Исследование рынка полнотекстовых поисковых движков. Выбор поискового движка для doc.appstandard.org

Юргин Павел, SE504

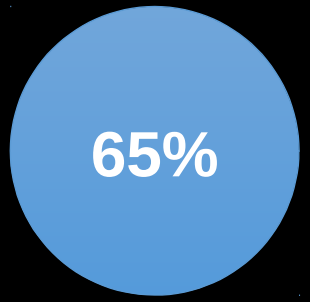
Руководитель: Поляков Николай

Parallels™

What Parallels is?

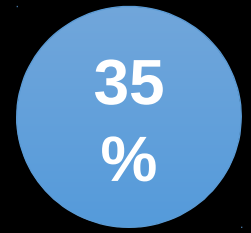
Hosting and Cloud Solutions

- ✓ Hosting and Cloud Automation
- ✓ Web management
- ✓ Cloud Infrastructure

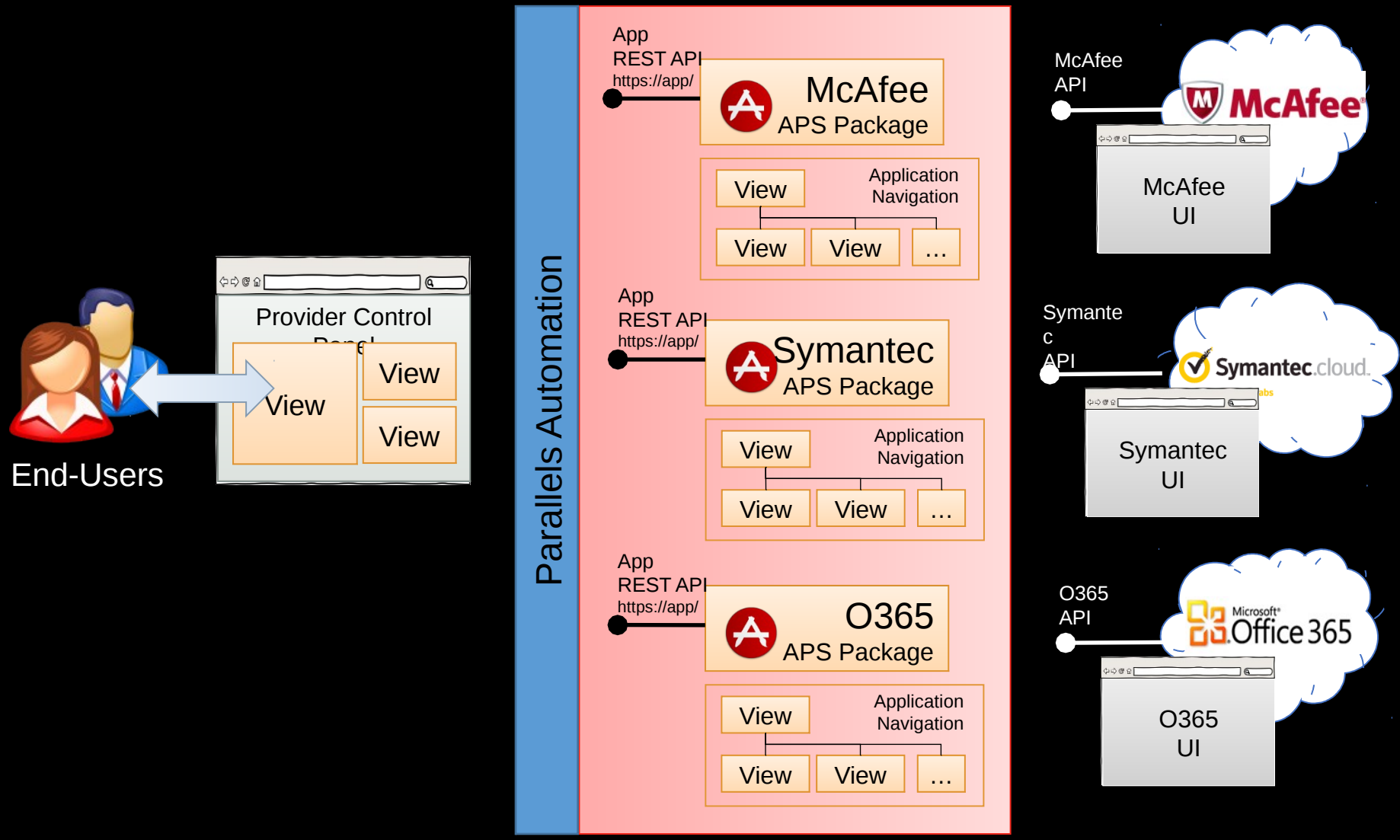


Cross Platform Solutions

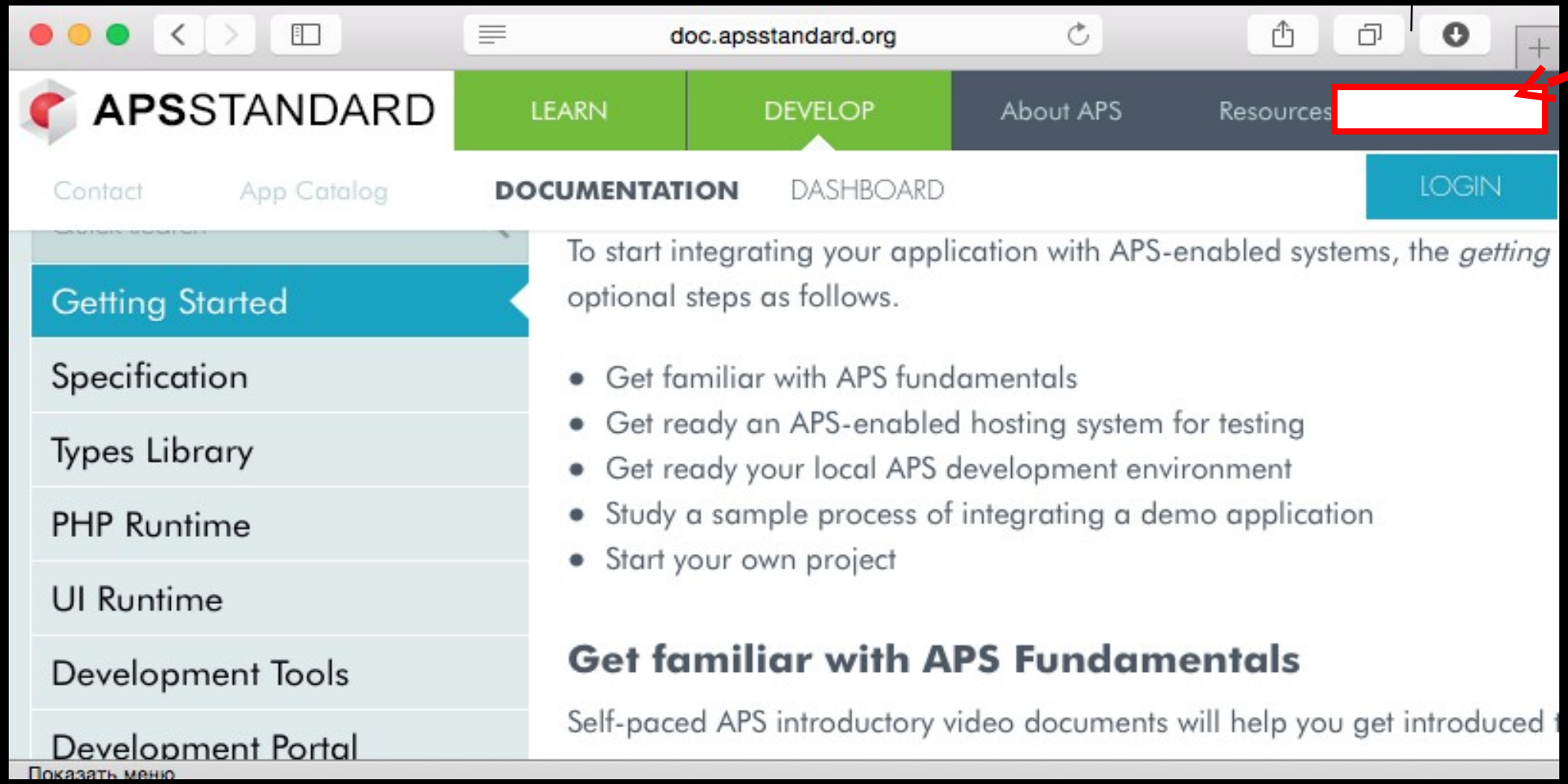
- ✓ Parallels Desktop for Mac
- ✓ Remote desktop access
- ✓ Macs management



What is APS?


End-Users
Provider Control
View
View
View
Parallels Automation
App
REST API
https://app/
McAfee
APS Package
View
Application
Navigation
View
View
…
App
REST API
https://app/
Symantec
APS Package
View
Application
Navigation
View
View
…
App
REST API
https://app/
O365
APS Package
View
Application
Navigation
View
View
…
McAfee
API
McAfee
McAfee
UI
Symante
c
API
Symantec.cloud
Symantec
UI
O365
API
Microsoft
Office 365
O365
UI

Постановка задачи:
Исследование рынка полнотекстовых поисковых движков.
Выбор поискового движка для appstandard.org



Выбор поискового движка. С чего начать?

