# Семантический поиск как средство взаимодействия с электронной библиотекой

© Д. А. Малахов © Ю. А. Сидоренко © О. М. Атаева © В. А. Серебряков

МГУ им. М.В. Ломоносова ВЦ им. А.А. Дородницына РАН

Москва

79155155577@ya.ru sidorenkoyury@gmail.com oli@ultimeta.ru

serebr@ultimeta.ru

**Аннотация**

Данная работа описывает решение проблемы семантического поиска по текстам документов. В качестве примера рассматривается семантический поиск по текстам книг цифровой библиотеки LibMeta. Представлен алгоритм построения иерархии ключевых слов и кластеров путем итеративного выполнения кластеризации и выделения ключевых слов. Построенная иерархия используется для генерации рефератов и индексации документов для семантического поиска.

## 1 Введение

Традиционно предполагается, что ресурсы электронных библиотек представляют собой библиографические записи традиционных библиотек и электронные копии документов, описываемых этими записями. Но развитие технологий переопределяет понятие как самих библиотек, так и ее ресурсов, которые не ограничиваются только библиографическими записями и их электронным представлением, но также выводит на передний план семантику этих ресурсов. Для этого могут использоваться различные виды классификации ресурсов библиотеки. Разработаны различные отраслевые рубрикаторы, которые позволяют более детально определить тематическую направленность ресурсов. Как правило, этих средств для описания семантики недостаточно, либо со временем появляются новые требования к описанию ресурсов библиотек, что приводит как к усложнению самих описаний, так и требует значительных трудозатрат на внедрение новых способов описаний, соответствующих текущим потребностям.

Используя новые возможности, которые появляются с развитием технологий, пользователь библиотеки может использовать больше средств для работы с ресурсами цифровых библиотек, имея возможность описывать область своих интересов в терминах предметной области на основе стандартов с привлечением тезаурусов словарей и онтологий. Это позволяет ему организовывать и описывать как собственные коллекции, так и собственные ресурсы, при необходимости детализировать описания ресурсов и свою область интересов, уточняя ее термины.

Персональная открытая семантическая цифровая библиотека LibMeta[1] характеризуется гибким хранилищем метаданных для своих ресурсов и типами описываемых информационных ресурсов. Такой подход к описанию ресурсов библиотеки обеспечивает универсальность описания ее типов ресурсов и объектов независимо от предметной области и области интересов пользователей. Структурированность описания обеспечивает поддержку связей между различными типами ресурсов.

Гибкость описания ресурсов обеспечивается использованием OWL онтологий для хранения метаданных. Такой подход дает ряд преимуществ:

- возможность выполнения SPARQL запросов;
- получение дополнительных знаний с помощью логического вывода;
- упрощение интеграции с другими библиотеками;
- возможность изменения схемы под изменившиеся потребности.

Семантический поиск – поиск документов по их содержанию. Библиотека LibMeta позволяет осуществлять семантический поиск по метаданным с помощью SPARQL запросов. При этом в библиотеке не реализован семантический поиск по текстам книг.

Целью работы является улучшение качества услуг, оказываемых библиотекой LibMeta, с помощью семантического поиска по текстам книг библиотеки.

Таким образом, необходимо реализовать систему семантического поиска по текстам книг библиотеки LibMeta. Поисковая система должна находить по

**Труды XVIII Международной конференции DAMDID/RCDL'2016 «Аналитика и управление данными в областях с интенсивным использованием данных», Ершово, 11-14 октября 2016**

поисковому запросу на естественном языке релевантные этому запросу тексты книг с учетом семантики. Подразумевается, что для поддержки семантики будут использованы словари синонимов и гипонимов.

## 2 Организация семантического поиска

Существуют разные подходы к организации семантического поиска по текстам. В последние годы наиболее популярным стало семантическое аннотирование текста. Существуют различные способы решения задачи семантического аннотирования. В каждом из них документу или части документа приписывается некоторый набор семантически близких документу меток. В дальнейшем можно искать документы по этим меткам. Кроме того, можно искать документы обычным полнотекстовым поиском, а потом учитывать эти метки при работе с документом, получая больше информации с помощью них [2]. Обычно в качестве меток используются персоны, места, организации или другие субъекты [3].

Для описания меток часто используются RDF хранилища, содержащие набор понятий и отношения между ними. Некоторые методы используют информацию из Wikipedia, как из масштабного источника знаний[4]. В последнее время методы семантического аннотирования все чаще обращаются к использованию массивного, взаимосвязанного облака Linked Open Data [3][5]. Например, с помощью средства семантического аннотирования GATE был проаннотирован Национальный Архив Великобритании (42 TB)[6].

