Wide Area Network Planning Drivers and Methods for Internet Services

Eric Kumar
School of Architecture
Carnegie Mellon University
Pittsburgh, PA USA
ekumar@andrew.cmu.edu

***Abstract*—This paper considers global-scale wide area network (WAN) capacity planning methodologies for distributed storage systems in which compute and data store resources are geographically dispersed in remote data-centers (DCs). The disparate data-center resources are bridged together by the WAN spanning across continents and seas. There are several motivations for geographic distribution of internet services. Namely, higher system availability in cases of regional disasters and proximity to end-users are considered by this work. Both the failure recovery strategy and end-user proximity require redundant data that is consistent and always available. Thus data-replication over the WAN is a key enabler for distributed systems that collectively make World Wide Web.**

To economically meet the objectives noted above; compute, storage, and network resources need to be balanced with consideration of trade-offs between them. Of the three resources, the WAN is a relatively cheap asset. Due to the relative costs, over-building the WAN is often the naive choice with high absolute costs and long lead times [31]. This work contributes a methodology to quantify the WAN demands with consideration of computational and storage nodes with-in the distributed system.

***Index Terms*—wide area network provisioning, replication, backbone networks, network capacity**

I. Executive Summary

Users of web services demand high availability of data in diverse sets of operational environments. Web services respond with databases that provide strongly consistent data for users regardless of their location. An online platform with distributed database architecture is also agnostic to component level failures. In these environments component failures are random and exponentially correlated with the scale of the systems. Compounding the equipment failures is the vulnerability presented when a network link is partitioned, preventing communication between fully functional nodes.

Despite the need, it has been proven that Consistency, Availability, and Partition tolerance (CAP) can not co-exist against in distributed systems by the CAP Theorem [4]. However, proper network planning with sufficient capacity and multi-path redundancy do mitigate the risk of network unavailability and minimize the chances of partitioning (cutting a link).

WANs are not just a diaster recovery mechanism. Strategic provisioning of WAN lifts the overall system performance as well. Reffering to Amdahl's law, which is based on that observation that when a process segment is sped up, the effect on the end to end process time is dependent on how significant the part originally was and the incremental speed gain. Table I lists the latency associated with different components in a distributed systems stack, [8]. At global scale, improvements in network latency can contribute upto 10 factors of magnitude more significance to speed up the end to end process than L1 cache. It is also generally accepted that higher ranking components in the list have higher unit cost per bit of capacity [3]. Given the impact and cost per bit of network, this work aims to quantify the capacity ceiling which alleviates the WAN as constraining factor for distributed systems. Here the several long term planning methods that bin pack multiple communications flows are logically decoupled is explored. Logically decoupled flows, in this context, are service communications that share source and destination points yet their bandwidths are independent variables. In such cases, a collection of flows may have diverging patterns or may simultaneously converge as independent variables.

TABLE I
Process Latency Scale, [8]

Component	**Latency** (ns)
L1 cache reference	0.5
Branch mispredict	5
L2 cache reference	7
Mutex lock/unlock	25
Main memory reference	100
Compress 1K bytes with Zippy	3,000
Send 2K bytes over 1 Gbps network	20,000
Read 1 MB sequentially from memory	250,000
Round trip within same datacenter	500,000
Read 1 MB sequentially from disk	20,000,000
Send packet from CA to Netherlands and back to CA	150,000,000

Considering the CAP theorem and Amdahls law, service operators need to understand their networks. Operators need to sufficiently plan the size of the networks for normal and disaster recovery operations in the face of many failure modes of within the holistic system. This paper surveys several different data base systems and extracts their network dependency. It then evaluates three approaches to network planning based on traces obtained from Wikipedia.

II. Introduction

Web services are transactional and exhibit four key properties [4]. (1) it is expected that transactions commit or fail

entirely (atomic), (2) they always provide consistent data regardless of the location they are executed from, (3) uncommitted transactions are isolated, and (4) once committed they are persistent. For modern internet services, these properties and objectives are achieved with global scale data-replications strategies. The four service level properties are attributed to the three key design objectives listed below.

- Durability: Persistent in case of physical failures.
- Accessibility: Be able to use when needed.
- Performance: Get to it quickly.

Data-replication over the WAN facilitates redundant data to be cast into multiple nodes that are spread across geographically disparate locations. Disparately located redundant data allows a system to tolerate failure modes such as node outages or network partitions, making it an attractive choice when durability and availability of a system are design objectives. Replication strategies are also used to enhance performance of globally distributed internet services. First by allowing faster user access to data with minimum distance optimizations when serving requests. Secondly for distributing the read requests over more nodes and balancing the load on them. The former reduces communication latency and the latter reduces queing latency that would arise if all read requests were sent to a single node. However, when there are multiple copies of the same data there is a risk for them to not be consistent with each other due to communication latencies and failures in the system.

