Journal of Software Engineering and Applications, 2014, 7, 686-693
Published Online July 2014 in SciRes. http://www.scirp.org/journal/jsea
http://dx.doi.org/10.4236/jsea.2014.78063

Scientific Research

Cloud Computing and Big Data: A Review of Current Service Models and Hardware Perspectives

Richard Branch, Heather Tjeerdsma, Cody Wilson, Richard Hurley, Sabine McConnell

Department of Computing and Information Systems, Trent University, Peterborough, Canada
Email: richardbranch@trentu.ca, heathertjeerdsma@trentu.ca, codywilson@trentu.ca, rhurley@trentu.ca, sabinemcconnell@trentu.ca

Received 9 May 2014; revised 5 June 2014; accepted 2 July 2014

Abstract

Big Data applications are pervading more and more aspects of our life, encompassing commercial and scientific uses at increasing rates as we move towards exascale analytics. Examples of Big Data applications include storing and accessing user data in commercial clouds, mining of social data, and analysis of large-scale simulations and experiments such as the Large Hadron Collider. An increasing number of such data—intensive applications and services are relying on clouds in order to process and manage the enormous amounts of data required for continuous operation. It can be difficult to decide which of the many options for cloud processing is suitable for a given application; the aim of this paper is therefore to provide an interested user with an overview of the most important concepts of cloud computing as it relates to processing of Big Data.

Keywords

Big Data, Cloud Computing, Cloud Storage, Software as a Service, NoSQL, Architectures

1. Introduction

Attempting to define cloud computing can be as nebulous an activity as the term itself implies. However, according to the National Institute of Standards and Technology (NIST), the definition of cloud computing is "*a model for enabling ubiquitous*, *convenient*, *on-demand network access to a shared pool of configurable computing resources* (e.g., *networks*, *servers*, *storage*, *applications*, *and services*) *that can be rapidly provisioned and released with minimal management effort or service provider interaction*" [1]. Other definitions have been put forward, but the above is one of the most accepted and best enumerated. One of the many reasons for the

How to cite this paper: Branch, R., Tjeerdsma, H., Wilson, C., Hurley, R. and McConnell, S. (2014) Cloud Computing and Big Data: A Review of Current Service Models and Hardware Perspectives. *Journal of Software Engineering and Applications*, **7**, 686-693. http://dx.doi.org/10.4236/jsea.2014.78063

ambiguity of the term and its use is the complicated interplay of service models, architectures, storage, and software deployed in various cloud applications today. In this paper, we review the most common approaches and aspects in an attempt to provide researchers a tool to guide them in the selection process when considering cloud applications for processing of Big Data.

2. The User Perspective

2.1. Software and Service Models

The NIST definition [1] mentioned above also describes the following as essential characteristics for cloud computing:

On-Demand Self-Service: Cloud users can obtain computing capabilities as required with the cloud provider without human interaction.

Broad Network Access: Cloud capabilities are accessible over a network while provided in such a way as to allow access via numerous heterogeneous client platforms.

Resource Pooling: The cloud provider combines multiple resources to service numerous cloud users dynamically, with different physical and virtual resources as demanded by the user. This is done in a manner which presents the resources as location independent while the physical component parts are invisible to the cloud user. The serving of multiple users is done using a multiple tenant model.

Rapid Elasticity: Presents a view of unlimited resources to the cloud user. Resources are dynamically allocated and deallocated as required in order to provide immediate and seemingly limitless scalability.

Measured Service: Cloud services and resources are automatically controlled and optimized via metering by the cloud system. Resource monitoring is leveraged in order to provide transparent resource utilization reporting to the cloud user and provider.

These services are provided via a variety of service models, three of which have been adopted as primary service models: *Software as a Service* [*SaaS*], *Platform as a Service* [*PaaS*], and *Infrastructure as a Service* [*IaaS*] [1]-[4].

SaaS provides access to applications which execute on the cloud providers infrastructure and are managed by the cloud provider [1]. Applications are accessible from numerous heterogeneous client interfaces, for example thin clients such as a web browser [1]. Cloud customers do not have control over the underlying cloud infrastructure or applications which they are using, except potentially for specific user configurations unique for each user [2].