Семантическое аннотирование не единственный способ организации поиска. Существуют решения, основанные на улучшении классического полнотекстового поиска расширением запроса синонимами. Так была создана онтология, основанная на терминах статей с помощью УДК[7], в дальнейшем она использовалась для расширения запроса пользователя. Кроме того, подход, использующий информацию о синтаксисе, морфологии и пунктуации, также кажется интересным [8]. К сожалению, описанные подходы не были внедрены и не используются повсеместно.

Было проведено множество экспериментов по использованию словарей синонимов и гипонимов для улучшения качества полнотекстового поиска. Известно, что при использовании синонимов и гипонимов растет полнота и часто существенно падает точность поиска [9].

Особенность предлагаемого подхода в том, что индексируется не весь текст, а только его значимые части, в зависимости от задачи, это могут быть абзацы, предложения, словосочетания или проставленная человеком метка, например, хэштег. За счет изменения размера значимой части можно контролировать точность и полноту. Например, если полнота маленькая и индексируются предложения, можно попробовать индексировать сочетания предложений. Кроме того, в предлагаемом подходе не используется транзитивность синонимов и гипонимов, для каждого слова нужно явно указать слова, которые могут быть использованы вместо него, это также упрощает контроль над качеством поиска.

## 3 Семантический поиск на базе S-тегов

Рассмотрим модель S-тег, которая предлагается для использования при реализации семантического поиска.

**Определение.** Алфавитом будем называть любое конечное непустое множество. Элементы этого множества называются символами данного алфавита.

**Пример.** В качестве алфавита может выступать любой алфавит естественного языка.

Пусть задан некоторый алфавит A.

**Определение.** Термином алфавита A будем называть любой упорядоченный конечный непустой набор символов алфавита A.

**Пример.** Слова и словосочетания выбранного алфавита естественного языка являются терминами этого алфавита.

Пусть задано множество терминов T алфавита A

**Определение.** S-тегом на множестве T будем называть любое непустое подмножество T.

**Пример.** Поисковый запрос, представляющий собой конъюнкцию слов и словосочетаний, образованных алфавитом естественного языка A является S-тегом на множестве T, где множество T является множеством слов и словосочетаний естественного языка алфавита A.

Пусть задано множество S-тегов ST.

Пусть $\forall t \in T$ задано множество $THS_t \subset T$.

**Определение.** Сужениями термина $t \in T$ будем называть множество:

$$R_t = \{t\} \cup THS_t.$$

**Пример.** Если в качестве терминов рассматривать слова и словосочетания, то в качестве множества $THS_t$ рассмотрим множество синонимов и гипонимов термина t. Тогда множество $R_t$ представляет собой множество, состоящее из термина t, его синонимов и гипонимов.

**Определение.** Классом термина $t \in T$ будем называть множество:

$$Class_t = \{ st \in ST \mid st \cap R_t \neq \emptyset \}.$$

**Пример.** Если S-тег является поисковым запросом, как было показано ранее, тогда $Class_t$ является множеством поисковых запросов, которые включают термин t или его синонимы, гипонимы.

