An Automated Approach to Cloud Storage Service Selection

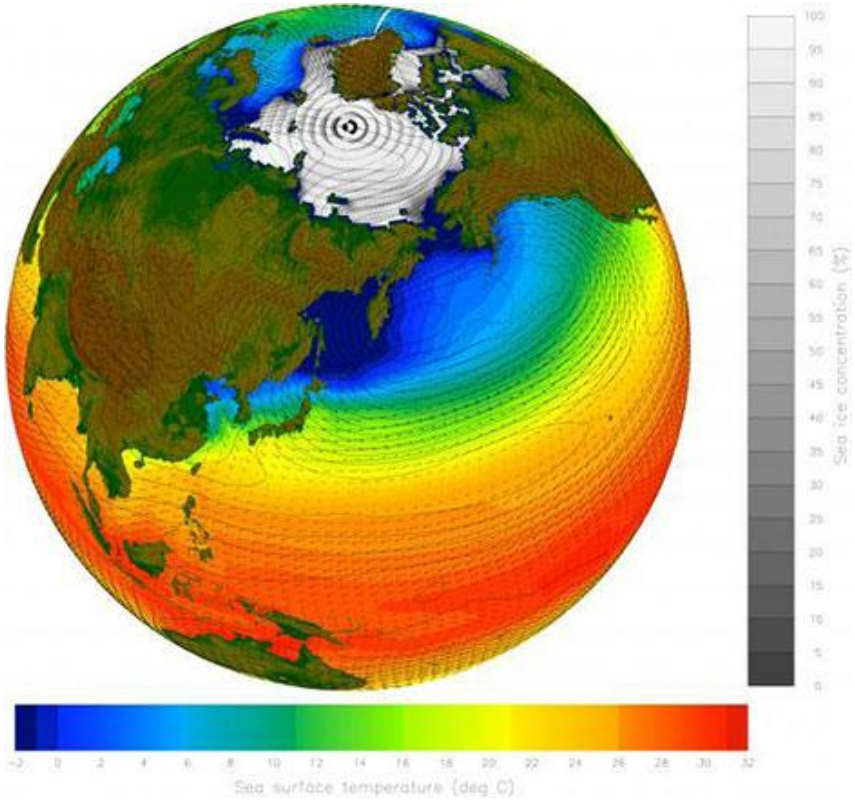
Science Cloud 2011

Arkaitz Ruiz Álvarez,
Marty Humphrey

6/8/2011

School of Engineering & Applied Science
University of Virginia

Advances in scientific computing require more storage and computation capabilities



Cloud computing provides on demand, cheap and scalable computation and storage

Problem Statement: How do cloud users choose storage services?

Scientists

- High level data requirements
- How much does it cost?
- How fast is it?

Cloud Services

- Different APIs
- Different capabilities, cost, performance
- Choice of geographically dispersed providers and datacenters

High level view of our approach

- Describe storage systems in a machine readable format
- Encode user requirements
- Attempt to match each dataset to each storage system, present results to the user

Our target storage systems are the most commonly used storage abstractions

- Amazon: S3, EBS, SimpleDB, Relational DB
- Azure: Blobs, Azure Drives, Tables, SQL Azure
- Local clusters: NFS, Hadoop, MySQL

We developed a XML schema to describe storage services

```
<xsd : element name="CloudProvider “ type="tns : CloudProviderType"/>
<xsd : complexType name="CloudProviderType">
   <xsd : element name="S t o r a g e Se r v i c e s ">
      <xsd : element name="St o r a g eSe r v i c e ">
         <xsd : element name="Regions">
            <xsd : element name="Cost">
            <xsd : element name="Performance">
       <xsd : element name="StorageAbs t r ac t ion">
          <xsd : element name="Container">
           <xsd : element name="Object">
<xsd : / complexType>
```

