

Gungho and Cloud Computing: A scalable crawling and processing framework

YAPC Asia 2008

Jeff Kim <jkim@chosec.com>



Crawler?

- ▶ A crawler is an automated script which, from a seed list of URLs visits each, one and finds new links and follows them.
- ▶ Typically used to either extract specific information or to index pages for search engine purposes.
- ▶ There are open crawls and closed crawls.
- ▶ Some crawlers play nice, others not so nice.



Crawler ≠ Scraper

- ▶ People often confuse the two since they both programmatically fetch and process web pages.
- ▶ Crawlers are more general and operate on a larger set of sites.
- ▶ Scraping usually is about getting a specific set of data from a fixed list of sites.
- ▶ For scraping:
 - ▶ pQuery
 - ▶ LWP
 - ▶ WWW::Mechanize



Why Crawling Sucks

- ▶ Linear
- ▶ Speed limits
 - ▶ Target site
 - ▶ Crawling box's bandwidth
- ▶ Must play nice!
 - ▶ robots.txt
 - ▶ Meaningful user agent description
 - ▶ Throttling
- ▶ I/O
 - ▶ Storage
 - ▶ Intensive reads / writes



Project Overview

- ▶ **Givens**

- ▶ Millions of domains in seed list

- ▶ **Constraints**

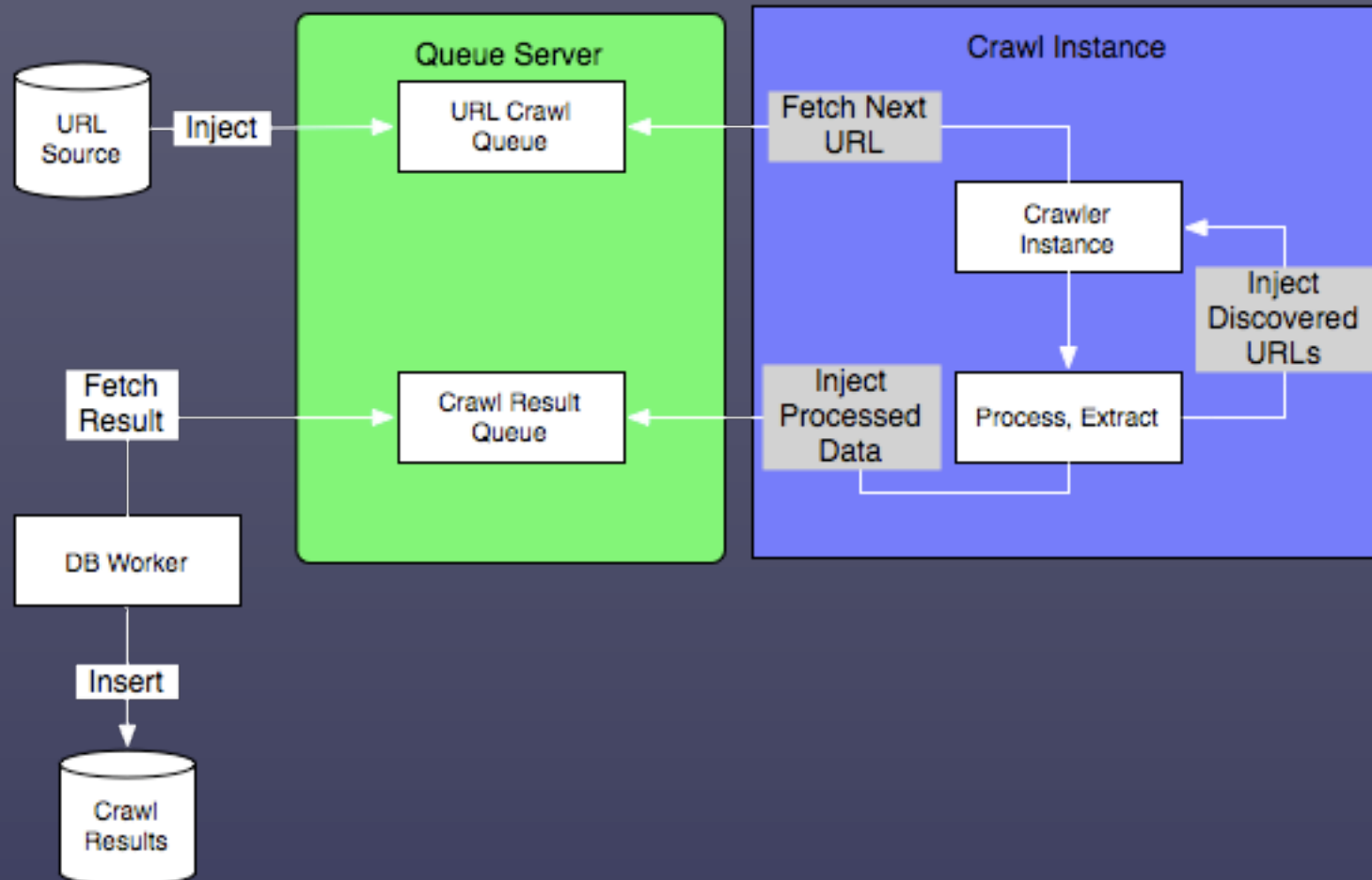
- ▶ Budget was tight
- ▶ Timeframe for execution was tight

- ▶ **Goals**

- ▶ Well behaved crawler
- ▶ Store all the raw HTML crawled
- ▶ Crawl no more than 20 pages per domain



Overall Architecture



Gungho?