**Определение.** Сужениями S-тега $st \in ST$ будем называть множество:

$$R_{st} = \bigcap_{t \in st} Class_t.$$

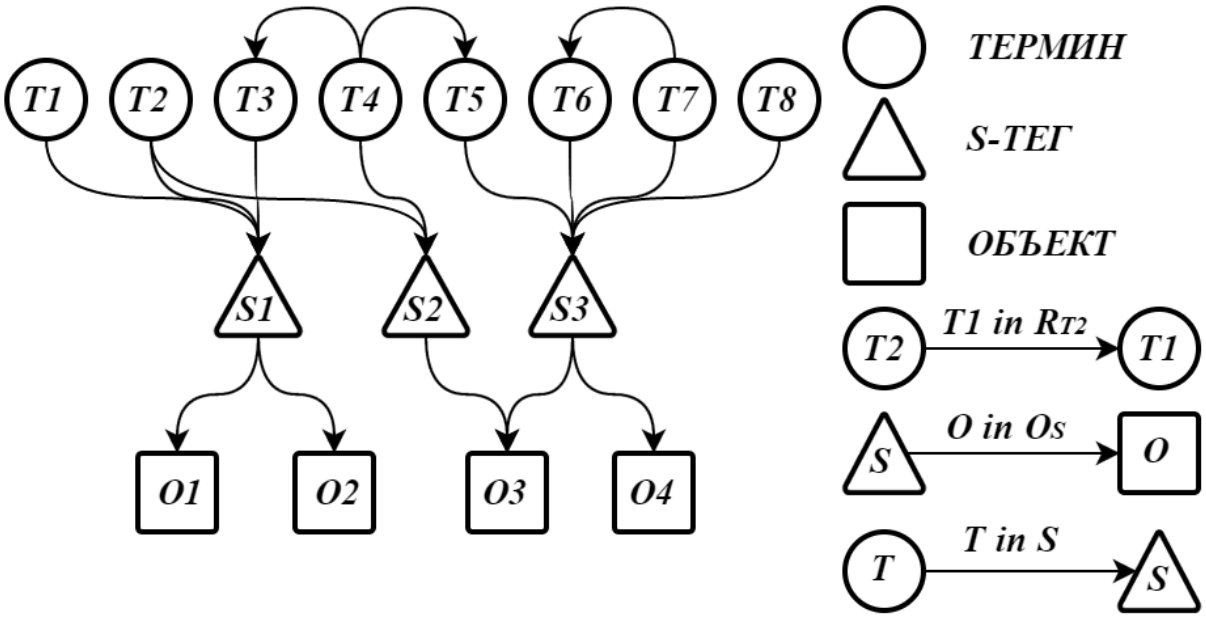


**Рисунок 1** Отношения терминов, объектов и S-тегов

**Пример.** Сужениями поискового запроса st являются более частные или эквивалентные запросы, которые для каждого термина запроса st содержат или этот термин или его сужение. Из определения следует, что сам запрос является своим сужением.

Пусть задано множество объектов O.

$\forall$st ∈ ST задано множество $O_{st} \subset$ O.

**Определение.** Классом S-тега st будем называть множество:

$$Class_{st} = \bigcap_{\text{rst} \in R_{st}} O_{rst}.$$

**Пример.** В качестве примера множества объектов O можно рассмотреть тексты книг. Из текста книги могут быть выделены запросы, которым этот текст является релевантным. Рассмотрим поисковый запрос st. Множество $O_{st}$ является множеством текстов, в которых выделен запрос st. Если текст является релевантным более частному запросу по сравнению с st, то он должен быть релевантным запросу st. Отсюда следует, что $Class_{st}$ является множеством релевантных запросу st текстов книг.

На Рис. 1 представлена визуализация связей между терминами, S-тегами и объектами. Явно указано, что объект O3 ∈ $Class_{S2}$. Кроме того, объекты O1 ∈ $Class_{S2}$ и O2 ∈ $Class_{S2}$, так как S1 ∈ $R_{S2}$. Объект O4 ∉ Class_S2, так как S3 ∉ R_S2. Это следует из того, что термин T2 и термины из $R_{T2}$ не включены в S-тег S3.

Под семантический поиском на базе S-тегов будем понимать поиск текстов книг, которые являются релевантными заданному поисковому запросу, который является S-тегом. Поиск может считаться семантическим, так как использует синонимы и гипонимы, позволяющие передать смысл текста.

Согласно приведенным примерам, задача поиска релевантных текстов книг по заданному поисковому запросу на естественном языке сводится к задаче нахождения $Class_{st}$ для заданного S-тега st. Рассмотрим решение этой задачи.

В первую очередь нужно найти $R_{st}$. Для этого достаточно найти $Class_t$ для каждого термина S-тега st.

Первый способ определения $Class_t$ требует хранения инвертированного индекса $II_{ST}$, где каждому t ∈ T соответствует инвертированный список S-тегов $IL_t$: {st | t ∈ st}. В этом случае поисковоый запрос st должен быть обогащен для каждого своего термина t терминами из $R_t$:

$$Class_t = \bigcap_{\text{rt} \in R_t} IL_{rt}.$$

Предложенный способ требует дополнительных затрат на получение $IL_{rt}$. В случае большого тезауруса эти затраты могут быть значительными.

Второй способ определения $Class_t$ требует хранения инвертированного индекса $II_{ST}$, где каждому t ∈ T соответствует инвертированный список тегов $IL_t = Class_t$. В этом случае размер $II_{ST}$ существенно больше, но скорость поиска выше.

