دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گزارش کارآموزی (هفته چهارم)

محل کارآموزی: شرکت سامانه گستر سحاب‌پرداز

نام استاد کارآموزی

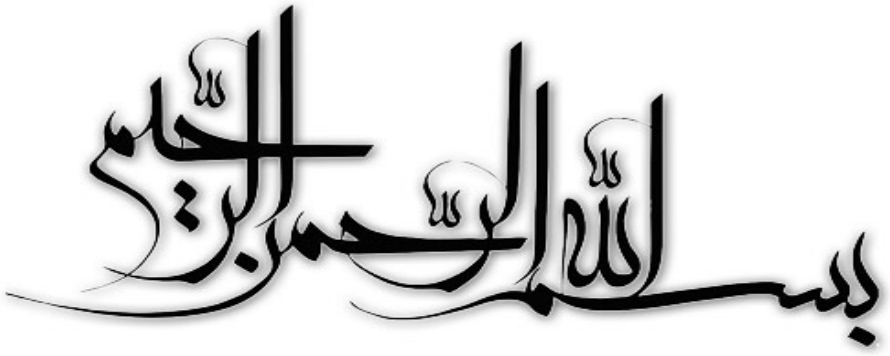
دکتر مسعود صبائی

نام دانشجو

امیرمحمد پیرحسینلو

۹۵۳۱۰۱۴

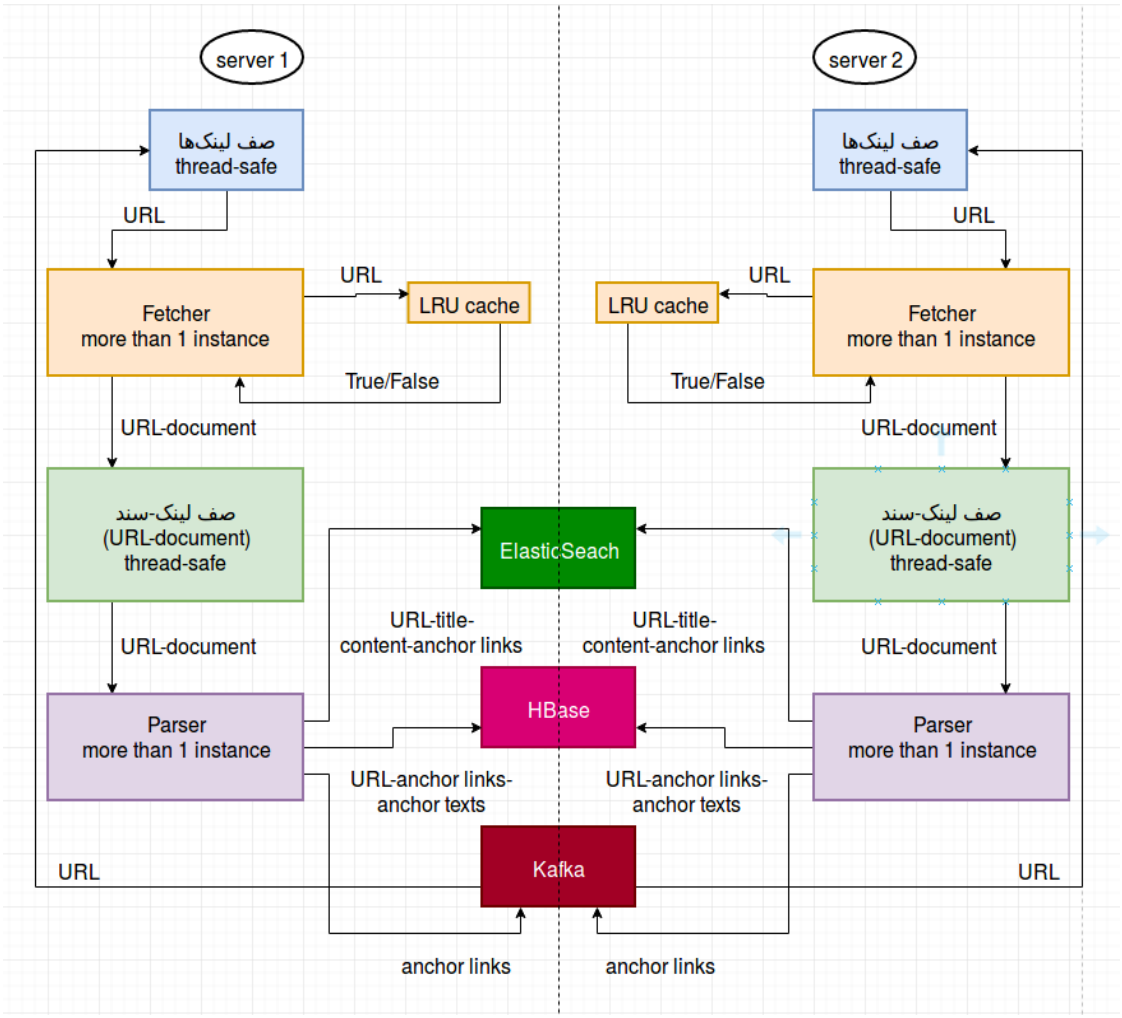
تابستان ۱۳۹۸


بسم الله الرحمن الرحيم

فهرست مطالب

فهرست اشکال

فصل اول
معماری موتور جستجو



شکل ۱-۱: معماری نرم‌افزار

۱-۱ مقدمه

در این هفته به طراحی معماری موتور جستجو پرداختیم. همچنین برنامه‌نویسی و پیاده‌سازی بخش‌های مستقل از معماری مانند بخش ارتباط با پایگاه‌داده‌ها انجام شد.

ابتدا هر کدام از افراد تیم معماری مدنظر خود را ارائه داد و در نهایت با ادغام طرح‌های موجود یک معماری جامع انتخاب شد. هر کدام از افراد تیم برنامه‌نویسی و پیاده‌سازی یک بخش از این معماری را بر عهده گرفتند. پیاده‌سازی بخش **HBase** به من واگذار شد.

در ادامه اجزای این معماری معرفی و توضیح داده می‌شود.

۱-۲ اجزای موتور جستجو

