Performance Analysis of Windows Azure Data Storage Options

Istvan Hartung(✉) and Balazs Goldschmidt

Department of Control Engineering and Information Technology,
Budapest University of Technology and Economics,
Magyar tudósok körútja 2, Budapest 1094, Hungary
{hartung,balage}@iit.bme.hu

Abstract. Windows Azure provides an IaaS cloud service with virtual machines, web and worker roles and practically unlimited, pay-as-you-go storage options which can be used for applications requiring big data or parallel computing which is important in many fields including biology, astronomy, nuclear physics and economics.

When moving an application or computation task to the cloud it is very important to perform proof of concept performance testing and to carefully choose the proper building blocks for the given tasks. Windows Azure provides multiple data management options with a relational SQL database for transactional data access, Azure Tables for auto scalable storage of unstructured data, and a blob storage for storing large amounts of binary data which is easily mountable to a given virtual machine.

In this paper we present a general performance analysis of the Windows Azure cloud with focus on cloud storage options. We present an environment to perform automated testing of the major features of Azure storage and we also present the preliminary results and suggestions regarding the usage of the different services.

1 Introduction

Recently the cloud computing paradigm [2] has led to novel solutions for storing and processing data both in the industry and in the academic world. Any resource-intensive task may be moved to the cloud which provides scalable and practically unlimited resources where the provisioning of 1000 virtual machines for one hour costs as much as provisioning one instance for 1000 hours. Many companies including Amazon, Google, HP, IBM and Microsoft offer public cloud infrastructures available to anyone providing virtual machine instances, novel storage services and traditional database solutions among other enterprise solutions.

The use of public cloud for scientific calculations in many fields is a logical consequence of the requirement of both large scale data storage options and parallel computing. Clouds provide a cheap alternative to specialized clusters and supercomputers. Amazon Web Services provides public data sets in a form of

I. Lirkov et al. (Eds.): LSSC 2013, LNCS 8353, pp. 499–506, 2014.
DOI: 10.1007/978-3-662-43880-0_57,

a centralized repository which can be accessed in a few minutes and are completely free [1]. These datasets contain terabytes of data from a wide spectrum of domains including biology, astronomy, economics, chemistry, mathematics and geography and are freely available with pre-configured Amazon EC2 instances.

The Windows Azure cloud provides Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) components that support parallel computing and storage of large data. The platform provides novel storage options for large binary data in blob storage, Tables storage with NoSQL capabilities for unstructured data, and a queue service for asynchronous communication between elements of a distributed system. Besides the large and scalable solutions Azure also provides a full featured traditional database-as-a-service, which is built on technologies of SQL Server and provides a traditional interface. The Azure platform is already used in various scientific projects, [8] but the proper storage solution must be carefully chosen for any application.

In this paper we introduce the storage options for the Azure cloud and provide a solution to benchmark the PaaS components of the Azure platform and present our first results and recommendations for the use of storage options.

The rest of the paper is organized as follows. In the second section, related work is examined. In the third section, the Azure platform is introduced, the measurement architecture is shown and finally the evaluation of the results is provided. The fourth section summarizes the results.

2 Related Work

In the past few years cloud computing has drawn attention from many researchers. Back in 2008 Vaquero et al. analyzed 22 different definitions for cloud computing and compared the paradigm with Grid technology [11].

Performance of cloud computing has been studied by many research groups. Many investigated the low level performance of virtualization in general and the performance of Xen, which is used for virtualization by many public and private cloud providers [3,7].

Others compared the performance of major public IaaS cloud providers in terms of network capabilities, memory, disk and CPU utilization, binary object storage and queue access [6]. Li et al. focused on comparing the common services provided by the public providers with an application that could be deployed on a virtual machine of the cloud. We present a distributed testing application that is freely scalable and provides testing of big parallel workloads on Azure.

Jackson et al. provide a comparison of the performance of Amazon EC2 and the standard supercomputing centers and showed that there is a correlation between the amount of time a given application spends communicating through network and its performance on Amazon EC2 [5]. The use of Amazon EC2 for scientific calculations was analyzed in aspects of virtual machine performance, resource acquisition and virtualization by Ostermann et al. [9].

There are papers about the performance of Windows Azure, [4] but storage services over went a major performance change in December of 2012 so none of

them provide up to date information about the capabilities of the PaaS storage service. Additionally our goal was to compare the performance of SQL Database with Azure Tables storage.

Windows Azure has been used in scientific computation in many fields including biology [10], where it has been shown that moving computation to data, supporting large data sets by the cloud simplify the implementation of some problems over traditional systems.