There are three methods for data-replication widely used today and each offers a different degree of trade-off between consistency and latency [1]. The first method has minimal latency where updates are multi-cast to all the replicas simultaneously. However, isolated clients (external consumers of the data-base) may send multiple updates that are received out of order due to communication delays leading to overwrites. The second scheme has updates routed to a designated master node which has the purest and most up-to date view of the data-base. The master node then replicates to all replicas either synchronously or asynchronously. The third scheme sends updates to an arbitrary node, which records that data and make to replicas and users with local access. One shortfall with this approach is upon failure of the arbitrary node, unrecorded and forwarded data maybe lost/

Given the dependence on replication for internet service and it's sensitivity to latency, it is crucial that WANs be properly provisioned with sufficient capacity and redundancy in order to negate congestion that leads to delays. In the motivational work section, several modern data-base systems and their replication strategies are reviewed. The review finds that the published DB system present an abstracted view of their network dependency. Therefore physical implementation of the network architecture may be proprietary and distinct from each other. To generalize the distinct implementations, the Background section first a develops a hierarchical clos topology to provide intuition about the rest of the paper. The Methodology section presents several approaches to estimating bandwidth requirements. These methodologies are then evaluated and compared to each other for a typical uses case in the results sections. This paper concludes with the Conclusion section. For brevity, this work only considers a single class of network traffic.

III. Motivational Works

Wide area networks serve as the backbone for many internet applications that operate across regional data-centers. The traffic on the inter data center links are segregated into three classes; interactive, large file transfers, and background [30]. Interactive traffic are blocking operations; where the byte level transactions are required to keep things proceeding. Large file transfers entail transmission of requested resources by some deadline. Background traffic are workloads that are opportunistically using the link, and preempting them does not expose the system of any idempotent side effects. DB communications traffic typically consists of workloads that are bound by deadlines and also pseudo-interactive. This section presents a sample of specific database system that serve as back-ends for many modern online platform architectures.

Megastore was the first large-scale system to use Paxos to replicate primary user data across regionally distributed data-centers for every write [13]. Megastore enables applications to have fine grained control for partitioning and locality by Entity Groups. Several entity groups in Megastore are provisioned across a set of data-centers. Locally, the distinct entity groups have loose consistency requirements however common entity groups across data-centers maintain atomic, consistent, isolated, durable (ACID) data over the WAN. Megastore relies on the underlying Bigtable [2] instances at each of Google's data-centers. Megastore implements a multiple version concurrency control schema, that allows DB fields to have multiple versions isolated by time-stamps. This eliminates interference between reads and writes, as reads will be fetched from the most completed time-stamp DB. Megastores has several types of logical actors. Namely they are leader, full-replicas, witness, read-only. Witness actors vote in Paxos rounds and store write ahead logs, but are thrift on storage for entities. Read only actors provide present views of the data based on a near real time snapshot.

Spanner is a gloablly distributed database in which writes are committed with the Paxos protocol at single leader node. Reads access the data directly from any replica that is sufficiently up-to-date, which can have service garantees in terms of consistency classes. The set of replicas is collectively a Paxos group [17]. As a globally distributed database, Spanner provides several interesting features, [17]. These features faclitate replication configurations to be dynamically controlled at by applications with . First, applications can specify constraints to control which data centers contain which data. Second, it allows application owners to specify how far data is from its users (to control read latency). Third, application owners to delegate how far replicas are from each other (to control write latency). Finally, applications can control how

many replicas are maintained (to control durability, availability, and read performance).

In the Google Cloud Platform Console, the most economical multi-region Spanner instance, nam3, notes the following specs. Nam3 consists of 3-way redundancy with-in North America. It provides an availability service level objective (SLO) target of 99.99%. In contrast to a single region Spanner instance which yields a 99.9% SLO.

Spanner Guidelines

- Each Cloud Spanner node in nam3 can provide up to 7,000 QPS (queries per second) per region, or 1,800 QPS of writes across all regions for writing 1 kB data per row with total storage capacity of 2TiB.
- For optimal performance in this configuration, it is recommended that CPU utilization be maintained at at or below 45%.
- A minimum of 3 nodes is recommended for production environments; ie user facing.
- Cloud Spanner's performance is highly dependent on workloads, SQL schema design, and data-set characteristics.