PaaS gives cloud users the ability to deploy user-created or obtained applications onto the cloud provider's infrastructure [2]. Deployment is provided through the use of programming languages, libraries, services, and tools which are made available by the provider [1]. As with SaaS, cloud users are not given access to the underlying cloud infrastructure. Instead, cloud users are provided with the ability to manage and configure the deployed applications, potentially having limited access to certain configuration settings within the operating system environment [1]. In general, an *Integrated Development Environment* (IDE) or multiple IDE's are provided in order to facilitate development for the cloud user. Examples of PaaS providers include Google App Engine [5] and Microsoft Azure [6].

IaaS gives cloud users the ability to manage the underlying infrastructure of the cloud provider from an abstract or virtual point of view [1]. This provides the user with on-demand provisioning of resources (e.g. CPU, Memory, and Storage) [3]. IaaS allows the cloud user to manage and deploy the software they choose, including the underlying operating system [2]. The physical infrastructure such as servers, hardware, and networks are still managed by the cloud provider and are not available to the cloud user [1]. An example of an IaaS provider would be Amazon EC2 [7].

2.2. Deployment Models

There are currently four commonly used deployment methods for clouds: *Private Cloud*, *Public Cloud*, *Community Cloud*, and *Hybrid Cloud*. Other labels for specific types of deployment models have been suggested; however they still tend to fall under these more general headings.

A *Private Cloud* is used by a single organization, potentially with multiple user groups within the organization [1] [3]. It may be maintained, owned, and operated by the organization, a third party, or some combination

of the two [1]. Physically it may exist at the site of the organization or elsewhere [1].

A *Public Cloud* is used by the general public [1] and is owned, operated, and provided by a business, academic institute, government organization, or some combination of the three [2]. In general, the Public Cloud will be housed at the physical location of the provider [2]. The resources of the Public Cloud are offered to cloud users as a service [3]. Currently Public Clouds are the most dominant deployment model in use [4].

A *Community Cloud* is used by some specific group or community of users from a combination of different organizations which share some common goal or concern [1]. Goals tend to be related to security, compliance, or some specific mission [1]. This cloud may be managed, operated, and constructed by a group, single organization, third party, or some combination of the three [2]. The cloud may be physically located at the site of a single organization, spread across a group of organizations, located at a third party, or some combination of the three [2].

A *Hybrid Cloud* is a combination of two or more of the above cloud deployment models [1]. Each cloud used in the combination are unique and independent clouds which are integrated in such a way as to allow portability between the unique clouds [1] [3]. Combinations of the different clouds allow cloud users to create a new cloud which can add additional benefits for the cloud user such as creating new services.

2.3. Programming Models

In contrast to more traditional, low-level High Performance models that interact with parallel and distributed hardware, such as OpenMP for shared memory and MPI for distributed memory systems, users typically interact with the cloud at a higher level using tools such as Hadoop. The more obvious tradeoff for this scenario is the loss in efficiency for gains in usability. A more implicit tradeoff is the lack of communication possibilities for interaction between different processing units as the program is deployed. This implies that cloud computing is inherently biased towards applications that are embarrassingly parallel scenarios, which require a minimal interaction between processing components. While traditionally these are easiest to implement from a parallel viewpoint and are a good fit for most business applications, they are not usually a good match for scientific applications, which often require a more rigorous communication scheme.

2.4. Practical Issues

The dominant practical issues regarding using Cloud Computing are interoperability, moving data to the cloud, and portability.

Interoperability between clouds, in the sense of communication between them [2], is somewhat problematic at this time. Multiple independent APIs exist for interacting with current cloud offerings. These unique APIs can be a barrier to cloud users who wish to combine the functionality of multiple clouds in order to provide new services [4]. Standardization of the cloud interfaces would certainly go a long way to mitigating this problem [4].

Moving data to a cloud can be problematic in the case of Public Clouds. Questions regarding how to quickly transfer data to the cloud in a secure fashion are still very relevant. This is especially a concern when using a cloud for Big Data analysis, and serves as a bottle neck for timely analysis of data when volume is an important factor for the data being analyzed. To illustrate this problem, consider a data set which is 10 Terabytes in size. If we were to move this data to a cloud provider using a connection which supports up to 5 Mb/s this would take approximately 185 days.