Доступные альтернативы.

SOLR

Apache Solr — предеставляет все возможности Lucene в простом для использования и быстром поисковом серевере с такими дополнительными функциями, как фасетинг, распределенность и многое другое.

Выбор поискового движка. С чего начать?

Доступные альтернативы.

SOLR

- Rest-like API
- Добавление документов через XML, JSON, CSV или HTTP.
- Запросы с помощи GET и получение результатов в XML, JSON, CSV или бинарном формате.
- Конфигурация в виде файла XML.
- Множество плагинов
- AJAX based admin interface

Выбор поискового движка. С чего начать?

Доступные альтернативы.

ElasticSearch

ElasticSearch - поисковый сервер на основе Lucene. Предоставляет такие возможности, как faceting, распределенность, percolation и многое другое.

Выбор поискового движка. С чего начать?

Доступные альтернативы.

ElasticSearch

- Документно ориентированный. Хранить структуированный JSON.
- Возможность не объявлять схему документа на момент индексации (возможно добавить после).
- RESTful API
- Продвинутая система плагинов.
- Данный добавляются при помощи PUT и POST, получаются GET, удаляются DELETE.

Выбор поискового движка. С чего начать?

Доступные альтернативы.

SphinxSearch

Standalone приложение, написанное на C++, предоставляющее быстрый релевантный полнотекстовой поиска. Тесно интегрируется с SQL базами данных. Предоставляет такие возможности, как faceting, распределенность и многое другое.

Выбор поискового движка. С чего начать?

Доступные альтернативы.

SphinxSearch

- Поддерживает real-time (экспериментально) и offline индексы.
- Может напрямую индексировать данные из SQL БД, XML, TSV.
- Язык запросов SphinxQL — диалект SQL. Для общения со Sphinx можно использовать любой SQL-клиент.
- Нельзя делать частичные обновления offline индекса без полной переиндексации.

Промежуточный итог.

- SphinxSearch - тесно интегрируется и SQL DBs. Обещает наиболее высокую скорость индексации и поиска и низкое потребление ресурсов. Нельзя частично обновлять индекс.
- ElasticSearch и Solr — легко интегрируются и JAVA приложениями, поддерживают частичное обновление индекса, проигрывают SphinxSearch в потреблении ресурсов и скорости, однако обладают большим набором фич.

Как же сделать выбор?

Возможные критерии:

- Features
- Скорость
- Виды источников данных
- Удобство API

Более подробно

Feature	Solr 5.1	ElasticSearch 1.5	Sphinx 2.3.1
API	✔ REST	✔ RESTfull	✔ Proprietary, SphinxQL (SQL dialect)
Client libraries	✔ PHP, Ruby, Perl, Scala, Python, .NET, Javascri	✔ PHP, Ruby, Perl, Scala, Python, .NET, Javascript, Erlang, Clojure	✔ Perl, C#, Haskell, Ruby-on-Rails and other.
3rd-party product integration	✔ Drupal, Magento, Django, ColdFusion, Wordpress etc.	✘ Drupal, Django, Symfony2, Wordpress, CouchBase	✔ Ruby on Rails, Wordpress, Yiiframework, Symfony, Mediawiki, Django etc.
Output	✔ XNL, JSON, CSV, bin	✔ JSON	✔ Via SQL client
Data import	✔ CSV, JSON, XML or DataImportHandler - JDBC, CSV, XML, Tika, URL, Flat File.	✔ JSON or Rivers modules(deprecated)	✔ SQL, XML, tsv datasource

Как же сделать выбор?

Возможные критерии:

- Скорость
- Функциональность
- Источник данных
- Удобство API

Тестируем производительность. Зачем?

- Никто не любит ждать. Выбираем поисковый движок, который работает на наших данных с приемлемой скоростью.
- Настройки поисковых движков сложны. Бенчмарк будет чрезвычайно полезен для выбора оптимальных настроек и отслеживания производительности.

Тестируем!

В качестве фреймворка для тестирования был выбран Basho_bench от Riak:

- Написан на Erlang
- Позволяет добавлять драйвера для своих приложений.
- Мощная конфигурируемость сценариев без необходимости пересборки приложения.

