CiteSeer^{x}: A Cloud Perspective

Pradeep Teregowda, Bhuvan Urgaonkar, C. Lee Giles
Pennsylvania State University

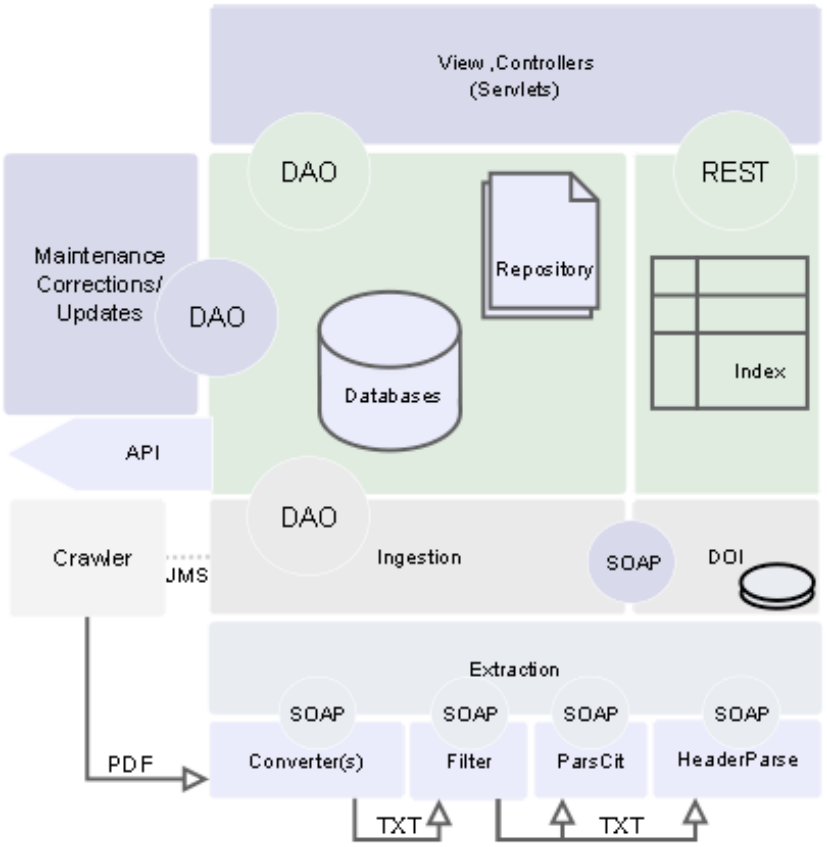
Problem Definition

- **Question:** How to effectively move a digital library, CiteSeer^x, into the cloud
  - Which sections, components, or subset of CiteSeer^x could be most cost effective to move?
- Our contribution – analysis from an economic perspective.
  - Solve by decomposing the application across
    - Components
    - Content
    - Peak load hosting

SeerSuite - CiteSeer^{x}

- SeerSuite
  - Framework for digital libraries
    - Flexible, Scalable, Robust, Portable, state of the art machine learning extractors, open source – use.
  - CiteSeer^{x}
    - Instance/Application of SeerSuite.
    - Collection of
      - > 1.6 million documents
      - > 30 million citations
    - Approximately 2 million hits per day

SeerSuite Architecture



- Web Application
- Focused Crawler
- Document Conversion and Extraction
- Document Ingestion
- Data Storage
- Maintenance Services
- Federated Services

Hosting models

- Component hosting
  - SeerSuite is modular by design and architecture, host individual components across available infrastructure.
- Content hosting
  - CiteSeerx provides access to document metadata, copies and application content
  - Host parts or complete set.
- Peak load loading
  - Support the application during peak loads
  - Support growth of traffic.

Component Hosting

- SeerSuite/CiteSeer^x is modular by design, composed of services which can be hosted in the cloud.
  - Expense of hosting the whole of CiteSeer^x is prohibitive.
  - Solution: Host a component or service i.e.,
    - Component/service code
    - Data on which the component acts
    - Interfaces, etc. associated with the component
  - Goal: Identify optimal subset/components.