Portability can vary depending on the service model that is in use. When examining portability from the view of IaaS, we see the requirement for a lower level of portability, such as being able to conveniently move a virtual machine from one cloud provider to another [1]. Portability can be considered when attempting to deploy a service on more than one type of cloud [1], or when copying data from one cloud provider to another [1]. This may be complicated, depending on the mechanism used within the cloud provider to store the data.

One of the advantages of the cloud is its scalability. The illusion of unlimited resources through the use of resource elasticity and on-demand self service provides a highly scalable environment [2]. This scalable environment can be leveraged for data storage and analysis. In addition, no upfront costs are required for public cloud users as the infrastructure is completely managed by the cloud provider [3]. Concerns of over-provisioning from the perspective of the cloud user are eliminated when using cloud services due to the on demand nature of public clouds [8], while data and analysis results can be easily shared among groups of users or organizations.

However, resource pooling means that multiple tenants are sharing physical resources which opens up oppor-

tunities for side channel attacks [4]. There are also costs associated with transferring data and communication with the cloud is increased [4]. Furthermore, the wide range of APIs which are in use by multiple clouds translates to a significant amount of time devoted to learning how to interface with these clouds [4].

3. The Data Perspective

3.1. Cloud Storage

Cloud storage may often be referred to as its own type of storage medium, however, cloud storage is not a different medium, but rather refers to where the data is stored (offsite) and how the data is accessed (online).

The challenges for storing big data include cost, access latency, reliability, and capacity. One approach used for balancing these requirements is the implementation of tiered storage which uses a range of storage technologies (flash/solid state, hard disk, tape) with various characteristics. To the end user, the access to the data on the different mediums is hidden. Software is used to virtualize the different storage mediums to the user and other applications so it is viewed as if it is a single storage device. By using a tiered system along with hierarchical storage management software, the data can be stored on the most appropriate medium for its purposes in a manner which is transparent to the user. The philosophy is to store infrequently used data on cheaper, high capacity mediums and data that needs to be accessed quickly and frequently on expensive, faster media. Older data is often accessed less frequently and can therefore be moved by an automated process to a slower, cheaper tier [9].

Flash arrays are commonly used for the top tier. Flash storage does not have any moving parts, has less power requirements then other storage systems, and has quicker access times when reading data [10]. These devices are more expensive and less durable than traditional hard drives. They are ideal storage mediums for high performance, low-latency, real time analytics. IBM has added flash storage to some of their enterprise servers, for example their system using flash arrays can analyze and process 40,000 credit-card transactions per second [11]. One caveat to using flash arrays, is that they have a limit of write-erase cycles and therefore must have appropriate measures in place to handle flash array failures [10].

Rather than maintaining massive storage devices, many organizations are outsourcing their storage requirements to the cloud. Storing data in the cloud presents a new set of challenges, including the movement of large amounts of local data to and from the cloud. Some cloud providers are offering services to import and export data by mailing storage devices to the service provider rather than relying on the internet and bandwidth limitations [12]. Another new challenge faced with cloud storage is associated with security and privacy. Using a third party to manage an organization's data requires a certain level of trust. It is critical that a consumer be aware of the risks involved before placing their data in the hands of a cloud provider [13].

3.2. Data Access via NoSQL

The term *NoSQL* was first used to describe a relational database that did not provide an SQL language to query the data. This original definition is unrelated to NoSQL as it is today. In 2009, an employee of Rackspace used the same term for a conference discussing open-source distributed databases [14]. Today, NoSQL is not a standard, but rather a general category covering a number of different technologies that store and retrieve data in a non-relational manner. This general category includes databases that do not adhere to the ACID (atomicity, consistency, isolation, durability) principles that are found in traditional relational databases. Instead, NoSQL databases rely on a distributed computing model with the principle of "eventual consistency" [14].

Traditional relational databases bring a new set of challenges when used in conjunction with big data. These challenges have been a driving factor in the movement to NoSQL databases. Nicholas Greene clearly outlines the advantages and disadvantages of NoSQL databases demonstrating that NoSQL is not a replacement for relational databases, but rather fits specific needs which must be met when working with large data sets [15].

