Манучарян Левон Ашотович

ОПТИМИЗАЦИЯ ПРОИЗВОДИТЕЛЬНОСТИ СИСТЕМ ИЗВЛЕЧЕНИЯ ИНФОРМАЦИИ

Адрес статьи: www.gramota.net/materials/1/2011/9/10.html

Источник

Альманах современной науки и образования

Тамбов: Грамота, 2011. № 9 (52). С. 35-37. ISSN 1993-5552.

Адрес журнала: www.gramota.net/editions/1.html

Содержание данного номера журнала: www.gramota.net/materials/1/2011/9/

существующим экспериментальным данным, имея гораздо меньшую погрешность по отношению к последним, чем существующие решения.

Применение теоретических разработок, сгенерированных в работе, позволит в значительной степени улучшить режимы работы оборудования, используемого в современном металлургическом производстве, снизить его массогабаритные показатели, увеличить его эксплуатационный ресурс. Полученные в работе теоретические расчётные данные по интенсифицированному теплообмену могут быть рекомендованы для расчёта виртуальных возможностей перспективного теплообменного оборудования современного металлургического производства в целях дальнейшего его совершенствования.

Список литературы

1. **Дрейцер Г. А., Лобанов И. Е.** Моделирование изотермического теплообмена при турбулентном течении в каналах в условиях интенсификации теплообмена // Теплоэнергетика. 2003. № 1. С. 54-60.
2. **Калинин Э. К., Дрейцер Г. А., Ярхо С. А.** Интенсификация теплообмена в каналах. М.: Машиностроение, 1990. 208 с.
3. **Кутателадзе С. С.** Основы теории теплообмена. М.: Атомиздат, 1979. 416 с.
4. **Лобанов И. Е.** Математическое моделирование интенсифицированного теплообмена при турбулентном течении в каналах: дисс. … доктора технических наук. М., 2005. 632 с.
5. **Лобанов И. Е.** Моделирование теплообмена и сопротивления при турбулентном течении в каналах теплоносителей в условиях интенсификации теплообмена // Труды Третьей российской национальной конференции по теплообмену: в 8-ми т. М., 2002. Т. 6. Интенсификация теплообмена. Радиационный и сложный теплообмен. С. 140-143.
6. **Мигай В. К.** Интенсификация конвективного теплообмена в трубах и каналах теплообменного оборудования: дисс. … доктора технических наук. Л., 1973. Т. 1. 327 с.; Т. 2. 85 с.
7. **Мигай В. К.** Моделирование теплообменного энергетического оборудования. Л.: Энергоатомиздат (Ленинградское отделение), 1987. 263 с.
8. **Мигай В. К.** Повышение эффективности современных теплообменников. Л.: Энергия (Ленинградское отделение), 1980. 144 с.
9. **Эффективные поверхности теплообмена** / Э. К. Калинин, Г. А. Дрейцер, И. З. Копп и др. М.: Энергоатомиздат, 1998. 408 с.

УДК 681.3

Левон Ашотович Манучарян
Воронежская государственная лесотехническая академия

ОПТИМИЗАЦИЯ ПРОИЗВОДИТЕЛЬНОСТИ СИСТЕМ ИЗВЛЕЧЕНИЯ ИНФОРМАЦИИ©

Извлечение смысловой информации является главным направлением эволюции глобальной сети на данный момент, а системы и приложения, предназначенные для извлечения информации, должны обеспечить высокую производительность для эффективного взаимодействия с конечным пользователем. Несмотря на факт, что многие системы извлечения применяются в коммерческих или исследовательных приложениях, опубликованные исследования, адресованные разным аспектом производительности при извлечении информации, начали появлятся только в последнее время. Системы извлечения могут быть внедрены в двух режимах. При одном режиме, неструктурированный источник заранее доступен, например, в закрытых системах вроде хранилища данных или системы обработки жалоб. При втором режиме, неструктурированный источник является открытым и довольно обширным, например, всемирная сеть. В таком случае часть процесса извлечения подразумевает выбор релевантного подмножества документов. Обычно, при первом режиме, пользователь заинтересован в аннотации всех встречающихся сущностей/связей в неструктурированном источнике, тогда как при втором режиме, целью является создание хранилища структурированных сущностей. Поэтому обрабатываются только документы, с наибольшей вероятностью содержащие новые сущности. В статье рассматриваются оптимизации, которые делают выбор документов эффективным. Далее следует фаза фильтрации внутри документа, с рассмотрением только релевантных (представляющих интерес) частей в документе. Например, в случае с системой цитирования, необходимо разработать быстрые тесты для нахождения заголовков и ссылочных разделов документов. Существующие решения в данном направлении предметно-специфичны, поэтому не существует обобщенной системы для рассмотрения. Конечным шагов является применение экстракторов на подмножестве выбранных документов. Большинство алгоритмов извлечения линейно масштабируются в зависимости от размеров входных данных. Но даже в таком случае, существует необходимость оптимизации производительности, так как предварительная обработка и генерация свойств являются довольно затратными задачами. Существующие системы, основанные на шаблонах извлечения, почти всегда базированы на интенсивном употреблении процессорных ресурсов. Ввод/вывод фигурирует только при нахождении совпадении сущностей с существующей базой данных сущностей. Оба метода, основанные на правилах и на статистике, для распознавания сущностей и связей