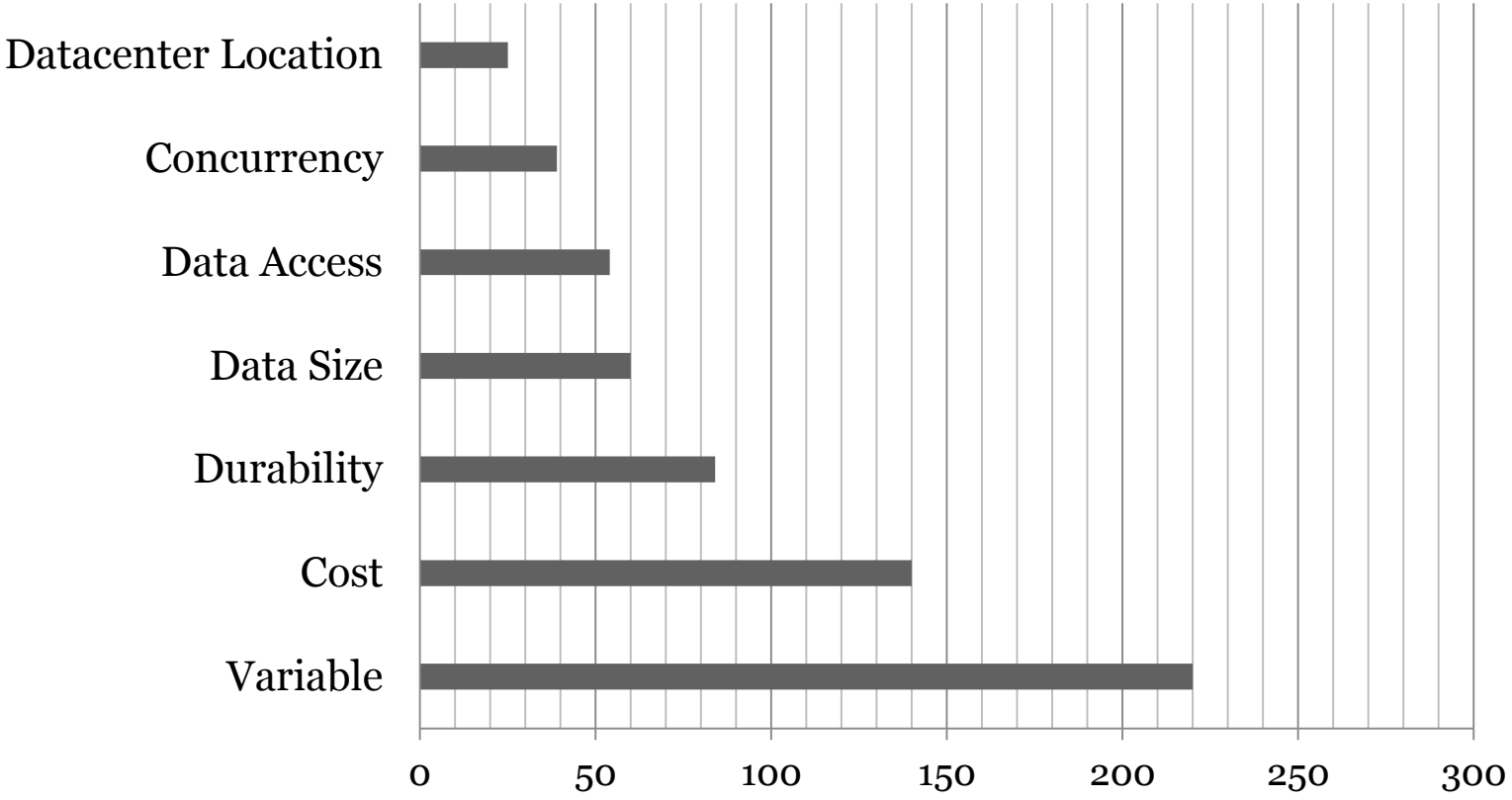
Example section of the Azure cloud description

```
<Object ID="AZURE_BLOB_PAGE" Name="Windows Azure Page Blob" Description="The Blob ..."
    NamingRegularExpression="^(?![0-9]+$)(?!-)[a-zA-Z0-9-]{,63}(?&lt!-)$"
    ModificationDate="true" CreationDate="false" MaxSizeKB="1073741824">
 <Interface>
    <CustomInterface RandomAccess="true">
      <Delete>Delete Blob</Delete>
      <Download>Get Blob</Download>
      <Upload>Put Blob</Upload>
      <CreateSnapshot>Snapshot Blob</CreateSnapshot>
      <ListParts>Get Page Regions</ListParts>
      <UploadPart>Put Page</UploadPart>
      <Lease Duration="60" API="Lease Blob"/>
      <Copy>Copy Blob</Copy>
    </CustomInterface>
  </Interface>
  <Metadata>
    <MetadataInterface>
      <CustomInterface>
        <Download>GetBlobMetadata;GetBlobProperties</Download>
        <Upload>SetBlobMetadata;SetBlobProperties</Upload>
      </CustomInterface>
    </MetadataInterface>
    <MetadataSet type="SystemMetadata" abstraction="ValuePair"/>
      <MetadataSet type="UserMetadata" abstraction="ValuePair"/>
    </Metadata>
    <Data DaysToExpiration="0" Formats="binary;text" ReadOnly="false">
      <RandomAccess/>
    </Data>
</Object>
```