Для решения задачи поиска $Class_{st}$ необходимо иметь инвертированный индекс $II_O$, где каждому st ∈ ST соответствует инвертированный список объектов $IL_{st} = O_{st}$:

$$Class_{st} = \bigcap_{\text{rst} \in R_{st}} IL_{rst}.$$

Решив задачи нахождения $R_{st}$ и $Class_{st}$ для S-тега st, мы получаем решение задачи семантического поиск текстов книг, которые являются релевантными заданному поисковому запросу, как S-тегу.

## 4 Выделение S-тегов из текстов

Как было показано ранее, можно организовать семантический поиск по S-тегам, если они выделены из текстов книг. Рассмотрим способ автоматического выделения S-тегов из текста.

Под ключевыми словами текста мы будем понимать слова и словосочетания, которые передают смысл текста и выделяют его среди других текстов в коллекции.

Чтобы соответствовать содержанию текста, S-тег должен содержать его ключевые слова. S-тег может являться ключевым словом, предложением или абзацем, в котором встретилось ключевое слово. Таким образом, задача выделения S-тегов может быть сведена к поиску ключевых слов в тексте.

Так как ключевое слово текста должно выделять его среди других текстов, то ключевые слова зависят как от текста, так и от коллекции текстов, которой противопоставляется этот текст. Если разбить множество текстов на группы похожих по смыслу текстов, то можно рассматривать ключевое слово как отличительный признак для текста, характеризующий его группу.

С другой стороны, если для разбиения текстов использовать в качестве признаков ключевые слова, то можно существенно повысить скорость и качество разбиения сокращением признакового пространства и фильтрацией шума.

Таким образом, кластеризация, как процесс разбиения коллекции текстов на группы, может использовать выделенные ключевые слова, в то время как алгоритм выделения ключевых слов может использовать результаты кластеризации. С помощью кластеризации можно разбить множество текстов на группы (кластеры), после чего находить ключевые слова для текстов относительно полученных групп. Этот процесс можно повторять несколько раз, чередуя выделение ключевых слов и кластеризацию.

В итоге получим иерархическую структуру документов в коллекции и соответствующую ей иерархию ключевых слов.

## 5 Выделение ключевых слов

Дано множество текстов D на множестве терминов T. Под термином будем понимать слово или словосочетание. Множество D разбито на множество кластеров C. Нужно выделить в текстах из множества D такие ключевые слова, которые характерны для кластера этого текста.

Традиционно выделение ключевых слов делится на два этапа. Первый этап представляет собой выделение кандидатов в ключевые слова. На этом этапе удаляются стоп-слова, могут фильтроваться части речи или, например, фильтроваться кандидаты, которые не содержатся в заголовках статей из Wikipedia. Второй этап заключается в проверке кандидатов на семантическую близость к данному тексту. Для решения этой задачи используют алгоритмы машинного обучения как с учителем, таки без него [10].

Основной особенностью нашей задачи в отличие от стандартной задачи выделения ключевых слов является то, что ключевые слова должны зависеть не только от самого документа, но и документов близких к нему с точки зрения некоторой предметной области. Предлагаемый подход к решению задач может быть улучшен с помощью алгоритмов решения стандартной задачи [10].

Рассмотрим простой подход к решению задачи. Пусть выбран случайный текст $d \in D$. Для каждого кластера $c \in C$ и термина $t \in T$. Оценим вероятность того, что $d \in c$ при условии, что $t \in d$:

$P(d \in c \mid t \in d) = \frac{|(t \in d \wedge d \in c)|}{|(t \in d)|}$, где

- $|(t \in d \wedge d \in c)|$ - количество документов из кластера c, в которых встречается термин t.
- $|(t \in d)|$ - количество документов из кластера c, в которых встречается термин t.

Пусть задан некоторый порог N, тогда будем считать, что термин t характеризует кластер c, если:

$$P(d \in c \mid t \in d) > N * \max_{c_i \in C}(P(d \in c_i \mid t \in d)).$$

Таким образом, для каждого документа выделяются все термины, которые характеризуют кластер этого документа и включены в этот документ, если оценка $P(d \in c \mid t \in d)$ достаточно велика.

## 6 Кластеризация

Дано множество документов D на множестве ключевых слов K. Нужно определить наилучшее число кластеров, на которые можно разбить множество документов D и произвести разбиение.

Для решения задачи воспользуемся методом кластеризации k-means++[11]. Он позволяет за линейное время разбить множество документов на k кластеров.