Отрывок конфигурационного файла.

```
{mode, max}.

{duration, 1}.

{concurrent, 1}.

{report_interval, 5}.

{driver, basho_bench_driver_sphinxsearch}.


{host, "0"}.

{port, 9306}.

{user, "test"}.

{password, ""}.

{generators, [
    {text_generator,
      {value_generator,
        {function, generators, text_generator, [[{path, "../../dataset/dicti
    {word_generator,
      {value_generator,
        {function, generators, word_generator, [[{path, "../../dataset/dicti
    {fh_exist_id_generator,
```

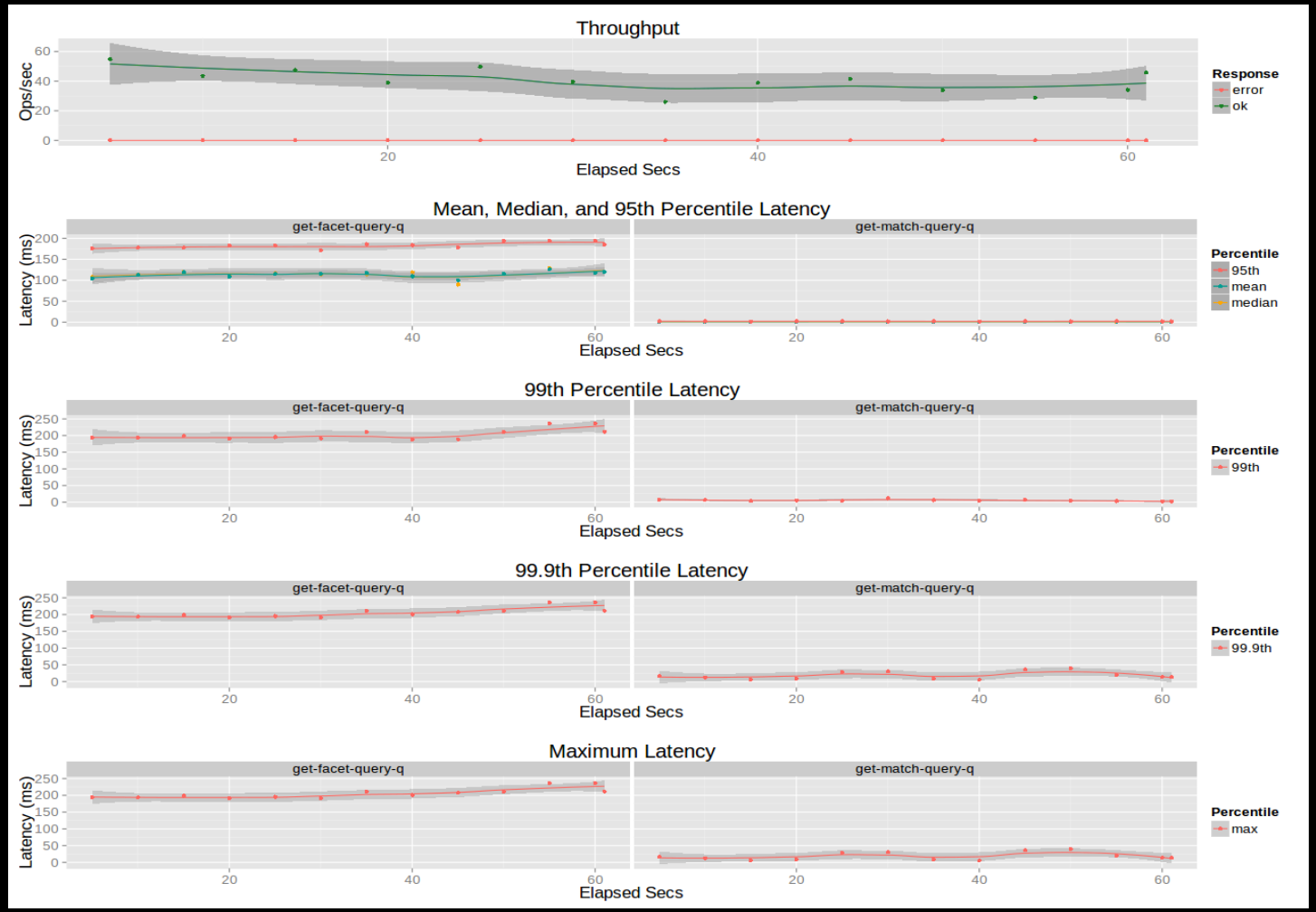
```
        {function, generators, word_generator, [[{path, "../../dataset/dict
    {fh_exist_id_generator,
      {value_generator,
        {function, generators, uniform_int_generator, [[{start, 0}, {stop,
    {sh_exist_id_generator,
      {value_generator,
        {function, generators, uniform_int_generator, [[{start, 250000}, {s
  ]}.

{queries, [
  {show_meta, {"show meta", []}},
  {range_query_q, {"select * from wiki", []}},
  {range_query_rt_q, {"select * from rt_wiki", []}},
  {match_query_q, {"select * from wiki_idx where match('@text ~s')", [word_
  {match_query_rt_q, {"select * from rt_wiki where match('@text ~s') option
  {facet_query_q, {"select * from wiki where id between ~p  and ~p facet use
  ]}.

{code_paths, ["./deps/emysql/ebin"]}.

{operations, [
    {{get, facet_query_q}, 1},
    {{get, range_query_q}, 1},
    {{get, range_query_rt_q}, 1},
    {{get, match_query_q}, 1}
]}.
```

Тестируем!