Component Hosting - Costs

Component	Amazon EC2		Google App Engine	
	Initial	Monthly Costs	Initial	Monthly Costs
Web Services	0	1448.18	0	942.53
Repository	0	1011.88	163.8	593.21
Database	0	858.89	12	348.05
Index	0	527.08	3.1	83.48
Extraction	0	499.02	0	90.6
Crawler	0	513.4	0	105

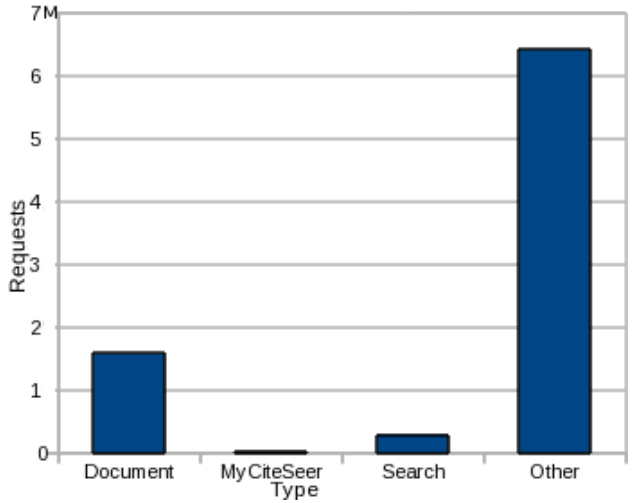
Most expensive - host web services.

Component Hosting – Lessons Learned

- Hosting components is reasonable
  - Having a service oriented architecture helps
- Amazon EC2
  - Computation costs dominate.
- Google App Engine
  - Refactoring costs ?
- Refactoring required not just for component, but other services.
- Storage and transfer costs maybe optimized
  - A study of data transfer in the application gives insights to costs.
- Approach suitable for meeting fixed budgets
  - How many components of an application can be hosted for a fixed budget.

Content Hosting

- Approach: Identify specific content
  - Static Web Application content
    - Javascript
    - Stylesheets
    - Images/Graphs.
  - Repository content
    - PDF files
      - Current Size: 1 terabyte
  - Database content
    - Partition database
      - Current size: 120 gigabytes



Analysis of Content Hosting

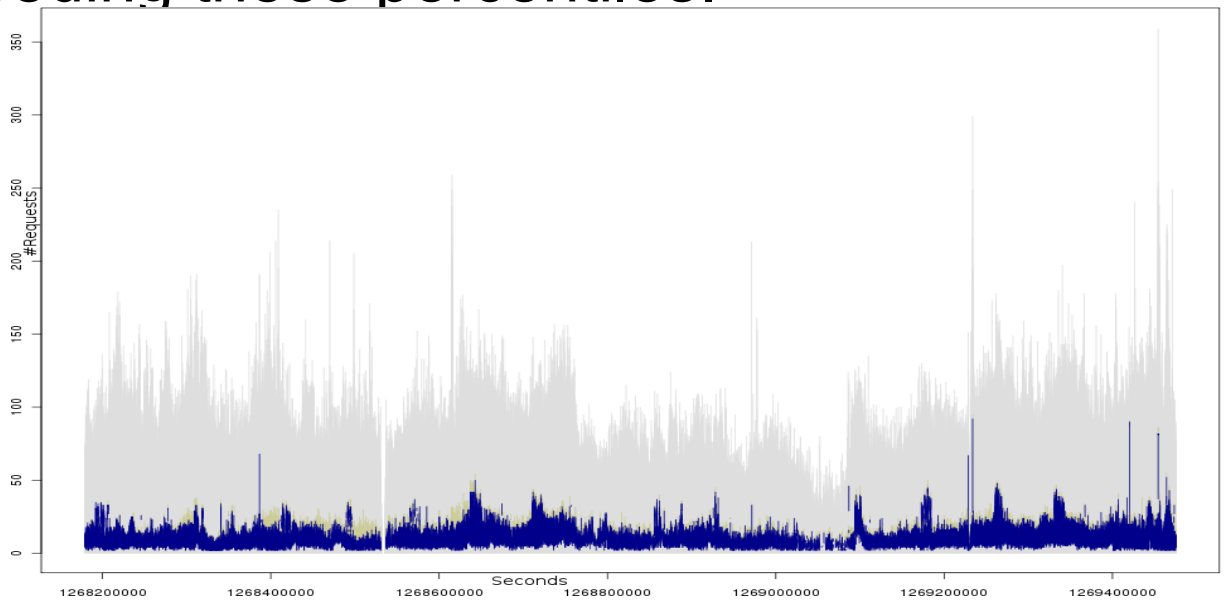
- Examining the traffic (requests) at peak loads.
  - Requests for stylesheets, images, javascript account for most of the requests.
- The size of these files is 2.2 MB
- Since these files are embedded in almost every web page, bandwidth consumed 390.3 GB.
- Costs < 142 dollars.
- Simpler to deploy
  - Move files to the cloud, update references to them in the presentation layer.