Relational databases scale up, requiring increased server size and power in order to scale up with the data. On the other hand, NoSQL databases scale out by distributing data across multiple hosts. This means that downtime is not necessary when expanding resources with NoSQL. The lack of downtime and the fact that NoSQL is not restricted to a specific schema makes it more flexible. NoSQL is also administrator friendly, less overhead management is required as NoSQL data stores are designed with the automated ability to repair and distribute themselves. Other benefits of NoSQL data stores are their simple data models and their open source nature. Typically these data stores can be implemented in a cost effective manner using cheap, low-grade commodity

computers. This is much more cost efficient than the high end servers which relational databases typically require.

However, it is unrealistic to assume that NoSQL data stores are a replacement for traditional relational databases such as SQL. As a result of NoSQL's open source nature and the variety of implementations that exist, not many reliable standards are available, causing portability to suffer. Performance and scalability are often put ahead of consistency and, as a result, NoSQL is often an unacceptable solution when working with data where consistency is critical, such as records of financial transactions. NoSQL data stores may be easier to manage, but due to a lack of maturity with this new technology, there are a limited number of administrators and developers with the knowledge and skills required. Finally, NoSQL data stores can be difficult to use for analyzing and reporting due to the nature of their non structured implementations [15].

3.3. NoSQL Subcategories

There are four main subcategories of NoSQL databases: *Key-Value Data Stores*, *Graph Databases*, *Document Databases*, and *BigTable Databases*.

Key-Value Data Stores are schema-less systems which utilize a key for each item. The concept of Key-Value pairs is a pattern used in low level computing, similar to hash tables, and is a technique known for its speed [16]. The values do not need to follow any strict data type restrictions and there are no validity rules enforced on the data. The data is essentially stored in a long thin table where the hash key is used to locate the data associated with it [17]. NoSQL systems which use the Key-Value design include DynamoDB, Riak, Redis, Memcache DB, and Project Voldemort.

Graph Databases use matrices which focus on the relationship between two entities, such as in social networking applications. In these graphs, the nodes are entities and the edges are relationships between the entities. As the focus lies on the relationships Graph Databases provide a fast solution when working with data where relationships are the important factor [18]. Systems which utilize Graph Databases include Neo4J, Orient DB, Allegro, and Virtuoso.

Document Databases store documents using key lookups, tagging and metadata. The data is stored within documents using a form of markup language. Markup languages include YAML, JSON and XML [19]. By including metadata within the contents, it is possible to not only locate data by a key, but also filter based on an item's metadata [17]. Systems which utilize Document Databases include MarkLogic, MongoDB, and Couchbase.

BigTable Databases are tabular, with a three dimensional key structure: row and column keys with a timestamp. This category of NoSQL data stores was first seen in Google's BigTable database, hence the name of the category [20]. Other systems that follow Google's BigTable concept are HBase (Built on top of Hadoop's HDFS), Accumulo, and Cassandra. These databases are also known as Tabular Stores. The timestamp allows for an item to be traced throughout its history. Google uses BigTable for projects including Google Earth, Google Analytics, and Google's personalized search [21].

NoSQL databases are still a fairly new method of storing and organizing data. They fit the needs and requirements that relational databases are not able to meet. However, they will not likely evolve into a replacement for relational databases in the foreseeable future, but will continue to develop in order to meet the needs, requirements, and challenges of big data which cannot be met by traditional relational databases.

4. The Hardware Perspective

4.1. Processing Units

The development of computing hardware has moved through multiple distinct architectures since the 1960s, from faster sequential machines to vector processors, massively parallel systems to multicore systems with accelerators. Although these innovations provide tremendous advantages to applications, they still suffer from I/O issues; access to the cloud via telephone systems is currently one of the weakest links given the time it takes to move the data. Another issue is that each of these distinct architectures require a different programming model, resulting in duplication of efforts and a loss of portability.

One of the more popular choices in current cloud systems are Graphics Processing Units (GPUs). The popularity and growth of GPUs was spurred by the ever-growing digital entertainment industry, specifically the gam-

ing industry, which demanded better rendering capabilities and higher resolutions. The acceleration of GPUs also brought about an interest in their use in computational problems. This led to the introduction of the Ge Force 8800 graphics card designed by NVIDIA in 2006, which allowed not only for the card to be used for graphics processing and game play but also for computing applications as well [22].

The current cloud computing scene is heterogeneous with respect to hardware, which, when combined with lessons learned from issues in parallel computing over the last five decades, implies that scalability, ease of use, portability, and efficiency will suffer.