What is this? Huh?



Gungho!

- ▶ Written by Daisuke Maki
- ▶ High performance crawling framework
- ▶ Asynchronous HTTP Requests
- ▶ Asynchronous DNS Lookups
- ▶ Automatic robots.txt handling
- ▶ Throttling friendly
- ▶ Very componentized (easy to extend!)



Why Gungho Rocks

- ▶ Linear → Async
- ▶ Speed limits → Async
- ▶ Must play nice! → Very Easily Handled

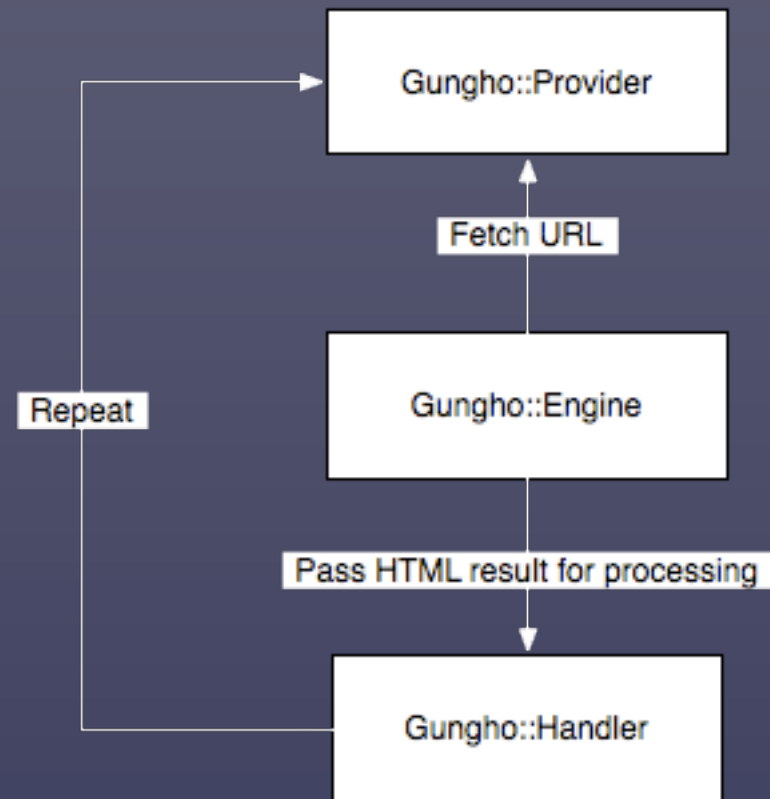


Gungho Basics

- 1 Provider (hands URLs to Gungho)
- 2 Engine (actual mechanism to fetch the URL)
- 3 Handler (what to do with the content / response that comes back)



Gungho General Flow



Gungho Provider

- ▶ Extended Gungho::Provider
- ▶ Pull a URL off of the URL processing queue
- ▶ Maintain count of number of pages we've discovered and processed from this base URL
- ▶ If the number of pages exceeds our per-URL cap, then back out and fetch new URL
- ▶ Otherwise hand over to the Engine to fetch



Gungho Engine

- ▶ `Gungho::Engine::POE`
- ▶ Spawns `POE::Component::Client::HTTP` sessions
- ▶ Can configure number of redirects to follow before bailing
- ▶ Number of clients to spawn initially
- ▶ Amount of time you want to delay before fetching another URL
- ▶ Tie in `Gungho::Component::Throttle::Simple`
 - ▶ Helps the crawler not try and process *too* many requests at once



Gungho Handler

- ▶ Extended Gungho::Handler
- ▶ Processing rules for HTML returned from the Engine
- ▶ Inject processed data into Job Processing Queue
- ▶ Store raw HTML on S3
- ▶ Tied in with GunghoX::FollowLinks
 - ▶ Discovery of new URLs to process within HTML returned
 - ▶ Definition of rules to pick which links are important and should be followed
 - ▶ Pushes discovered links back into Provider



Messaging / Job Queue

- ▶ POE::Component::MessageQueue
 - ▶ Streaming Text Oriented Messaging Protocol (STOMP) based
 - ▶ Pluggable Storage Engine
 - ▶ Messaging queue for URLs to feed into the crawler

- ▶ Swarmage
 - ▶ Built upon POE::Component::MessageQueue
 - ▶ Distributed Job Queue
 - ▶ For writing the crawl results into



EC2

- ▶ Amazon's Elastic Compute Cloud
- ▶ Xen-based virtualized customizable instances
- ▶ Geographically Split up (somewhat)
- ▶ Pay by usage-hour and bandwidth used
- ▶ Currently in open Beta