Content Hosting – Lessons Learned

- Hosting specific content relevant to peak load scenarios
  - Easy to do – minimal refactoring required, affects a minimal set of components (presentation layer).
- More complex scenarios need to be examined
  - Hosting papers from the repository
  - Hosting shards of the index
  - Database

Peak Load Hosting

- Part of the load can be handled by an instance hosted in the cloud
- Approach
  - Look at various percentiles of the load (90%)
  - Consider utilizing the cloud instance only at loads exceeding these percentiles.



Peak Load Hosting - Costs

Costs		Quantity	Amazon	Google
Initial Setup	Data In	1820.4 GB	0	182
Monthy	Stored	1820.4 GB	182.4	273.06
	Data In	14.78 GB	0	1.48
	Data Out	298.7 GB	44.8	35.84
	Transaction	368 TPS	9.27	0
	CPU	70 HRS	285.6	7
Total (Montly)			521.7	317.38

CPU and Data Transfer costs dominate.

Peak Load – Lessons Learned

- Hosting only during peak load conditions is economically feasible.
- Growth potential
  - Can be used to handle growth in traffic, instead of procuring new hardware.
  - Hosting a specific component under stress; such as a database
    - In such a case it will cost 385 dollars to host the database in Amazon EC2.

Conclusions

- SeerSuite/CiteSeer^x and different approaches were proposed for hosting CiteSeer^x .
- Investigated cost of hosting for
  - Component
    - Economically reasonable
    - Refactoring costs
  - Content
    - Simplest approach
    - More complex scenarios require deeper study
  - Peak load
    - Very reasonable
    - Support for growth and scalability.

Future Work

- Cost of refactoring – particularly for Google App Engine.
- Cost comparisons for other cloud offerings – Azure, Eucalyptus.
- Privacy and user issues – myCiteSeer and private clouds.
- Technical issues with cross hosting – load balancing, latency needed to be addressed.
- Virtualization in SeerSuite, components built with cloud hosting in mind (Federated Services).

Q & A

Appendix

Assumptions

Instance sizes are larger then expected load (15% average usage for current infrastructure).

Instances include libraries and or allow these libraries to be included.

Maintenance traffic is not accounted (< %1).

Effort required to maintain – extra personnel costs are not included (Assumed to be the same as existing).

Naïve clustering and load balancing.

DB		Amazon	Google	Initial	REP		Amazon	Google	
Stored	120	12	18	12	Stored	1638.4	163.84	245.76	163.84
Data In	0	0	0		Data In	30	0	3	
Data Out	2150.4	322.56	258.05		Data Out	2270.4	340.56	272.45	
Transactions	134	34.73	0		Transactions	69	17.88	0	
CPU		489.6	72		CPU		489.6	72	
		858.89	348.05				1011.88	593.21	

INDEX		Amazon	Google		WS		Amazon	Google	
Stored	32	3.2	4.8	3.2	Stored	30	3	4.5	3
Data In	2	0	0.2		Data In	4253.9	0	425.39	
Data Out	54	8.1	6.48		Data Out	3072	460.8	368.64	
Transactions	101	26.18	0		Transactions	20	5.18	0	
CPU		489.6	72		CPU		489.6	72	
		527.08	83.48				1448.18	942.53	

EX		Amazon	Google		CR		Amazon	Google	
Stored	0	0	0	0	Stored	0	0	0	0
Data In	150	0	15		Data In	150	0	15	
Data Out	30	4.5	3.6		Data Out	150	22.5	18	
Transactions	19	4.92	0		Transactions	5	1.30	0	
CPU		489.6	72		CPU		489.6	72	
		499.02	90.6				513.40	105	