Cassandra is a NoSQL database open sourced by Facebook [5]. Cassandra's architecture is based on a coordinator paradigm, where the consistency level is user defined and a coordinator node is responsible for forks and joins of the data to yield either a single view, quorum view ($(\frac{Nodes}{2}+1)$), or all views of the data. Each node in the Cassandra architecture is configured to directly communicate with clients. To illustrate, clients reading some data connect with a single Cassandra node. Upon receiving the request, the Cassandra node presumes the coordinator responsibility and retrieves data from its local storage and/or coordinates its retrieval based on the consistency level required by the request.

MongoDB Atlas is a open sourced MongoDB NoSQL database offered on the cloud [14]. Cloud distributed Atlas consists of a set of upto 50 replicas with one primary replica member and the balance serve in secondary replica member roles. Upon a failure of the primary, one of the secondary members is automatically upgraded to primary based on an election process. The promotion of a secondary node to primary takes on the order of a few seconds, at which point clients direct their connection to the new primary for writes. Reads are still facilitated at all available nodes without disruption. Atlas' primary election process is implemented with the Raft consensus protocol [15]. First, the protocol evaluates replica based on their state, biasing towards replicas that have the most recent updates applied. Second, Atlas' RAFT checks the heartbeat and connectivity status of a quorum of replica set members. Finally, the algorithm allows administrator to configure a deterministic preference of the replica set to promote. With a replicas promoted all secondary replicas redirect their connection to it.

In MongoDB Atlas data is horizontally spread across physical servers by using sharding techniques. Sharding of values in a data set can be done locally with-in a single data-center, or distributed globally across the replica-sets. Each shard has a replica set consisting of primary and secondary members as shown in 1. The shards are made transparent to clients with the use of local "Query Routes" as an abstraction layer. When reading, the default is to read from the primary member but users can set reads to be from the nearest replicas for minimum latency at the cost of consistency. Writes can be acknowledged from the primary only, a deterministic count of replicas, the quorum of replicas, or all replicas. There is an option for completely unacknowledged writes. To relieve the burden on the network demand, application layer data is compressed upto 80% for transport across the network. Users can specify weight for writes to specific data-centers and apply higher relative weights to Primary DC's.

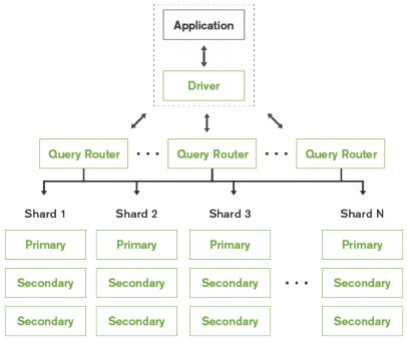

Fig. 1. MongoDB Atlas Sharding Scheme [14]

To appreciate the complexity of distributed systems quantitatively, the Storage Configuration Compiler (SCC) provides a broadview of the interaction between various componets of a distributed storage system [16]. The Storage Configuration Compiler is a tool that takes an application's service level objectives, storage hardware parameters, and compute server parameters along with their costs to suggest optimum cluster configurations. At the application level SCC separates tasks and data sets. Data-set accessibility is treated as a non-deterministic variable and each operation is modeled as a probability function. It is shown explicitly in SCC that network delays hang-up machine resources waiting for responses, wasting expensive resources.

Storage I/O throughput definitions for SCC:

- read/write gap parameter for non-sequential seeks.
- $\frac{size}{rate}$ + gap is the latency to serve

I/O operations definition for SCC:

- Number of records read in an I/O operation
- Number of records written in an I/O operation
- Bytes count of each record
- Are the reads parallel

Task Dependencies:

- The count of invocations being performed
- are the invocations in parallel
- is the whole dependency block or non-blocking

Capacity Parameters:

- data set size

- does it need to be persistent
- local or remote

IV. Background

This section provides context about the current industry trends. The next paragraph argues that local costs optimization increases the reliability burden for global presence. The following subsection then presents the bounds of data communication networks is presented from the data-center level to the wide area networks that connect data centers.

The current data center market has two paradigms for cost savings, one is hardware based and another infrastrucutre based. The hardware based paradgim consists of the mentatlity that more of cheaper hardware are better than less of expensive ones. Parrallelsim is increased with more hardware node counts leading to faster end to end processes, compared to using few high cost machines yeild a performance/$ efficiency gains. More hardware counts do allow more local redundanency. Replication at locally housed servers increases data availability in the presence of individual server failures, but only with diminishing returns [13].

The infrastructure costs savings are based on relaxing tried and trued industrial practices, examples being free air cooling that introduces contamination or removing UPSes from the power delivery systems. The utility scale systems such as power, cooling, and long haul tele-communication networks introduce additional risk that is not offset by local replication. It has been pointed out that hyperscale internet data-centers are susceptible to construction value engineering that risk some level of facility-wide failure modes [29].