3 Performance Measurements in Windows Azure

This section introduces the Azure platform, the measurement architecture and the results of the measurements.

3.1 Windows Azure

The Windows Azure platform is the public cloud provider of Microsoft. It offers both Linux and Windows based virtual machines as well as scalable compute instances, known as roles. The role instances can be categorized into Web Roles, which have a public interface, run a preconfigured IIS and are accessible via HTTP or HTTPS and Worker Roles that typically run background computation tasks. Azure also provides components that aid communication between roles and solutions to store terabytes of data as well as an SQL database as a service component for relational data.

The Azure Blob storage provides storage for binary data and metadata in containers. The container provides a logical grouping and defines the level of sharing. There first of the two types of supported blobs, block blobs are targeted for streaming workloads, have a maximum size limit of 200 GB and are updated with commit-based semantics. Each blob consists of a list of blocks, which can be modified by first uploading the new uncommitted blocks for the blob file, then a single call with the list of the new blocks will commit all changes on the blob. Page blobs may be files up to 1 TB of size and support random write workloads. Each page blob consists of an array of pages, which can be immediately updated and support change in only a portion of the file.

The Azure Tables service allows storage of enormous amounts of data with efficient querying and insertion. Each created table contains a set of entities which can hold up to 255 properties where the size of the entities must be under 1 MB. A single table may contain different types of entities the only restriction is that each entity within a table must have a unique PartitionKey and RowKey. These are the only two columns in a table that are indexed, further on the partition key is used by the Tables service to distribute data across the storage instances automatically. Entities with the same PartitionKey within a table will always be stored on the same instance.

The Windows Azure SQL Database is a relational database with size up to 150 GB with almost full Transact-SQL support, including creating and executing stored procedures, functions, transactions and triggers, supporting security

via multiple logins and users, and a configurable firewall. The database provides federation via data partitioning to support larger data set sizes, but lacks transaction support over multiple databases.

3.2 Measurement Architecture

When choosing the proper environment for testing the Azure service it was very important to keep certain requirements in mind. For testing the capabilities of Azure storage it is very important for the network connection to be sufficient, because it could easily become a bottleneck when testing. It is also important to have multiple instances to run tests to provide multiple endpoints. Otherwise load balancers, firewalls or other network components may throttle connections or filter traffic in order to defend against denial of service based attacks. Another important aspect is for the clients to store all measurement data persistently and in a way that does not limit the measurement itself. The measurement data should be collected independently and should be converted to in an easy to process form.

Our constructed measurement system is built from three layers of components (see Fig. 1). The input layer consists of a scheduler role, which is an Azure Web Role which provides a web service interface and accepts a set of queries and tasks that are to be run on the system under test. The virtual machine places these received tasks in an Azure Queue and are later processed and executed by one or all of the tester roles with given amount of execution counts.

The number of tester roles can be scaled up from the Azure Management Portal dynamically during the execution of the test. The results of the given test (elapsed time, query identifier, number of results) are logged using Windows Azure Diagnostics. The data this way will be automatically stored in memory or on the local computer and will be periodically transferred to a given storage account by a different process. Data here is stored in a standard form in Azure Tables.

The third layer (containing the report role) reads this table and parses the log entries. It also puts the entries in chronological order and converts the data to a standard csv form. The test results are stored in merged and consolidated

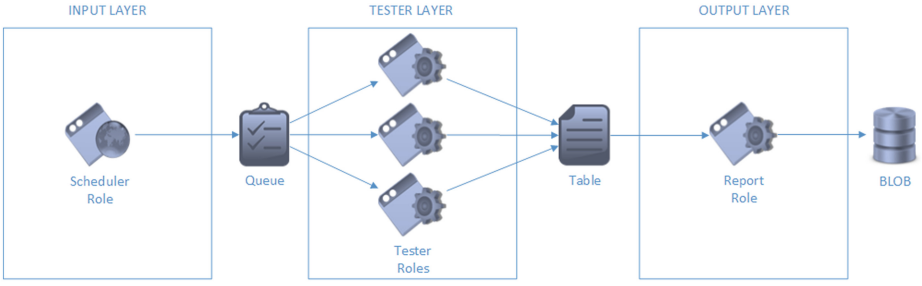

Fig. 1. Measurement architecture

format on a blob storage and are downloadable for later processing. It is not required to run the input and the output layer on the same cloud environment as the tester roles.