شکل ۱-۱ نمای کلی از معماری موتور جستجو و یک معماری توزیع‌شده **(distributed)** را نشان می‌دهد. دو سرور داریم که دقیقا عین هم هستند و پایگاه داده‌های آن‌ها به هم متصل است. طبیعتا پایگاه داده‌ها مثل **Hbase** و موتور جستجوی **ElasticSearch** و پلتفرم **Kafka** را در حالت توزیع شده نصب و اجرا کرده‌ایم.

حال به شرح اجزا در شکل ۱-۱ می‌پردازیم:

۱-۲-۱ صف لینک‌ها

این صف شامل لینک‌ها (**URL**[۱]) هایی است که باید بازدید شده و محتوای آن‌ها مورد پردازش قرار گیرند. این صف در ابتدا با **URL** های زیر به عنوان مقادیر اولیه (**seed** اولیه) پر می‌شود:

- https://en.wikipedia.org/wiki/Main_Page
- https://us.yahoo.com
- https://www.nytimes.com/
- https://www.msn.com/en-us/news
- http://www.telegraph.co.uk/news/
- http://www.alexa.com
- http://www.apache.org
- https://en.wikipedia.org/wiki/Main_Page/World_war_II
- http://www.news.google.com
- https://www.geeksforgeeks.org
- https://mvnrepository.com/

به دلیل این‌که **object** های زیادی از این صف داده می‌خوانند، این صف باید بین نخ ها امن [۲] باشد.

۱-۲-۲ Fetcher

object ای است که به طور متناوب از صف لینک‌ها (در قسمت قبل توضیح داده شد.) **URL** دریافت می‌کند سپس محتوای آن را دانلود می‌کند و **URL** و محتوا را در صف سندها قرار می‌دهد.