A. Context Setting

To motivate interest in the rest of this paper, let's visit some typical conisderations that are encountered in industry. First, applications need to understand their distributed process latencies. In the absense of network congestion, the ideal elapsed time (t_e) to transfer B, bytes, to R, replicas, is shown in equation 4. T is the network throughput rate in bytes per unit of time and L is latency to transfer bytes between two machines [10]. Network links are typically on the order of O(xx)-Gbps (T), and L is far below 1 ms as seen for component level contributions in Table I. Therefore, 1 MB can ideally be distributed in about 80 ms.

$$t_e = B/T + RL \tag{1}$$

Second, applications need to understand the statistical distribution of their communication capacities. To demonstrate this, it is most intuitive to consider a local data-center environment as seen by a server on a rack. Maximum bandwidth that server can send out of a rack is limited by the top of rack (TOR) outbound channels. As an example, if a rack consists of 35 servers, each connected to the TOR at 10 Gbps channels. With a TOR capacity of 100 Gbps to its external adjacencies, only 10 of the servers can sustain full bandwidth. Equation 2 indicates the limit of the flows from all servers in a rack must be equal to or less than the TOR's external facing capacity. However, servers with-in a rack can also communicate with each other and not be limited by the TOR's external limits. An depiction of the complete topology is shown in 2.

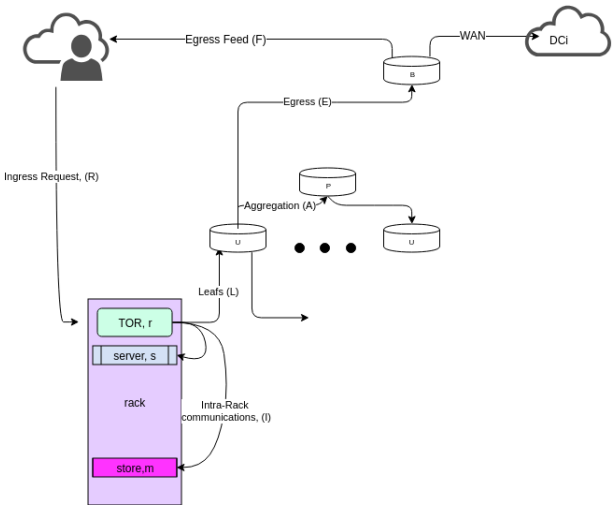

Fig. 2. Request Ingress and Responses from rack perspective.

Link saturation at rack level:

$$TOR \geq \sum I_i \tag{2}$$

Where:

TOR: Rack level ingress and egress bandwidth to external adjacency.

I_i: Server level ingress and egress network bandwidth.

V. Methodologies

In this section three forecasting methods are are discussed. The methods are namely historical based, transaction based, and power based. Wikipedia page view data is used to demonstrate the application of each method presented. The approach described in this section resemble those used by aggregate network planners, where numerous independent services are bin packed into a network flow from a source node to a destination node. As is typically the case for network operators responsible for the Cloud, where service level network benchmarks are not exposed at time of planning.

Having benchmarks that correlate to other resource dimensions may provide more robust signals for network capacity demands earlier. In a simple example, workers (computers) are required to generate network traffic. If benchmarks that correlated computer counts with a network limits, the correlation can be used to provision network capacity preemptively as more computers were ordered but not set in production.

The problem is compounded further by the independent nature of the services. Network traffic is known to have Markovian properties [11]. Their simultaneous behavior is at best another layer of probabilistic functions.

In Cloud enviroments agility is also a key criterion [17]. First agility is needed when new clients are on-boarded. Secondly, new features that increase network efficiency or new use cases of network are continuously added. Finally, the

decoupling of physical resources from logical resources have enabled container type, elastic workloads. The containers not only dynamically change resource dimensions but can also physically migrate across the globe without much effort.

Given the above, the coarse approaches explored in this work are sufficient to forecast IP network traffic in the leads times that operators are bound to. Specifically, the method used to identify the source of the traffic for the Wikipedia page view is heuristics based, yet since it is non-proprietary it suffices for academic research and demonstrates suitability for industrial research.

A. Data Set

Traffic to 145,063 Wikipedia Pages for 803 days from July 2015 through September 2017 is used demonstrate the scale of replication required for global services. The data will be used to train and test the models developed here. The training set will be partitioned to be inclusive of traffic from July 2015 to December 2016. The balance of the data will be used to test accuracy of the models. The pages are segregated into 7 different languages; namely English, Japanese, German, French, Chinese, Russian, and Spanish. The average volume of visits per language is shown in Figure 3. Figure 4 shows the distribution of the 95 percentile usage values per respective language.

