ю.А., Преображенский Ю.П. Принципы индексации информации в поисковых системах / Вестник Воронежского института высоких технологий. 2010. № 7. С. 98-100.
2. Зяблов Е.Л., Преображенский Ю.П. Построение объектно-семантической модели системы управления / Вестник Воронежского института высоких технологий. 2008. № 3. С. 029-030.
3. Преображенский Ю.П. Разработка методов формализации задач на основе семантической модели предметной области / Вестник Воронежского института высоких технологий. 2008. № 3. С. 075-077.
4. Преображенский Ю.П. Оценка эффективности применения системы интеллектуальной поддержки принятия решений / Вестник Воронежского института высоких технологий. 2009. № 5. С. 116-119.
5. Зазулин А.В., Преображенский Ю.П. Особенности построения семантических моделей предметной области / Вестник Воронежского института высоких технологий. 2008. № 3. С. 026-028.
6. Преображенский Ю.П. Разработка методов формализации задач на основе семантической модели предметной области / Вестник Воронежского института высоких технологий. 2008. № 3. С. 075-077.
7. Navarro G., Baeza-Yates R., Sutinen E., Tarhio J. Indexing Methods for Approximate String Matching. In IEEE Data Engineering Bulletin, volume 24(4), pages 19-27, 2007.
8. Иванов М.С., Преображенский Ю.П. Разработка алгоритма отсечения деревьев / Вестник Воронежского института высоких технологий. 2008. № 3. С. 031-032.
9. Зяблов Е.Л., Преображенский Ю.П. Разработка лингвистических средств интеллектуальной поддержки на основе имитационно-семантического моделирования / Вестник Воронежского института высоких технологий. 2009. № 5. С. 024-026.
10. Чопоров О.Н., Чупеев А.Н., Брегеда С.Ю. Методы анализа значимости показателей при классификационном и прогностическом моделировании / Вестник Воронежского государственного технического университета. 2008. Т. 4. № 9. С. 92-94.
11. Завьялов Д.В. О применении информационных технологий / Современные наукоемкие технологии. 2013. № 8-1. С. 71-72.
12. Чопоров О.Н., Наумов Н.В., Куташова Л.А., Агарков А.И. Методы предварительной обработки информации при системном анализе и моделировании медицинских систем / Врач-аспирант. 2012. Т. 55. № 6.2. С. 382-390.

13. Чопоров О.Н., Агарков А.И., Куташова Л.А., Коновалова Е.Ю. Методика преобразования качественных характеристик в численные оценки при обработке результатов медико-социального исследования / Вестник Воронежского института высоких технологий. 2012. № 9. С. 96-98.
14. Малышев В. А. Основные проблемы научных исследований при разработке и совершенствовании технических систем / Вестник Воронежского института высоких технологий. 2015. № 14. С. 8-11.
15. Питолин М. В., Мачтаков С. Г. Постановка задачи классификации ассоциативного поиска объектов в базе данных / Вестник Воронежского института высоких технологий. 2015. № 14. С. 37-39.
16. Ахо Альфред, Хопкрофт В., Джон, Ульман, Джеффри Д. Структуры данных и алгоритмы. - Издательский дом "Вильямс", 2000. - 384 с.
17. Wu S., Manber U. Fast Text Searching with Errors. In Communications of the ACM, volume 35 pages 83-91, 2009.
18. Бойцов Л.М. Использование хеширования по сигнатуре для поиска по сходству // Прикладная математика и информатика , ВМиК МГУ, No 8, 2001. С . 135-154.
19. Кубенский А.А. Структуры и алгоритмы обработки данных. / БХВ-Петербург, 2011. - 464 с.

M.A.Demikhov

THE METHODS FOR FUZZY SEARCH IN INFORMATION SYSTEMS

Voronezh Institute of High Technologies

The paper discusses the features of the fuzzy search methods which are used in modern information systems. The role of the Levenshtein distance, which measures the characteristics of the algorithms, is pointed out. The possibility of sampling methods is discussed.

Keywords: fuzzy search, information system, text, method, indexing.