پیش از دانلود محتوای یک لینک بررسی می‌شود که آیا درخواست به **host** این لینک مجاز است یا خیر. در صورت مجاز نبودن درخواست، لینک در انتهای صف لینک‌ها قرار می‌گیرد. حال سوالی که مطرح می‌شود این است که معیار مجاز بودن چیست؟

اگر در ۳۰ ثانیه اخیر به یک **host** درخواست داده باشیم، دیگر مجاز نیستیم که به آن درخواست دهیم زیرا ممکن است آدرس **IP** ما را مسدود **(block)** کند (ممکن است فکر کند برنامه ما یک ربات مولد درخواست برای ایجاد حمله **DOS**[۳] است.). برای بررسی این وضعیت از یک حافظه از نوع **LRU**[۴] استفاده می‌کنیم.

چندین نمونه از **Fetcher** در برنامه اجرا می‌شوند تا سرعت پردازش و دریافت لینک‌ها افزایش یابد. البته تعداد آن‌ها به مواردی مانند

[۱] **Uniform Resource Locator**
[۲] **Thread-safe**
[۳] **Denial Of Servise**
[۴] **Least Recently Used**

منابع سیستم (میزان **RAM** ، قدرت **CPU** ، میزان حجم دیسک ذخیره سازی جانبی (**HDD**[۵]))، تعداد سایر نخ‌ها (**thread** ها) و ... بستگی دارد.

۱-۲-۳ صف اسناد

این صف شامل جفت **(pair)** های لینک-سند **(url-document)** است که باید محتوای بخش سند این جفت‌ها تجزیه **(parse)** شود و اطلاعات زیر از آن‌ها استخراج شود:

- عنوان سند **(title)**
- محتوای پاراگراف‌ها یا برچسب **(tag)** های <p>
- **anchor link** ها و عنوان‌های هر کدام

به دلیل این‌که **object** های زیادی از این صف داده می‌خوانند، این صف باید امن [۶] باشد.

۱-۲-۴ Parser

object های **Parser** وظیفه دارند به صورت تناوبی از صف اسناد جفت‌های لینک-سند را خوانده و اطلاعاتی که در بخش صف اسناد توضیح داده شد را استخراج کنند. پس از استخراج عملیات‌های زیر انجام می‌شود:

- قرار دادن لینک سند، محتوای پاراگراف‌های سند، عنوان سند **(title)** و **anchor link** ها در **ElasticSearch**
- قرار دادن لینک سند، **anchor link** ها و عنوان‌های هر کدام در **HBase**
- تحویل **anchor link** ها به **Kafka**

چندین پروسه **Parser** در برنامه داریم تا سرعت استخراج اطلاعات افزایش یابد. البته تعداد آن‌ها به مواردی مانند منابع سیستم (میزان **RAM** ، قدرت **CPU** ، میزان حجم دیسک ذخیره سازی جانبی (**HDD**[۷]))، تعداد سایر نخ‌ها (**thread** ها) و ... بستگی دارد.

۱-۲-۵ Kafka

Kafka مدیریت صف لینک‌ها را بر عهده دارد. **anchor link** ها را از **Parser** گرفته و آن‌هایی که تاکنون مشاهده نشده‌اند را در انتهای صف لینک‌ها قرار می‌دهد.
این نوع معماری پردازش **BFS**[۸] برای گراف وب را فراهم می‌کند.

۱-۳ نتیجه گیری

در این هفته معماری نرم‌افزار طراحی شد و بخشی از کار برنامه‌نویسی و پیاده‌سازی انجام شد. در هفته‌های آتی برنامه‌نویسی و پیاده‌سازی به طور کامل انجام خواهد شد.

---

[۵] **Hard Disk Drive**
[۶] **Thread-safe**
[۷] **Hard Disk Drive**
[۸] **Breadth First Search**

واژه‌نامه‌ی انگلیسی به فارسی

B

block . . . . . مسدود کردن

D

distributed . . . . . توزیع شده

document . . . . . سند

F

fetcher . . . . . دریافت کننده

H

host . . . . . میزبان

O

منظور یک **instance** از یک **class** در زبان جاوا است.

object

P

pair . . . . . جفت، دوتایی

parser . . . . . تجزیه کننده

T

در اینجا منظور برچسب های فایل **HTML** است مانند تگ <**p**>

tag . . . . . .

title . . . . . عنوان