зависят от свойств заданного текста. Ресурсы, потраченные на оценку свойств, варьируются. Для проверки, начинается ли слово с заглавной буквы, трата минимальная, тогда как для проверки, имеет ли данная последовательность большое соответствие с базой данных сущностей, траты существенны. Далее в статье будет представлено описание методов для эффективной оценки таких затратных соответствий при больших базах данных. Также будут представлены методы оптимизации, применяемые при извлечении, которые зависят от смешанных затратных и незатратных для оценки свойствах.

Стратегии выборки документов

Когда источник действительно объемный, не существует альтернативы, кроме ручного ограничения набора, посредством определения списка адресов, с которых быдет проводится извлечение. Например, DBLife [5] использует список URL адресов, указывающих на домашнние страницы исследователей баз данных, сайтов конференции и списков рассылки. Когда такое ручное позиционирование невозможно, появляется необходимость в дополнительных уровнях сокращения. Для данной задачи, были предложены два решения. Первое, применимое только к гиперссылочным источникам, заключается в применении некоторой формы сфокусированного сканирования [2]. Второй вариант заключается в использовании уже существующих индексов в неструктурированном источнике для рассмотрения только документов, представляющих интерес. Это создает дополнительные вопросы, например, как искать индекс и как создавать соответствующие индексы для извлечения, ответ на которых будет дан далее в статье. Даже когда документ уже выбран, применение полноценного алгоритма извлечения на целом документе зачастую довольно затратно. Есть возможность создания более передовых тестов, таких как статистический классификатор общего документа в виде фильтра второго уровня релевантности. Существует интересный компромисс между полнотой и потраченным временем между следующими тремя вариантами выборки документов: фокусированное сканирование, поиск по ключевым словам и фильтрация документов после выбора документов посредством применения классикатора. Данная задача рассмотрена как проблема оптимизации в [8].

Методы поиска индекса

Взависимости от числа индексов, запросы могут быть двух типов: стандартные запросы с ключевыми словами IR-типа и шаблонные запросы для фильтрации на уровне сущностей. Метод с ключевыми словами обычно применяется для грубой фильтрации документов на уровне заголовков. Например, фильтры на ключевых словах «вакцина» и «лекарство» используются в [6] для помечивания подмножества документов, содержащих информацию о вспышках заболевания. Шаблонные запросы часто предлагают более тонкую фильтрацию с интересующами сущностями в текстовом корпусе. Специализированные индексы и алгоритмы поиска необходимы для поддержки запросов на уровне шаблонов, особенно если предполагается поддержка шаблонных запросов на уровне символов, например, “<a href=.* ˙ ogg>”. Некоторые стратегии по поиску инвертированого индекса при помощи регулярных выражений были предложены в [4].

Создание индексов для эффективного извлечения: Использование индексов для фильтрации при извлечении информации порождает много вопросов, которые не были должным образом адресованы. Впринципе, индекс должен обеспечить эффективную поддержку для запросов, шаблонов регулярных выражений и делать возможным эффективное хранение тегов вроде тегов части речи, грамматических оборотов и общих тегов сущностей, например имен компании и людей, а также тегов стандартных онтологий, в виде *WordNet* [10]. *Cafarella and Etzioni* [1], например, предлагают дополнить стандартный инвертированный индекс для поиска ключевых имен информацией о соседних тегах вместе с каждым вхождением в инвертированном списке. В то время как это является эффективным решением для некоторых типов запросов, расходы с хранением такой информации существенны. Для того, чтоб эффективным образом поддержать возможность запросов, включающих регулярные выражения, на уровне символов, необходимо зайти дальше существующих инвертированных индексов на уровне слов. Два возможных решения, использование индексов суффиксного дерева (suffix trees) и q-грамм (q-gram). Суффиксное дерево является классическим решением для шаблонных поисковых запросов, однако является слишком требовательным к пространству и таким образом, неэффективно в случае с большим объемом данных. Инвертированные индексы являются общепринятым вариантом для поисков ключевых имен в больших количествах. Один из вариантов поддержки поиска на уровне символов на инвертированных индексах является индексация q-грамм [9].