Конфигурация:

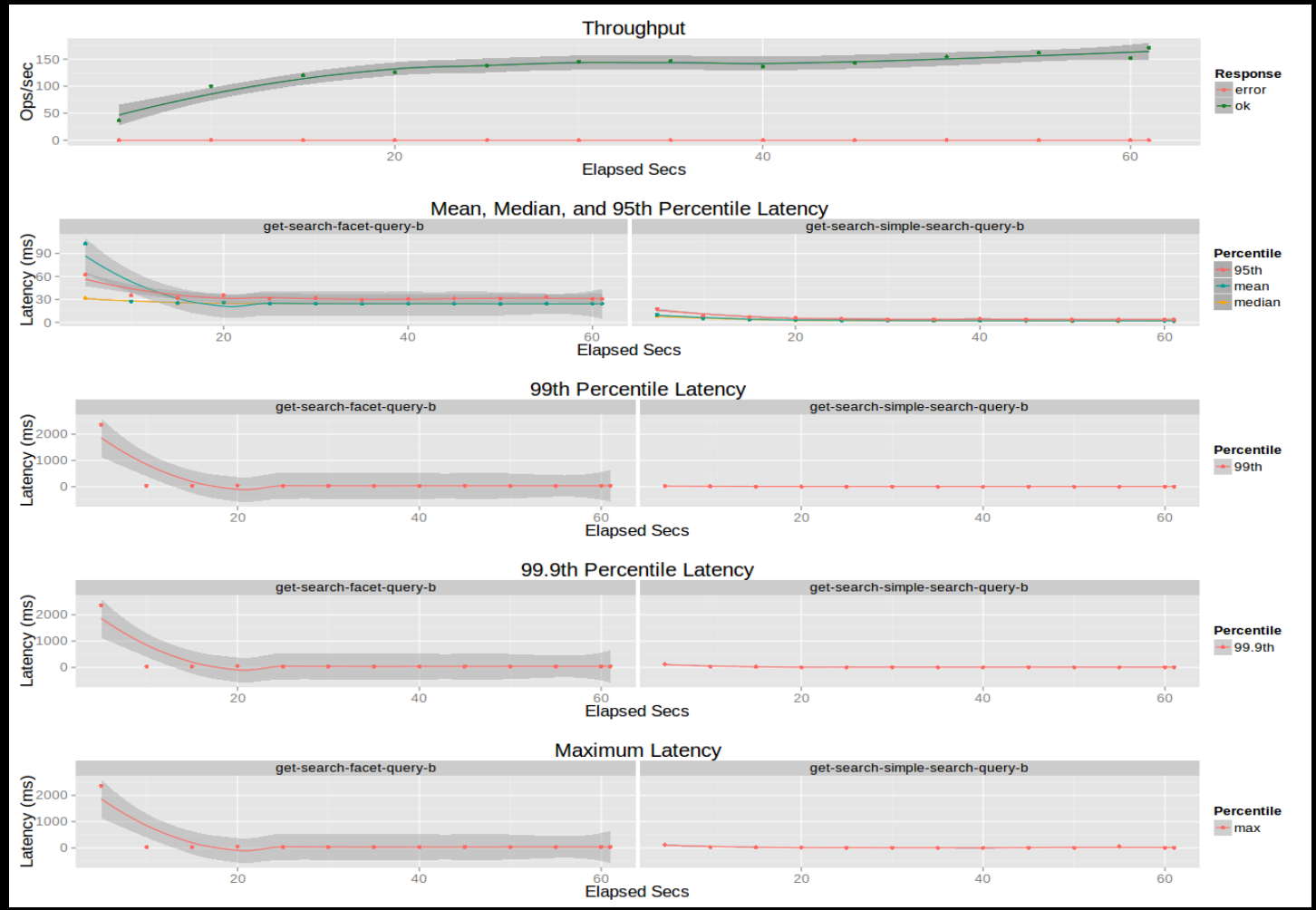
- Источник данных: часть xml дампа wikipedia.org. Размер 7 GB. 500000 документов. Схема примерно та же, что и у реальных данных.
- Настройки движков «похожие».
- CPU: Intel Core i-5 2400m, Storage SSD, 6 GB Ram.