EC2

- ▶ Amazon's **Elastic** Compute Cloud
- ▶ Xen-based virtualized customizable instances
- ▶ Geographically Split up (somewhat)
- ▶ Pay by usage-hour and bandwidth used
- ▶ Currently in open Beta



Integrating EC2

- ▶ Create Xen-based images:
 - ▶ Gungho crawler
 - ▶ Message Queue
 - ▶ Database
- ▶ Setup startup / shutdown scripts
- ▶ Start / stop Gungho instances as needed
- ▶ Store raw HTML on S3
 - ▶ Transfers between EC2 and S3 are quick and free

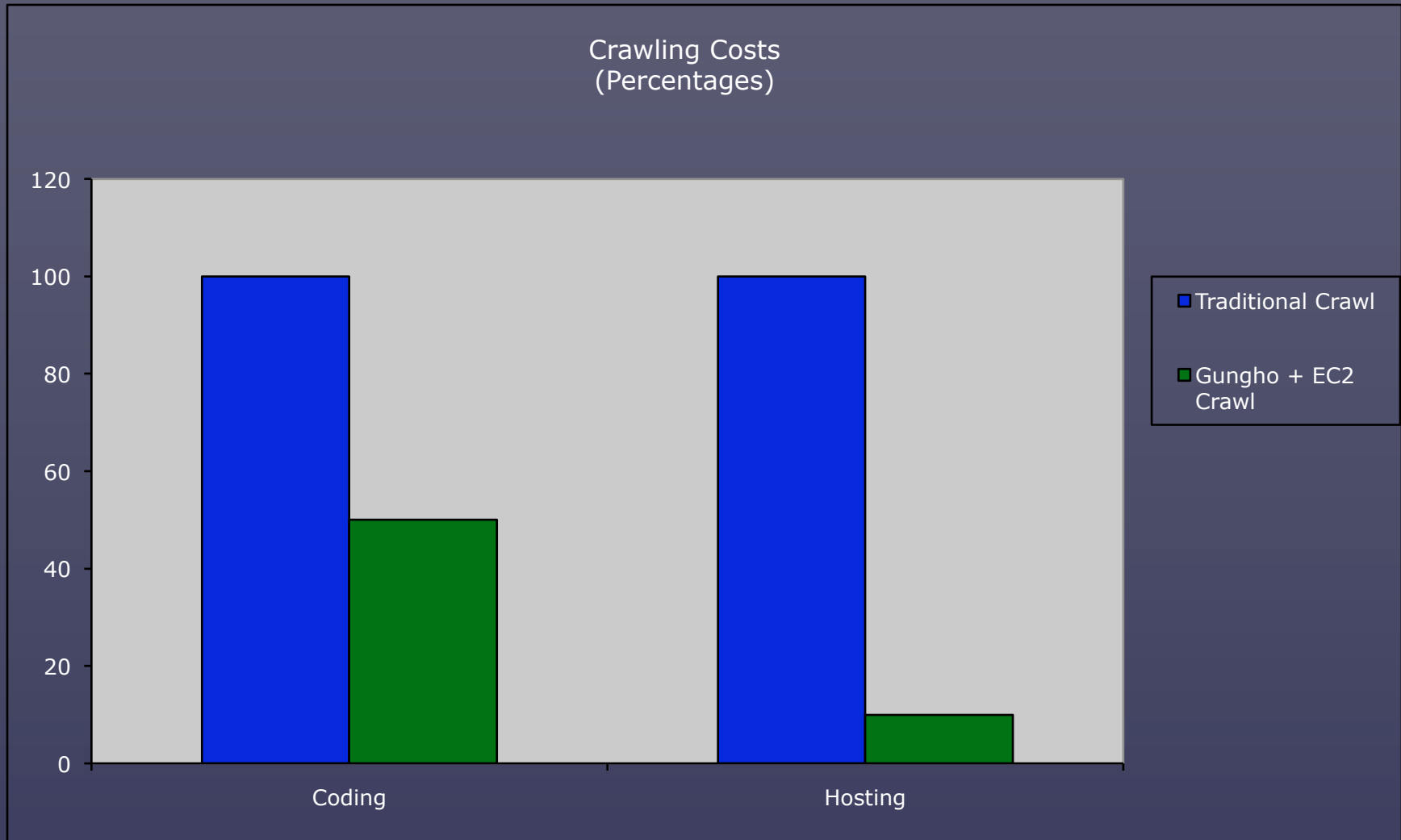


Physical Configuration

- ▶ Queuing / Messaging server
 - ▶ 1 - EC2 X-Large instance (16GB Ram, 4x2 virtual cores)
- ▶ Database server
 - ▶ 1 – EC2 X-Large instance (16GB Ram, 4x2 virtual cores)
- ▶ Gungho Crawlers
 - ▶ 20 – EC2 Small instances (1.7GB Ram, 1 virtual core)
 - ▶ 2 Gungho crawl instances per EC2 instance



Project Costs



Results

- ▶ Millions of sites crawled
- ▶ Because of the elastic nature of EC2, crawl finished ahead of schedule
- ▶ Crawler was respectful
- ▶ No complaints from any sites crawled



Questions?

質問?