3.3 Azure Storage Evaluation

The first goal of our research was to test the capabilities of Azure Tables storage compared to Azure SQL Database. Azure Tables promises to provide a scalable NoSQL service which is simultaneously accessible by many clients. The known advantage of SQL Database is the full potential of a relational database and the ability to add additional indexes and constraints on the dataset. The other advantage is the power of SQL query language, with aggregate functions, wildcards and joins. The disadvantages are that in SQL only less than 100 GB data, has limitations in parallel executable queries and costs more in Azure. For the Tables storage one must choose the PartitionKey and RowKey carefully, because it determines the indexes, the placement of data and implicitly the performance of queries.

In our first experiment we converted a free flight database containing 10 million records of world-wide flight data to a proper Azure Tables and an SQL table format. We filled the database continuously with batch insertions and queried the database with four different types of queries simultaneously. The first type of query selected a single row from the database which was identified by the primary key in SQL and by the PartitionKey in Azure Tables. The second type of query selected rows via a portion of the primary key (PartitionKey) and via a portion of an other indexed row (the RowKey in Azure Tables). The third query selects multiple rows based only on an indexed row (in the case of Azure Tables only the RowKey). The fourth query selects rows based on a portion of the primary key and a non-indexed column.

The test was executed multiple times with different storage accounts across regions and we got similar results with each run. One instance was continuously inserting entities with multiple threads into the given data provider while other 10 instances where executing queries parallel. The experiments showed that in case of Azure tables the insert operation is slightly faster than in case of the SQL database (see Table 1). The first, second and the last query ran with no significant difference in case of a few rows and millions of rows. Both Azure SQL and tables performed well with few million rows.

Based on only these numbers only there is no real gain in using Azure Tables over SQL Database. But there is one more aspect that must be examined.

Table 1. SQL and Tables performance (Milliseconds/Returned row)

	Insert	1. Query	2. Query	3. Query	4. Query
Azure Tables	4	102	620	274399	56
SQL database	17	995	1015	14767	7.16

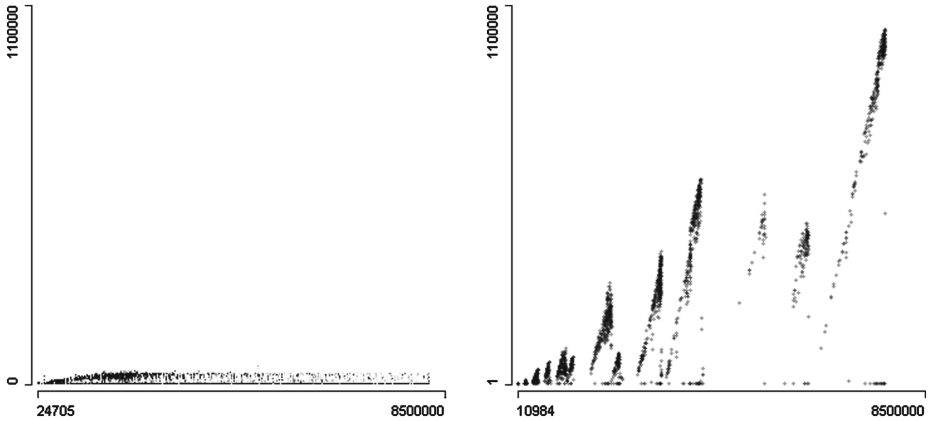

Fig. 2. *Left*: The 3. Query in SQL, horizontal: number of entities in database, vertical: retrieval time for a single row (in milliseconds), *Right*: The 3. Query in Azure Tables, horizontal: number of entities in database, vertical: retrieval time for a single row (in milliseconds)

The percent of failed requests in case of the SQL database was close to 10 % while the number of failed requests in case of the Tables test was under 0.1 %. One SQL Database can only execute 180 worker threads simultaneously, so all other connection attempts will be rejected. Based on the documentation Azure Tables detects denial of service attacks, so it may filter incoming requests, but in case of a slowly growing load it is able to server 20,000 entities per second, though we where not able to get near this limit (our maximum value was 9,000 entities per second) due to the lack of instances in our tests.

The other significant difference was the round trip times of the third query. The execution of this query required a full table scan and the size of the returned data is also significant. Azure Tables stores data on multiple servers and can only return data in batches containing a maximum of 1000 entries. A single query is executed relatively fast (within a few seconds), but with more requests the runtime of the query linearly grows with the number of parallel requests. If we look at the results when calling the SQL Database there is a significant difference. After a certain amount of data is in the database the retrieval time per row is relatively constant (see Fig. 2). The exact cause of the phenomenon requires more research, but it is highly probable that it's due to the fact that only 180 parallel queries are served by the database simultaneously, while during the high response times Tables was serving more than 2,000 concurrent requests.