Результаты тестирования.
SphinxSearch (4 match / 1 facet)

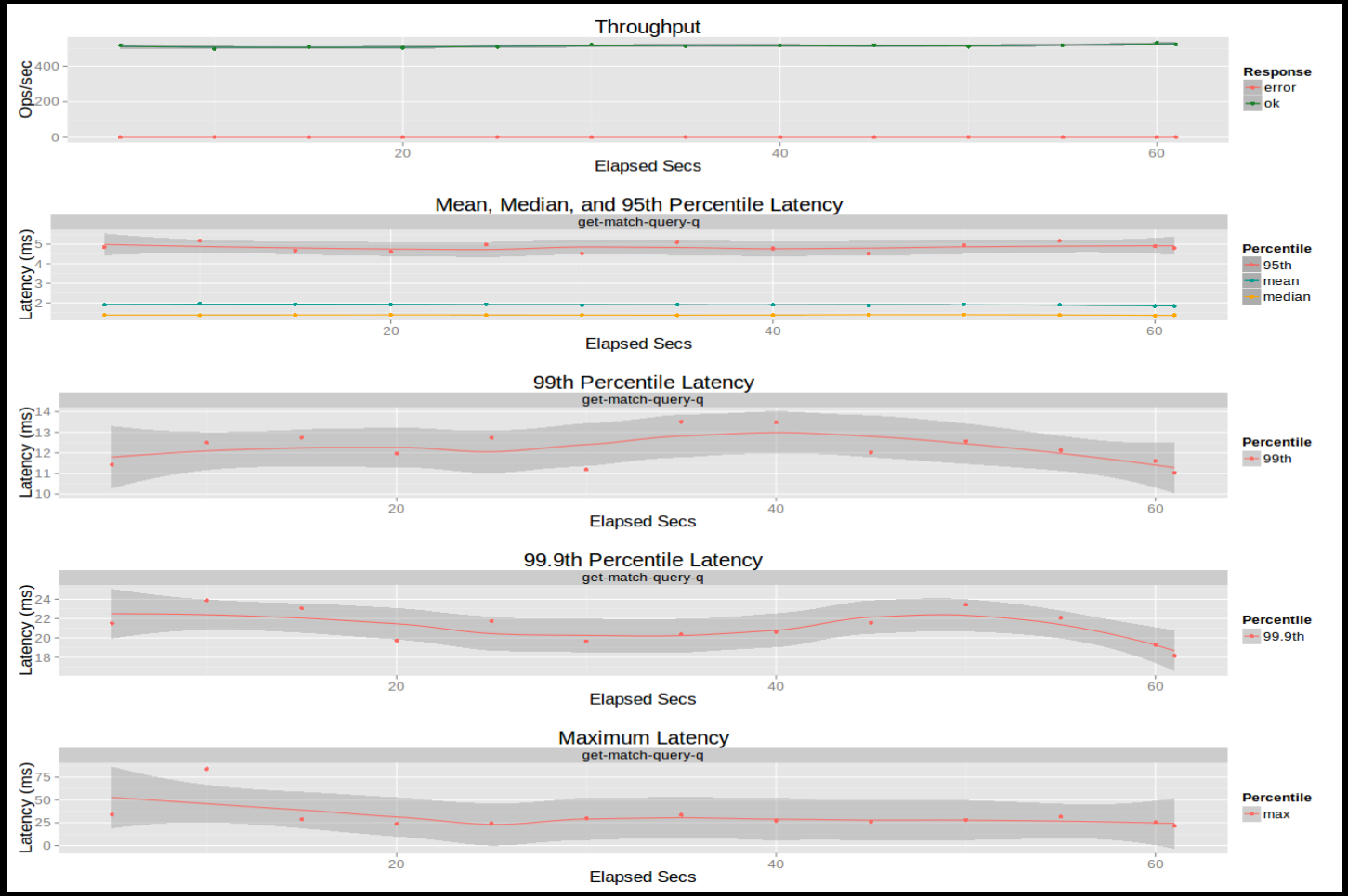


Parallels