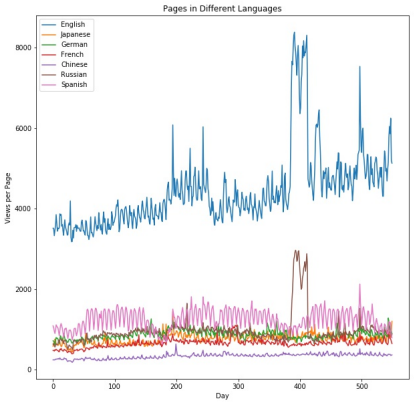

Fig. 3. Average Daily Visits to Study Set of Pages by Language

After segregation of the pages by language and dropping pages without language indicators, the languages are mapped to countries in which they are the official tongue. Figure 5 shows the mapping of languages to their respective countries by color groups; English(blue), Japanese(dark gray), German(yellow), French(red), Chinese(green), Russian(orange), and Spanish(magenta). Red markers indicate the locations of Wikipedia data centers from around the world as listed in Table II. Ashburn has the primary (P) data-center for application services, and Carrollton serves as the redundant (R) node. The remaining 3 sites are smaller edges of the Wikipedia network where user traffic ingresses and egresses Wikipedia network.

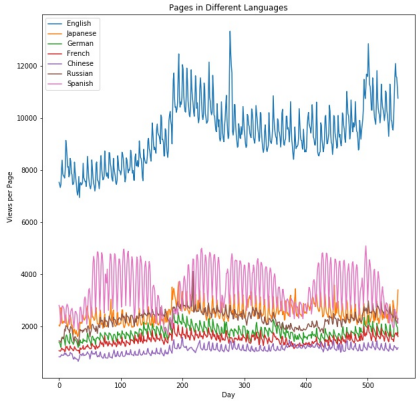

Fig. 4. 0.95 Quantile Visits to Study Set of Pages by Language

Throughout this paper the associated set of Wikipedia data-centers will be used as the model network and several assumptions will be called out in regards to attributes of the system that are not in the public domain. Furthermore, some of the system complexity will be reduced for brevity and academic objectives of this work. The traffic generating in a particular country is associated with their closest ingress points as listed in Table III by using a minimum distance function of the respective geographical codes. This table shows that English is the most dominant language with traces appearing at all of Wikipedia's ingress sites. English is the most frequently searched language also (see figure 3), here it is shown to also be the most distributed globally.

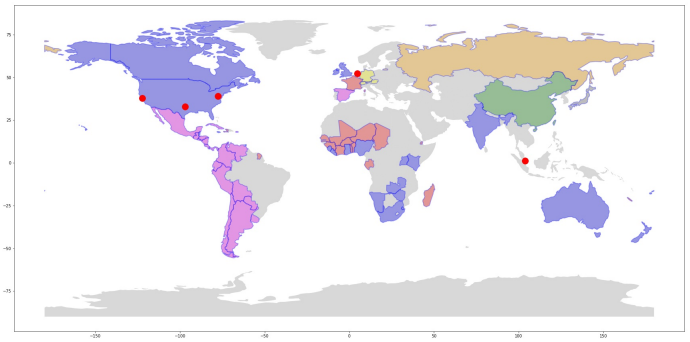

Fig. 5. Countries where set of languages for Wiki Page is the primary language

No power capacity values for 'eqsin' data-center was exposed by Wikipedia. Therefore the power demand for the Singapore data-center has been estimated by doing a qualitative similarity analysis. The time-series plot of the hit rate at each of the ingress data-centers results in Figure 6. Singapore's average page views per day visually appears to dominate other data-centers for ingress traffic, yet it is a caching site so is only comparable to other caching sites. Haarlem generally tracks Singapore well, with only slight divergence during their

TABLE II
WIKIPEDIA NETWORK NODES

	ID	Address	Power(kW)	Type
1	eqiad	Ashburn, Virginia, USA	130-kW	Applications-P
2	codfw	Carrollton, Texas, USA	77-kW	Applications-R
3	esams	Haarlem, Netherlands	$<$ 10-kW	Caching
4	ulsfo	San Francisco, CA, USA	$<$ 5-kW	Caching
5	eqsin	Singapore	10-kW	Caching
6	eqord	Chicago	-	Transit
7	eqdfw	Dallas	-	Transit
8	knams	Amsterdam	-	Transit

TABLE III
LANGUAGES TO INGRESS SITES

DC	en	ja	de	fr	zh	ru	es	**total**
Ashburn	18	-	-	5	-	-	8	**31**
Carrollton	2	-	-	-	-	-	11	**13**
Haarlem	17	-	4	16	-	-	1	**38**
San Francisco	3	-	-	1	-	-	-	**4**
Singapore	9	1	-	4	5	1	-	**20**

peak traffic period. When quantified, the average hit rate of Singapore is 3% higher than Haarlem. The traffic allocation approach is indicated in Table IV. Given the uncertainty of ¡ 10kW conditional value provided by Wikipedia for Haarlem, Singapore is also determined to have a 10-kW demand.