Критерием качества разбиения с параметром k будем считать значение $Q_k$, равное сумме среднеквадратичных отклонений центров, полученных кластеров за N итераций. Таким образом, чем меньше $Q_k$, тем более устойчивым и качественным является разбиение.

Если k выбрано слишком большим или слишком маленьким, то это скажется на качестве дальнейшего выделения ключевых слов. Поэтому подбирая параметр k, нужно задать верхнюю оценку k1 и нижнюю оценку k2.

Наилучшее значение k находится перебором от k1 до k2. Выбирается такое значение k, при котором значение $Q_k$ за N итераций является минимальным.

**Пример.** Допустим k1 = 8, k2 =16, n =10. Тогда за 80 итераций может быть найдено наилучшее разбиение с точки зрения устойчивости с минимальным значением $Q_k$.


**Рисунок 2** Схема взаимодействия

## 7 Реферирование

Реализовав семантический поиск по текстам книг, мы столкнемся с проблемой отображения результатов поиска. Можно использовать полный текст книги, удовлетворяющий запросу, аннотацию книги или заранее автоматически изготовленный реферат. Более предпочтительным кажется вариант генерации реферата книги по запросу пользователя.

Реферирование – процесс построения краткого содержания (реферата) документа. Реферирование используется для визуализации результатов поиска. Рефераты бывают статические и динамические.

Статические рефераты используются для предоставления краткой информации обо всем документе. Статический реферат формируется один раз и не зависит от поисковой потребности пользователя.

Динамические рефераты генерируются в момент выполнения поискового запроса пользователя и представляют краткую информацию о релевантных запросу частях текста.

В рамках работы был выбран простой алгоритм генерации рефератов, заключающийся в объединении всех выделенных S-тегов и их контекста в случае статических рефератов. В случае генерации динамических рефератов по запросу находятся его сужения и объединяются вместе со своим контекстом.

Предложенный алгоритм может быть улучшен с помощью алгоритма, основанного на доминантах[12]. Подобные подходы крайне популярны.

## 8 Архитектура системы

На Рис. 2 продемонстрирована схема взаимодействия внутри системы семантического поиска по библиотечным данным.

### 8.1 Загрузка данных

Данные необходимо получать из двух источников.

- Источник метаданных поставляет библиографические записи, Метаданные попадают в RDF хранилище, откуда пользователь может их получать с помощью SPARQL запросов. RDF хранилище реализовано на базе библиотеки Jena.
- Источник документов поставляет тексты книг, которые сохраняются в файловую систему. И в индексы СУБД Postgres.

### 8.2 Получение иерархии и ключевых слов

После поступления новых текстов запускается процесс кластеризации, а затем процесс выделения ключевых слов.

Далее для каждого кластера запускается процесс кластеризации на множестве ключевых слов, после чего для каждого кластера снова выделяются ключевые слова.

Эти два процесса выполняются поочередно, пока не будут получены иерархия текстов и множество ключевых слов.

Результаты сохраняются в СУБД Postgres.

### 8.3 Формирование индексов и рефератов

В качестве S-тегов используются предложения, содержащие ключевые слова.

Выделенные S-теги индексируются в СУБД Postgres. Поиск по тегам осуществляется с помощью GIST и GIN. Для каждого S-тега формируем список документов, к которым этот S-тег привязан.

С помощью выделенных S-тегов и их контекста производится генерации статических рефератов.

Для генерации динамического реферата по запросу пользователя находятся все его сужения с помощью полнотекстового поиска СУБД Postgres. На основе контекста найденных S-тегов формируется реферат.

### 8.4 Поиск и тезаурус

Пользователь задает запрос на естественном языке. Перед выполнением поисковый запрос обогащается множеством синонимов и гипонимов из тезауруса.

По запросу система находит его сужения, я для них списки привязанных документов. Для каждого документа формируется динамический реферат.

Пользователь может находить документы с помощью SPARQL запросов по RDF хранилищу.

Тезаурус хранится в файле, где каждому слову соответствует строка, содержащая список его синонимов и гипонимов. Редактируя файл, пользователь может влиять на результаты поиска.

## Заключение

В рамках данной работы был реализован прототип системы семантического поиска по библиографическим данным и текстам книг.

На примере семантической библиотеки LibMeta была продемонстрирована актуальность данной работы. Внедрение описанных подходов позволяет улучшить качество предоставляемых библиотекой услуг.