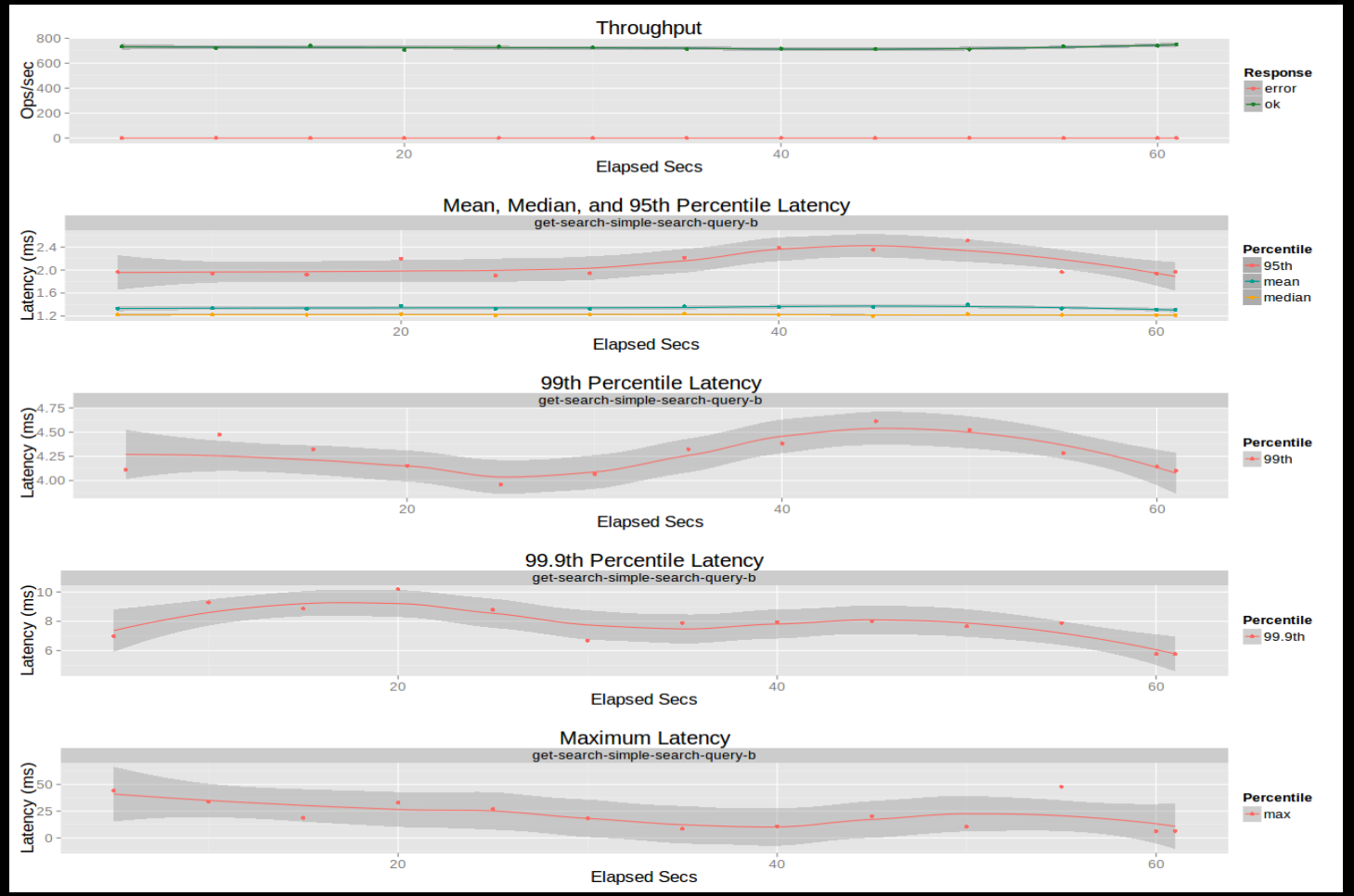
Результаты тестирования.
ElasticSearch (Solr): (4 match / facet)