TABLE IV
METHOD FOR TRAFFIC ALLOCATION TO EACH INGRESS SITE

Objective:	Determine the hit rate of each ingress site.
Step 1:	Map language to page.
Step 2:	Map languages to countries where they are the first tongue.
Step 3:	Map ingress sites to countries with minimum distance function.
Step 4:	Get distribution of number of countries served by each ingress site.
Step 5:	For each ingress site, sum up the daily views of pages by the language. Multiply to relative fraction from step 4 - origin countries with language to site / total number of (Count countries with language).

In Table II there are three transit sites in the Wikipedia network also. These are considered as passive network nodes, in that no processing or storage is facilitated at the sites. Figure 7 illustrates Wikimedia's autonomous network system. The nodes are per Table II and the colored lines indicate discrete network connections, annotated with their carrier and the latency. The adjacency matrix of source and destination locations as vertices is shown in Table V

The data-set from Kaggle does not provide an explicit flag indicating read or writes. In this work, all of the access tasks listed in Kaggle are considered to be reads. The write values are indicated by the content generated as a percentage of total Wikipedia article count in 2016.

B. Historical Extrapolation

In this approach, the past observation is indicative of future demand.

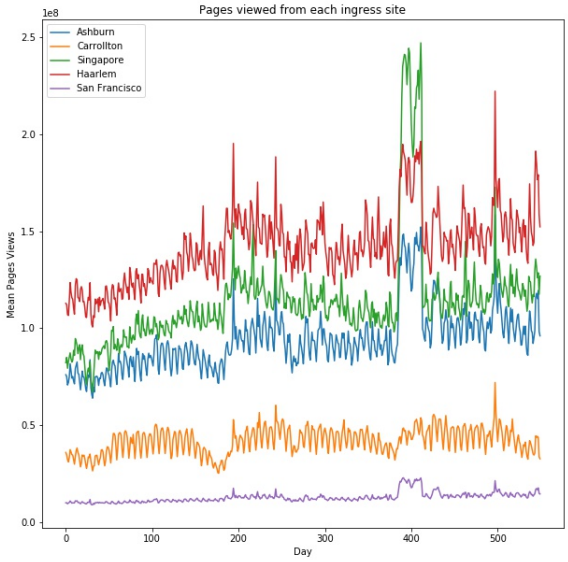

Fig. 6. Traffic based on source country and its min(cost=distance) ingress site.

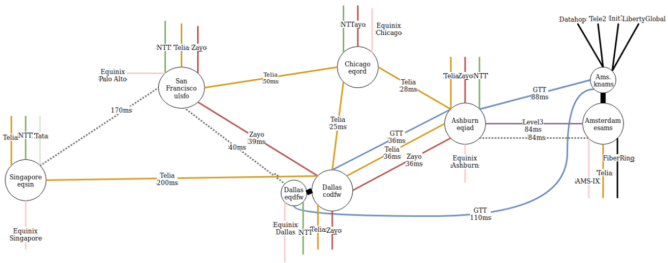

Fig. 7. Wikimedia's AS14907 Network. Source [24]

TABLE V
DC ADJACENCY MATRIX

	EQIAD@	ESAMS@	EQORD@	KNAMS@	CODFW@	ULSF@	EQSIN@	EQDFW@
EQIAD	-	2	1	1	3	-	-	-
ESAMS	2	-	-	1	-	-	-	-
EQORD	1	-	-	-	1	1	-	-
KNAMS	1	1	-	-	-	-	-	-
CODFW	3	-	1	-	-	1	1	1
ULSFO	-	-	1	-	1	-	1	1
EQSIN	-	-	-	-	1	1	-	-
EQDFW	-	-	-	-	1	1	-	-

Segregate data by consumer and application Use existing capacity as baseline and compare to usage? *In a system like BwE usage may not represent demand, as flows get throttled at the TCP socket Usage data sampling Get daily statistical representative. Level the daily data across the time range of study. Historical data is required for a representative range of time. Outliers n the data must be filters out.