4 Conclusion

In this paper we presented a framework that may be generally used to execute any kind of performance tests in the Azure cloud. The application may be deployed in the same data center as the system under test thus providing low

network latency. The architecture provides option to scale to virtually any number of instances and can execute various types of tasks queued by a scheduler. The results are stored persistently in Azure Tables and are later processed by a different virtual machine.

By executing tests we ascertained that storing data in the novel storage options of Azure can be very efficient, the throughput of 20,000 entities per seconds can be reached with enough parallel endpoints with an ingress bandwidth of 10 GB/s and egress bandwidth of 15 GB/s. The greatest challenge is to choose a proper PartitionKey and RowKey for stored data. The format of the stored data specifies the queries that will run efficiently on the table service due to the fact that additional indexes can not be added to the entities. It is also very important to consider the PartitionKey in particular, because entities with different values may be stored on different instances, so queries that affect multiple PartitionKeys may have to call multiple server instances. The other important factor when choosing Azure Tables over SQL may be the fact that it supports extremely high values of simultaneous requests. On the other hand the Azure SQL Database provides multiple indexes, stored procedures and database constraints. If our dataset needs these kinds of capabilities then we should consider using a federated database to achieve higher degree of parallelism.

Acknowledgments. The work reported in the paper has been developed in the framework of the project "Talent care and cultivation in the scientific workshops of BME" project. This project is supported by the grant TÁMOP-4.2.2.B-10/1–2010-0009.

References

1. Amazon Web Services: Public data sets catalog, March 2013. http://aws.amazon.com/datasets
2. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M.: Above the clouds: a berkeley view of cloud computing. Technical report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, February 2009. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html
3. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, SOSP '03, pp. 164–177. ACM, New York (2003). http://doi.acm.org/10.1145/945445.945462
4. Hill, Z., Li, J., Mao, M., Ruiz-Alvarez, A., Humphrey, M.: Early observations on the performance of windows azure. In: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, HPDC '10, pp. 367–376. ACM, New York (2010). http://doi.acm.org/10.1145/1851476.1851532
5. Jackson, K.R., Ramakrishnan, L., Muriki, K., Canon, S., Cholia, S., Shalf, J., Wasserman, H.J., Wright, N.J.: Performance analysis of high performance computingapplications on the amazon web services cloud. In: Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science, CLOUDCOM '10, pp. 159–168. IEEE Computer Society, Washington, DC (2010). http://dx.doi.org/10.1109/CloudCom.2010.69

6. Li, A., Yang, X., Kandula, S., Zhang, M.: Cloudcmp: comparing public cloud providers. In: Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, IMC '10, pp. 1–14. ACM, New York (2010). http://doi.acm.org/10.1145/1879141.1879143
7. Menon, A., Santos, J.R., Turner, Y., Janakiraman, G.J., Zwaenepoel, W.: Diagnosing performance overheads in the xen virtual machine environment. In: Proceedings of the 1st ACM/USENIX International Conference on Virtual Execution Environments, VEE '05, pp. 13–23. ACM, New York (2005). http://doi.acm.org/10.1145/1064979.1064984
8. Microsoft research: cloud research projects, March 2013. http://research.microsoft.com/en-us/projects/azure/projects.aspx
9. Ostermann, S., Iosup, A., Yigitbasi, N., Prodan, R., Fahringer, T., Epema, D.: A performance analysis of EC2 cloud computing services for scientific computing. In: Avresky, D.R., Diaz, M., Bode, A., Ciciani, B., Dekel, E. (eds.) Cloudcomp 2009. LNICST, vol. 34, pp. 115–131. Springer, Heidelberg (2010)
10. Qiu, X., Ekanayake, J., Beason, S., Gunarathne, T., Fox, G., Barga, R., Gannon, D.: Cloud technologies for bioinformatics applications. In: Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers. MTAGS '09, pp. 6:1–6:10. ACM, New York (2009). http://doi.acm.org/10.1145/1646468.1646474
11. Vaquero, L.M., Rodero-Merino, L., Caceres, J., Lindner, M.: A break in the clouds: towards a cloud definition. SIGCOMM Comput. Commun. Rev. **39**(1), 50–55 (2008). http://doi.acm.org/10.1145/1496091.1496100