Эффективные запросы к базе данных сущностей для извлечений

Так как решения об извлечении зависят от ряда шаблонов, только один из которых будет соответствовать существующей базе данных, обычно нужно найти значения соответствий для каждого возможного измерения во входном документе. Также, бессмысленно искать точное совпадение между входным неструктурированным источником и базой данных сущностей, имея в виду, что незначительное несоответствие в формах неизбежно в реальных данных. Проблема формализуется следующим образом: Допустим, на входе имеется последовательность слов x и объемная база данных сущностей D. Целью является нахождение всех возможных сегментов в x, схожесть которого с записью в D больше, чем заданный порог. Это называется проблемой поиска Batch-Top-K. Далее концентрация будет сделана на оценку схожести TF-IDF [7], который показал себя, как наиболее эффективный метод при текстовом поиске. Оценка схожести между двумя записями n_1 и n_2 определяется как:

$$TF-IDF(n_1,n_2)=\sum_{t\in n_1\cap n_2}V(t,n_1)V(t,n_2)$$

$$V(t,n)=\frac{V'(t,r)}{\sum_{t'\in n}V'(t',n)^2} \quad (1)$$

$$V'(t,n)=\log\left(TF(t,n)+1\right)\log\left(IDF(t)\right)$$

В формуле, IDF делает вес слова обратно пропорциональным частоте появления в базе данных, а TF делает его пропорциональным частоте в записи. Таким образом, часто встречающимся словам назначаются небольшие значения веса, в то время как редко встречающимся - высокие. Базовая проблема поиска Top-K со степенем сходства TF-IDF была интенсивно изучено [3]. Таким образом, простым механизмом решения Batch-Top-K проблемы является запуск базового Top-K алгоритма для каждого сегмента во входном предложении. Однако, так как сегменты содержат совпадающие слова, можно достичь хороших результатов, группируя результаты вычисления.

Список литературы

1. **Cafarella M. J., Etzioni O.** A Search Engine for Natural Language Applications // WWW Conference. 2005. P. 442-452.
2. **Chakrabarti S., Punera K., Subramanyam M.** Accelerated Focused Crawling through Online Relevance Feedback // Ibidem. 2002.
3. **Chaudhuri S., Ganjam K., Ganti V., Motwani R.** Robust and Efficient Fuzzy Match for Online Data Cleaning // SIGMOD. 2003.
4. **Cho J., Rajagopalan S.** A Fast Regular Expression Indexing Engine // ICDE. 2002. P. 419-430.
5. **DeRose P., Shen W., F. C. 0002, Lee Y., Burdick D., Doan A., Ramakrishnan R.** DBLife: a Community Information Management Platform for the Database Research Community (Demo) // CIDR. 2007. P. 169-172.
6. **Grishman R., Huttunen S., Yangarber R.** Information Extraction for Enhanced Access to Disease Outbreak Reports // Journal of Biomedical Informatics. 2002. Vol. 35. P. 236-246.
7. **http://ru.wikipedia.org/wiki/TF-IDF**
8. **Ipeirotis P. G., Agichtein E., Jain P., Gravano L.** Towards a Query Optimizer for Text-Centric Tasks // ACM Transactions on Database Systems. 2007. Vol. 32.
9. **Kim M.-S., Whang K.-Y., Lee J.-G., Lee M.-J.** N-Gram/2l: a Space and Time Efficient Two-Level N-Gram Inverted Index Structure // VLDB'05: Proceedings of the 31st International Conference on Very Large Data Bases. 2005. P. 325-336.
10. **Resnik P., Elkiss A.** The Linguist's Search Engine: an Overview (Demonstration) // ACL. 2005.

УДК 519.816

Виктор Сергеевич Раков
Братский государственный университет

ДИАЛОГОВЫЕ ГРАФИЧЕСКИЕ МЕТОДЫ И МОДЕЛИ МНОГОКРИТЕРИАЛЬНОЙ ОПТИМИЗАЦИИ©

Решение многокритериальных задач представляется в виде множества допустимых решений, которое сводится к одному из его подмножеств, называемому подмножеством эффективных решений. Решение эффективно, если не существует такого же хорошего по всем критериям и строго лучшего хотя бы по одному из них. Множество допустимых эффективных решений может быть представлено ориентированным графом, в котором варианты решения вершины, а связи между вариантами - дуги графа. В таком многокритериальном графе, каждой вершине которого присвоена определенная векторная оценка (индикатор), может быть определен эффективный путь среди множества допустимых решений, при С1, С2, ..., Ci, ..., Ск - различных критериях (1<=i<=к) и Qj индикаторах (1<=j<=m).

Для многокритериальных задач типичным является определение такого эффективного пути подграфа, который состоял бы только из оптимальных индикаторов. Геометрическая интерпретация эффективного подграфа выражается траекторией, проходящей через все точки, отображающие оптимальные индикаторы, и характеризуется величиной

$$S_Q=\sum_{j=1}l_j(Q_j) \quad (1)$$

где lj(Qj) - расстояние от начала координатной системы до j-й точки, отображающей Qj индикатор. Алгоритм многокритериальной оптимизации включает:

- построение графа вариантов решений;
- разбиение графа на блоки;
- составление упорядоченного списка индикаторов;
- добавление фиктивного индикатора Qj=(O...O)T, если имеется несколько начальных или конечных вершин;