Future projections are based on uniform multipliers by consumer or application. Consumer level forecasts are more robust than service due to spill over economic effects

C. Event based

$$F_l = f(src, dst)_a \tag{3}$$

$$D_{l_i} = \sum src_i * dst_j \tag{4}$$

$$\begin{aligned} D &= demand \\ F &= flow \\ _l &= language \\ _i &= ithsource \\ _j &= jthsource \\ src &= source \\ dst &= destination \\ a &= arc/between/srcanddst \end{aligned}$$

In this section, the Wikipedia traffic is correlated with data replication network demands. The spikes in Russian queries and the more pronounced English queries are seen around August 2016. Which does raise an interesting political connotation to the Trump election campaign. Regardless of the election coincidence, the Wikimedia Foundation attributes the anomaly to bugs in Chrome 41 and Windows 10/7 that inflated the Homepage statistics.

With the ojective to determine the network requirements, the number of page views is converted to network flow rates first. The method used in this work determines the page size based on language. The language based sizes are upper bounded by Wikipedia's limit per page of 2048 kilobytes. Files about a subject that exceed this maximum value are split into multiple pages leading to a wider distribution of pages sizes [26].

The size distribution of Wikipedia articles is not readily avaiable. Several resources were consulted to determined the page size [21], [27], [28]. The page size is determined as noted in Table VII. Wikipedia Statistics site provided an incomplete set of data for the languages of interest [27]. In the Table the average English article consists of 640 words [21]. Aplhabetic characters translate to 1 Byte of data. As an example, English words have an average length of 8.23 characters by equality they are 8.23 Bytes long. The values obtained in Table VII were compared with the historical trends from [27]. The historical trend is shown in Figure 8. These values are spot checked for a small sample of articles at Xtools, a site for Wikipedia meta data [28].

We can now sum up the packet counts also. The IP packets can be transmitted in 64k-Bytes sizes, however due to limitations of hardware the maximum size of packets are limited to be as low as 1500 Bytes [19]. Packets also carry some overhead that don't scale with packet size, so net payload is lower.

TABLE VI
TYPICAL BYTE OF C LANGUAGE DATA TYPES [3]

C declaration		Bytes	
Signed	Unsigned	32-bit	64-bit
[signed]char	unsigned short	1	1
short	unsigned short	2	2
int	unsigned	4	4
long	unsigned	4	8
int32_t	uint32_t	4	4
int64_t	uint64_t	8	8
char*		4	8
float		4	4
double		8	8

TABLE VII
LANGUAGES TO PAGE SIZE CLASSIFICATION

ID	Count	Language	Word length	Words/ page	Bytes/ character	Bytes/ page (ave)
zh	17229	Mandarin	1.00	1032	2	2064
fr	17802	French	10.09	640	1	6457
en	24108	English	8.23	640	1	5267
ru	15022	Russian	9.97	640	1	6380
de	18547	German	11.66	640	1	7462
ja	20431	Japanese	1.00	1600	2	3200
es	14069	Spanish	8.80	640	1	5632

D. Power Allocation Method

The power allocation method is an extension of the Gravity Model as proposed for capacity planning Internet backbone traffic in [9]. Strength of the interaction (S) between two cities is proportional to the product of their populations (P) divided by the distance between them squared (d), See equation 5. It has been proven accurate for estimating telephone traffic exchange between area codes.

$$S = \frac{P}{d^2} \tag{5}$$

Data-center capacity is typically measured in terms of nominal power supported by the facility [29]. For performance, power translates linearly to the computational capability of the system for a given set of hardware architecture. Here nominal power values are considered in a network flow directed graph,

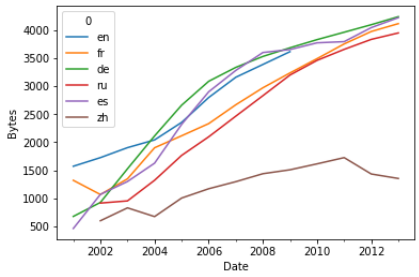

Fig. 8. Wikipedia Article Sizes in Bytes. Source [27]

where the data-centers are the nodes (vertices) and the network communication link between two nodes are edges (arcs) of the graph. The rest of this section describes a power allocation method for establishing node and edge weights.

Consider the network of Wikipedia data-centers listed in Table II. The caching sites consists of front-end services, they only store fresh or the most popular copies of data for the local user base (see Figure 5 and III). The data is presumed to be read only at the cache sites, and all writes are done in the primary application site.

The average power use for each of Wikipedia's data-centers is shown in Table VIII. The power sums to 232-kW for all the data-centers. Notice that EQIAD constitutes 56% of the total power since all data is processed at there. This is meaningful for network topology in that all data not cached locally at the ingress data-centers are sourced there. This drives the demand for sufficient bandwidth between EQIAD and all other sites for reads. Additional write traffic has not been accounted for in the Kaggle data-set, it will be factored in as a multiplier based on the the language specific corpus growth rates for this time period.