4.2. The Network

Working with and analyzing big data sets can require a substantial amount of processing power. As a result, distributed computing is used in order to achieve more processing power, overcoming the limits of a single machine. The gains of additional processing power from the multiple machines are offset by the networks, while the communication costs associated with them become the bottleneck. Compared to storing and retrieving data from RAM even10-gigabit Ethernet costs several orders of magnitude more [23]. This cost is the reason why the network and topology used for big data applications is a critical piece of the solution. Big data management and analysis requires network architectures which differ from the traditional client-server applications. The change in network traffic direction, and unpredictable bursts of data, have attributed to the evolution of network architectures as traditional client-server/north-south networks do not meet the requirements of big data applications.

Determining the best network topology requires an understanding of how the data will flow in the application. The challenge with big data is that modeling this traffic is very difficult, if not impossible. A north-south flow of data from clients to servers is much easier to model as the flow of communication between nodes is known. The clients communicate with the server, and the server to the clients, but the clients do not require communication between each other. Big data applications require communication in an east-west direction, and predicating which node will need to communicate with which node is much more difficult [24].

In addition to the difficulty in predicting data flow, the increased use of virtualized environments brings another set of challenges. In a virtualized environment, locality of the machine is no longer controlled by the application in exchange for dynamic use of network resources. This could result in two heavy bandwidth machines being located on the same switch which would affect network communication costs. Some placement of resources may be attempted by the virtualization system but typically isn't guaranteed. These differences in big data networks compared to the client-server style applications, requires a different type of network to overcome these challenges [25]. One example of a new network topology that has grown in popularity for use in applications that require more east-west communication is the *spine fabric*, which provides equidistant and uniform bandwidth for the hosts [25]. Another tool used to help manage data flow is Data Center Bridging (DCB) which is a technology that is often available on 10 G and 40 G networks. It enables the traffic to be segregated based on classes and priorities of the data being transmitted. This enables a more predictable flow of traffic on the network [26].

Another characteristic of working with big data is a non-steady flow of traffic which requires the network to have the ability to handle traffic bursts effectively. A network that is not set up to handle traffic bursts can result in lost packets during transmission, which leads to poor performance as dropped packets need to be re-sent. Using Equal-Cost Multipath (ECMP) routing, where multiple routes are available to get from one location to another with the same number of hops, provides a much better performance for bursts of communication than other routing methods [27]. Such approaches are particularly important for scientific applications which tend to produce more bursty traffic than business applications, on both long and short term time scales.

5. Summary

In this paper we reviewed the most common approaches and aspects of Cloud Computing in an attempt to provide researchers a tool to guide them in the selection process when considering cloud applications for processing of Big Data. We did this by viewing Cloud Computing through the lens of the user, data, and hardware perspective.

From the user perspective we have given a brief overview of the current service and deployment models, and what distinguishes them from each other. This provides a guideline for researchers in order to assist them in making decisions regarding what might be a best fit for their goals. Highlighting the practical issues with inter-

operability, moving data to the cloud, and portability show opportunities which exist for future development of cloud computing and point to areas of concern that researchers need to address when making decisions about cloud computing. Particularly, security is of paramount importance to researchers who wish to keep their sensitive data private.

From the data perspective we see that traditional relational database schemes are being replaced by more unstructured methods of storing and accessing data through the use of NoSQL and its many and varied implementations. Additionally, storage mechanisms implemented via flash arrays are being embraced by cloud providers, implemented in ways which are transparent to the cloud user. Organizations are faced with the choice of maintaining their own expensive storage devices or utilizing the cloud for their storage needs.

Finally, from the hardware perspective we see an increasing interest in the use of distributed systems and GPUs as processing units for cloud providers, while disadvantages become apparent when network configuration is brought into the mix, creating bottlenecks associated with communication.

References

[1] NIST (2011) The NIST Definition of Cloud Computing. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

[2] NIST (2011) NIST Cloud Computing Reference Architecture. http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909505

[3] Zhang, Q., Cheng, L. and Boutaba, R. (2010) Cloud Computing: State-Of-The-Art and Research Challenges. *Journal of Internet Services and Applications*, **1.1**, 7-18.