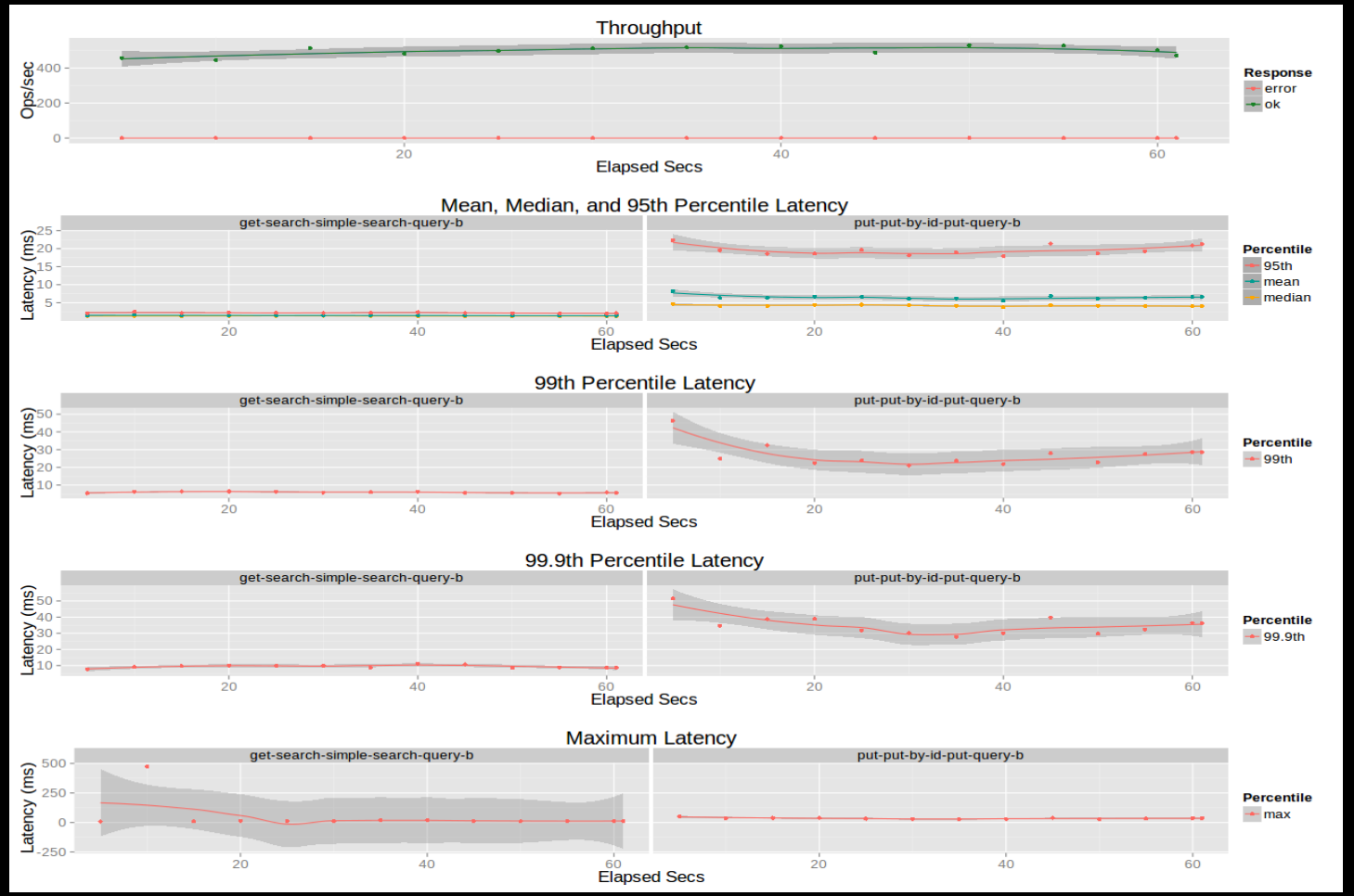
Результаты тестирования.
SphinxSearch (match)



Результаты тестирования.
ElasticSearch (Solr) (match)



Результаты тестирования.
ElasticSearch (Solr) (9 match / 1put)



Обсуждение

- Lucene-based движки потребляют на порядок больше оперативной памяти. Загруженность процессора во время поиска была примерно одинаковая.

Обсуждение

- SphinxSearch в первом тесте (match + facet) работает медленнее 2.5 раз, из-за того, что ES и Solr имеют более продвинутое внутреннее кэширование запросов.
- Во втором тесте (match) ситуация аналогичная.

Обсуждение

- В третьем тесте (9 match / 1 put) участвовал только ES и показал результаты, примерно сравнимые с результатами для SphinxSearch из второго теста.
  Вывод: ES выигрывает у SphinxSearch в первых двух тестах благодаря кэшу запросов, несмотря на случайную генерацию последних.

Замечание

В зависимости от настроек окружения и поисковых движков результаты могут существенно меняться, что дополнительно показывает важность отслеживания таких изменений с помощью нагрузочного тестирования.

Итог

Для app-standart.org оптимальным выбором оказался является Elasticsearch на основе следующих критериев:

- Достаточные ресурсы окружения.
- Сценарий предполагает больше чтений, нежели записей.
- Возможность кластеризации, real-time обновлений документов, RestFul API.
- Замечательная документация (в сравнении с Solr особенно) и широкое сообщество.

Parallels

Используемые технологии.

Lesson learning

- Бенчмаркинг - дело хитрое.
- Поисковые сервера - это весело.
- Функциональные языки программирования с динамической типизацией — довольно своеобразно, но с этим можно жить.

Исходный код бенчмарка:

https://github.com/igrocki/search-engine-benchmarks-application

Спасибо за внимание!