http://www.cs.virginia.edu/~ar5je/SCPaper.html

School of Engineering & Applied Science
University of Virginia

Our prototype encodes user requirements as extended classes



Number of C# lines of code for each class extending *Requirement*

Use Cases

- Design of an application
- Cost savings analysis
- Cost and performance estimation
- Amazon EC2 to Eucalyptus

In our first use case we recommend storage services based on user's requirements

- Each dataset is matched against each storage service
- Possible matches meet user's requirements (if none, partial matches are shown)
- Results include an estimation of the performance and cost of the service

Dataset	Amazon	Azure	Local Cluster
Satellite Data	S3	Page Blob	Hadoop*, NFS*
Intermediate Results	S3 RRS*, SimpleDB*	Page Blob*, Table*	NFS*
Experimental Results	S3	Page Blob, Block Blob	NFS

In our second use case we estimate cost savings by switching storage services

Current Amazon Service		Service Recommendation			Savings	Pros	Cons
Service	Region	Cloud	Service	Region			
S3	US CA	Azure	Page Blob	US	$11	2.09x better latency	
S3	US CA	Azure	Block Blob	US	$11	2.07x better latency	
S3	US CA	Amazon	S3	US	$36		
S3	US CA	Amazon	S3 RRS	US	$153.5		0.0099999% less durability
S3	US CA	Amazon	S3 RRS	US CA	$127.5		0.0099999% less durability
S3	US CA	Local	NFS	US	$407.5	117.6x better latency	0.499999% less durability
SimpleDB	US CA	Amazon	SimpleDB	US	$.2		
RDS	US CA	Amazon	RDS	US	$92		
RDS	US CA	Azure	SQL	US	$130	1.31x better latency	