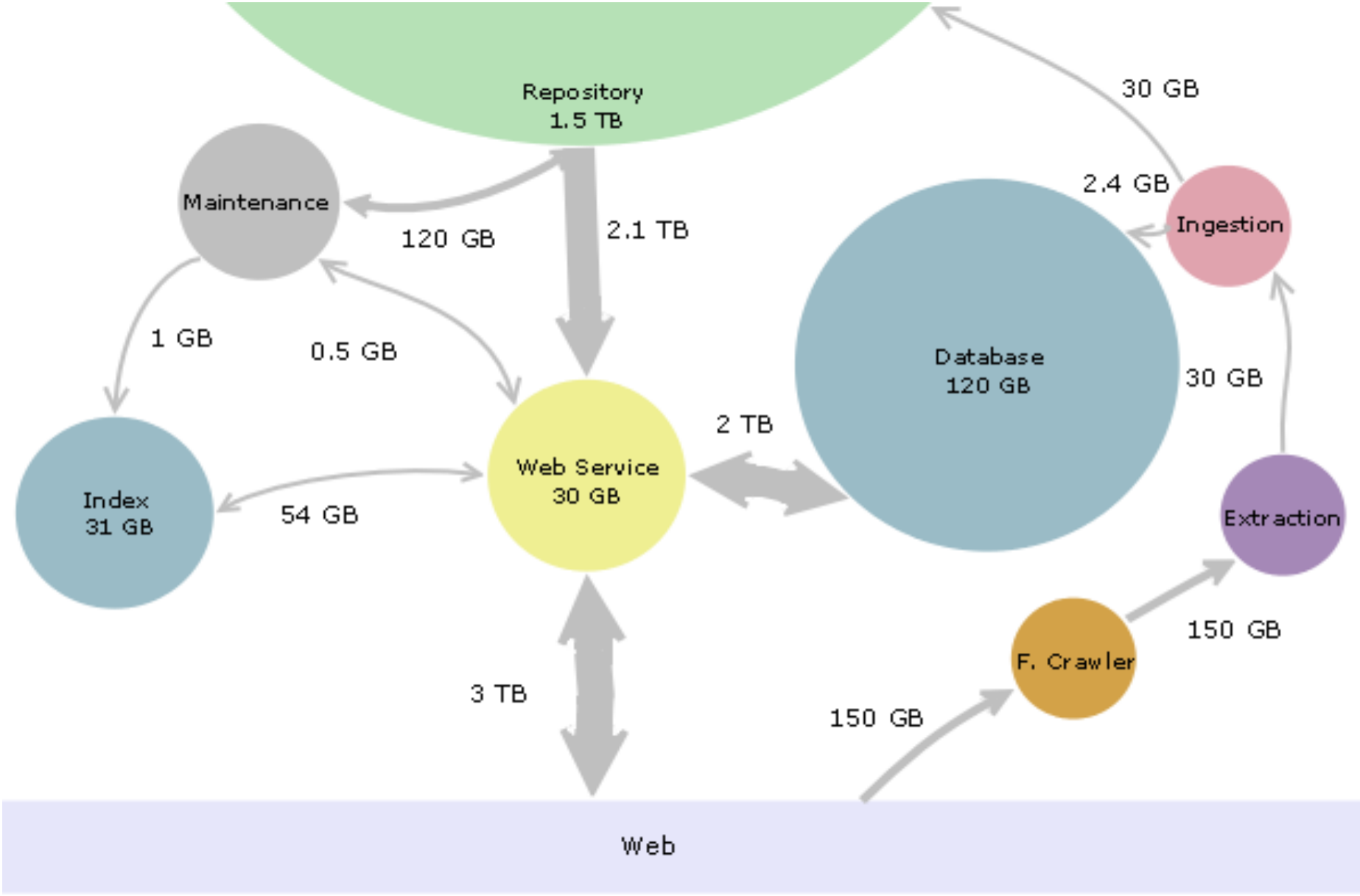
SeerSuite Architecture

- Web Application
  - User interaction, supports various interfaces.
  - Built using the java Spring framework.
- Focused Crawler
  - Acquire documents from the web specific to a particular topic
- Document Conversion and Extraction
  - Process acquired documents to enable ingestion into the collection.
- Document Ingestion
  - Add processed documents to the collection.

SeerSuite Architecture

- Data Storage
  - Store acquired documents – persistence, faster access and use.
- Maintenance Services
  - Processes, which help maintain freshness – statistics, index, graphs.
- Federated Services
  - Services, not yet completely part of SeerSuite, but may share the same framework, infrastructure.

Appendix - Digital Libraries


Repository
1.5 TB
30 GB
Maintenance
120 GB
2.1 TB
2.4 GB
Ingestion
1 GB
0.5 GB
Database
120 GB
30 GB
2 TB
Index
31 GB
54 GB
Web Service
30 GB
Extraction
150 GB
F. Crawler
3 TB
150 GB
Web

Outline – HotCloud 2010

- Introduction
- Motivation/Our Contributions
- SeerSuite
- Component Hosting
- Content Hosting
- Peak Load Hosting
- Future Work
- Conclusions