[4] Dillon, T., Wu, C. and Chang, E. (2010) Cloud Computing: Issues and Challenges. *Proceedings of the* 24*th IEEE International Conference on Advanced Information Networking and Applications* (*AINA*), Perth, 20-23 April 2010, 27-33. http://dx.doi.org/10.1109/AINA.2010.187

[5] Google (2013) Google App Engine. https://developers.google.com/appengine/

[6] Microsoft (2013) Microsoft Azure. http://www.azure.microsoft.com/en-us/

[7] Amazon (2013) Amazon EC2. http://aws.amazon.com/ec2/

[8] Ambrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. and Zaharia, M. (2010) A View of Cloud Computing. *Communications of the ACM*, **53.4**, 50-58. http://dx.doi.org/10.1145/1721654.1721672

[9] Feldman, M. (2013) The Big Data Challenge: Intelligent Tiered Storage at Scale. http://www.cray.com/Assets/PDF/Integrated_Tiered_Storage_Whitepaper.pdf

[10] Rouse, M. (2008) Flash Storage. http://whatis.techtarget.com/definition/flash-storage

[11] Lawson, S. (2014) IBM Updates All-Flash Storage Array to Complement X6 Servers. http://www.infoworld.com/t/solid-state-drives/ibm-updates-all-flash-storage-array-complement-x6-servers-234432

[12] Amazon (2010) AWS Import/Export. http://docs.aws.amazon.com/AWSImportExport/latest/DG/whatisIE.html

[13] Jansen, W. (2011) Guidelines on Security and Privacy in Public Cloud Computing. National Institute of Standards and Technology, U.S. Department of Commerce, Computer Security Division, Gaithersburg.

[14] Williams, P. (2012) The NoSQL Movement—What Is It? http://www.dataversity.net/the-nosql-movement-what-is-it/

[15] Greene, N. (2013) The Five Key Advantages (And Disadvantages) of NoSQL. http://greendatacenterconference.com/blog/the-five-key-advantages-and-disadvantages-of-nosql/

[16] Williams, P. (2012) The NoSQL Movement: Key-Value Databases. http://www.dataversity.net/the-nosql-movement-key-value-databases/

[17] Loshin, D. (2013) An Introduction to NoSQL Data Management for Big Data. http://data-informed.com/introduction-nosql-data-management-big-data/#sthash.NuRvqbd4.dpuf http://data-informed.com/introduction-nosql-data-management-big-data

[18] Williams, P. (2012) The NoSQL Movement—Graph Databases. http://www.dataversity.net/the-nosql-movement-graph-databases/

[19] Williams, P. (2012) The NoSQL Movement: Document Databases. http://www.dataversity.net/the-nosql-movement-document-databases/

[20] Williams, P. (2012, November 13). The NoSQL Movement—Big Table Databases. http://www.dataversity.net/the-nosql-movement-big-table-databases/

[21] Fay Chang, J.D. (2006) Bigtable: A Distributed Storage System for Structured Data. *OSDI'*06: *Seventh Symposium on Operating System Design and Implementation.*

[22] Nickolls, J. and Dally, W.J. (2010) The GPU Computing Era. *IEEE Micro*, **30**, 56-69. http://dx.doi.org/10.1109/MM.2010.41

[23] Jacobs, A. (2009) The Pathologies of Big Data.acmqueue.

[24] McGillicuddy, S. (2013) IBM Selects Juniper QFabric for Big Data Networking. http://searchnetworking.techtarget.com/news/2240207684/IBM-selects-Juniper-QFabric-for-big-data-networking

[25] Sammer, E. (2012) Hadoop Operations. O'Reilly Media, Inc., Sebastopol.

[26] Merchant, S. (2011) Is a Fabric Architecture in Your Future? http://www.datacenterknowledge.com/archives/2011/08/04/is-a-fabric-architecture-in-your-future/

[27] Cisco (2014) Big Data in the Enterprise—Network Design Considerations White Paper. http://www.cisco.com/c/en/us/products/collateral/switches/nexus-5000-series-switches/white_paper_c11-690561.html

Scientific Research Publishing (SCIRP) is one of the largest Open Access journal publishers. It is currently publishing more than 200 open access, online, peer-reviewed journals covering a wide range of academic disciplines. SCIRP serves the worldwide academic communities and contributes to the progress and application of science with its publication.

Other selected journals from SCIRP are listed as below. Submit your manuscript to us via either submit@scirp.org or Online Submission Portal.