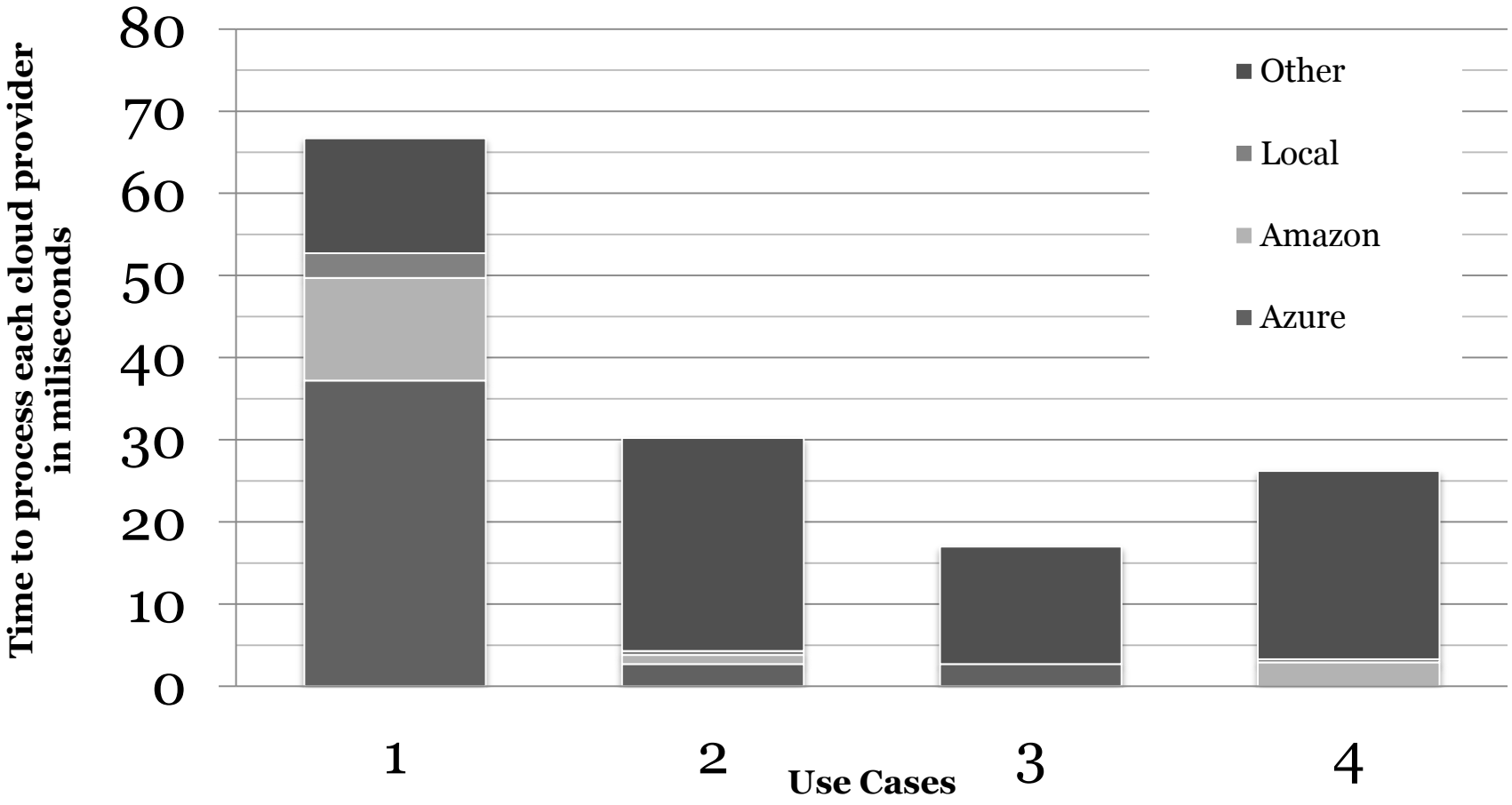
In our third use case we estimate cost and performance for current storage services

- User inputs several rate growth scenarios (size of data, number of clients)
- Our application outputs estimates of cost and performance for each scenario

In our fourth use case we compare storage options to assist on cloud migration

Current Storage S.	New Storage S.	Latency	Throughput	Comments	Cost
S3	NFS	>1ms	39.02 MB/sec	S3 offers 1% more durability (99.99999999%) NFS container capacity is 10 GB (2500 GB req.)	$250 one-time $0 monthly
S3	Hadoop DFS	N/A	N/A	S3 offers 0.00099999% more durability (99.99999999%) Hadoop container capacity is 1024 GB (2500 GB req.) Hadoop does not support random access	$250 one-time $0 monthly
S3	GPFS	N/A	N/A	S3 offers 0.00099999% more durability (99.99999999%)	$250 one-time $0 monthly
S3	S3 (no change)	205 ms	3.17 MB/sec	Data transfer fees incurred by each data access	$0 one-time $362.5 monthly
SimpleDB	MySQL	3.45 ms	288.8 items/sec	SimpleDB offers 1% more durability (99.99999999%) Interface differences: SQLInterface and AttributeValue	$0.8 one-time $0 monthly
SimpleDB	SimpleDB (no change)	35.46 ms	28 items/sec	Data transfer fees incurred by each data access	$0 one-time $5.88 monthly
RDS	MySQL	>1 ms	14359 items/sec	RDS offers .5% more durability (99.5%)	$0.7 one-time $0 monthly
RDS	RDS (no change)	13 ms	14172 items/sec	Data transfer fees incurred by each data access	$0 one-time $328.2 monthly

Performance Evaluation



Future Work

- Include the cost of computation
- Automatically select best matching storage service based on latency and/or cost
- Explore automatic computation (job) placement given current storage locations

Summary

- Our approach is based on a machine readable description of storage services and extensible code to represent user's requirements
- Our output is a match of application's datasets to storage services that meets storage requirements and provides cost and performance estimations
- We explored different use cases for cloud users