Были рассмотрены различные подходы к реализации семантического поиска по текстам. Введена модель S-тега и задача поиска сужений S-тега. Задача поиска текста по поисковому запросу была сведена к задаче поиска сужений S-тега. Был представлен алгоритм решения задачи поиска сужений S-тега. Рассмотренная модель была использована при реализации поиска на базе СУБД Postgres.

В рамках работы рассмотрен алгоритм выделения S-тегов из текста. Для работы алгоритма необходимо множество выделенных ключевых слов.

Продемонстрирован процесс построения иерархии ключевых слов с помощью итеративного процесса сменяющих друг друга кластеризации и выделения ключевых слов. Предложенный алгоритм выделения ключевых слов позволяет использовать информацию о кластере документа. Для кластеризации был выбран алгоритм k-means++.

В качестве визуализации результатов семантического поиска по текстам был представлен подход к выделению статических и динамических рефератов.

Предлагаемые алгоритмы могут быть улучшены с помощью существующих решений, но в рамках прототипа были намеренно использованы простые решения. В рамках дальнейшей работы планируется:

- Улучшение качества предложенных алгоритмов выделения ключевых слов, генерации рефератов.
- Проведение экспериментов по улучшению качества кластеризации.
- Реализация эффективного хранилища S-тегов.
- Реализация распределенного выделения ключевых слов на Hadoop кластере;
- Переход к распределенной системе поиска;
- Проведение экспериментов по выделению S-тегов с помощью иерархии классификатора УДК;
- Использование контекстов терминов для семантического поиска;
- Реализация эффективного хранилища S-тегов.

## Литература


[1] Атаева О. М., Серебряков В. А. Персональная цифровая библиотека Libmeta как среда интеграции связанных открытых данных. Труды 16-й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» — RCDL-2014, Дубна, Россия, 2014 .

[2] GiannopoulosG. etal. GoNTogle: a tool for semantic annotation and search .The Semantic Web: Research and Applications, p. 376-380, Springer Berlin Heidelberg, 2010.

[3] Bontcheva K., Tablan V., Cunningham H. Semantic search over documents and ontologies. Bridging Between Information Retrieval and Databases, Springer Berlin Heidelberg, 2014.

[4] Berlanga R., Nebot V., Pérez M. Tailored semantic annotation for semantic search .Web Semantics: Science, Services and Agents on the World Wide Web, p. 69-81, 2015.

[5] Alahmari F., Magee L. Linked Data and Entity Search: A Brief History and Some Ways Ahead. Proceedings of the 3rd Australasian Web Conference, 2015.

[6] Maynard D., Greenwood M. A. Large Scale Semantic Annotation, Indexing and Search at The National Archives. Lrec, p. 3487-3494, 2012.

[7] И.В. Захарова. Об одном подходе к реализации семантического поиска документов в электронных библиотеках. Вестник Уфимского государственного авиационного технического

университета, 2009. http://cyberleninka.ru/article/n/ob-odnom-podhode-k-realizatsii-semanticheskogo-poiska-dokumentov-v-elektronnyh-bibliotekah

[8] А.Л. Воскресенский, Г.К. Хахалин. Средства семантического поиска. Международная конференция по компьютерной лингвистике и интеллектуальным технологиям «Диалог», 2006. http://www.dialog-21.ru/digests/dialog2006/materials/html/Voskresenskij.htm

[9] Н. В. Лукашевич. Тезаурусы в задачах информационного поиска. 2010

[10] Hasan K. S., Ng V. Automatic Keyphrase Extraction: A Survey of the State of the Art //ACL (1), 2014, p 1262-1273. http://acl2014.org/acl2014/P14-1/pdf/P14-1119.pdf

[11] k-means++: The Advantages of Careful Seeding. 2006. http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf

[12] О.Г. Чанышев. Ассоциативные поля доминант и анализ текста. Институт Математики им. С.Л. Соболева СО РАН, 2011. http://elib.ict.nsc.ru/jspui/bitstream/ICT/1376/1/3OHT2.pdf

## Semantic search as a means of interaction with the digital library

Dmitriy A. Malakhov, Yury A. Sidorenko, Olga M. Ataeva, Vladimir A. Serebriakov

This work is devoted to solving the problem of semantic search for document texts. As an example, we consider the semantic search for text of LibMeta digital library books. The proposed approach provides a hierarchy of documents keyword by iteratively performing clustering and selection of keywords. The hierarchy of documents keyword is used to generate abstracts and indexing documents for semantic search.