TABLE VIII
POWER DISTRIBUTION OF DCS

	avg_power_(kW)	fraction_power
EQSIN	10	0.04
ULSFO	5	0.02
EQDFW	0	0.00
CODFW	77	0.33
EQORD	0	0.00
EQIAD	130	0.56
ESAMS	10	0.04
KNAMS	0	0.00

Total average power of the data-center network.

$$P_T = \sum P_{dc_i} \tag{6}$$

$$p_{dc_i} = \frac{P_{dc_i}}{\sum P_{dc_i}} \Rightarrow \frac{P_{dc_i}}{P_T} \tag{7}$$

$$BW_T = \sum BW_{l_i} \tag{8}$$

$$bw_{l_i} = \Big(\frac{P_{dc_a} + P_{dc_z}}{P_T}\Big) BW_T \Rightarrow \big(p_{dc_a} + p_{dc_a}\big) P_T \cdot BW_T \tag{9}$$

Math consider total system power -

Classes of Network QoS

Since nominal power represents a specific set of hardware architecture, the link requirement may change as the hardware evolves.

In the public domain, only the average power values have been found for Wikipedia's data-centers. To apply the power attribution method, power and traffic are considered to have a linear correlation. In this work mean traffic values are used as the controlled variable, where average power is the control variable.

The size of the data table differ. tables with mean data were 8.6 KB and 95% quantile data were 28.6 KB

VI. DISCUSSION

The memory allocation of the data tables differ by more than 3 times. For example the dataframe with mean data was 8.6 KB and dataframe with 95% quantile data was 28.6 KB for the ingress traffic series. However negligible trade-off is shown with time complexity. Time complexity is a metric that provides the relative execution time of an algorithm.

TABLE IX
QUALITATIVE EVALUATION MATRIX

Model	**View**	**Objective**	**Use Case**
Historical	Non-Markovian	Conservative	Organic Growth
Transaction	Fine Grained	Precision	Inorganic Growth
Power	Coarse	Long Term Stability	Physical Expansion

VII. CONCLUSION

Temporal stability Long Term Scale Random Samples

VIII. FUTURE WORK

This work sets the foundations for an environmental footprint of global scale data center networks by establishing the logical dependencies between distributed facilities. The power demands of the Wikipedia data centers are negligible compared to the hyperscale, content rich services such as search engines and social media. However, the methodology presented here can scale to fit the hyperscale sites.

Each node has a a environmental vector. The values in the connection matrix indicate the degree of connection between two sites, where degree is the dependency for a pair of sites. With this approach, the redundancy and communications overheads head can be fully captured.

TABLE X
WEIGHTED ENVIRONMENTAL VECTOR MATRIX

	EQIAD@	ESAMS@	EQORD@	KNAMS@	CODFW@	ULSF@	EQSIN@	EQDFW@
EQIAD	-	2	1	1	3	-	-	-
ESAMS	2	-	-	1	-	-	-	-
EQORD	1	-	-	-	1	1	-	-
KNAMS	1	1	-	-	-	-	-	-
CODFW	3	-	1	-	-	1	1	1
ULSFO	-	-	1	-	1	-	1	1
EQSIN	-	-	-	-	1	1	-	-
EQDFW	-	-	-	-	1	1	-	-

REFERENCES

[1] D. J. Abadi, "Consistency Tradeoffs in Modern Distributed Database System Design: CAP is Only Part of the Story," in Computer, vol. 45, no., pp. 37-42, 2012. doi:10.1109/MC.2012.33

[2] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber, 2006. Bigtable: A Distributed Storage System for Structured Data Google Inc

[3] Randal E. Bryant and David R. O'Hallaron, Computer Systems: A Programeers Perspective. Pearson 2015 ISBN: 978-93-325-7390-1

[4] Seth Gilbert and Nancy Lynch. 2002. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News 33, 2 (June 2002), 51-59. DOI: https://doi-org.proxy.library.cmu.edu/10.1145/564585.564601

[5] S. Dipietro, G. Casale, G. Serazzi, 'A Queueing Network Model for Performance Prediction of Apache Cassandra', VALUETOOLS'16 proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools on 10th EAI International Conference on Performance Evaluation Methodologies and Tools, Pages 186-193

[6] Haskel, Jonathan; Westlake, Stian. 2018 Capitalism without Capital - The Rise of the Intangible Economy, Princeton University Press.

[7] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall and W. Vogels, 'Dynamo: Amazons Highly Available Key-value Store',SIGOPS Oper. Syst. Rev., vol. 41, no. 6, pp. 205-220, December 2007.

[8] Jeff Dean, "Designs, Lessons and Advice from Building Large Distributed Systems", url: http://iepg.org/iepg/2009-11-ietf76/dean-keynote-ladis2009.pdf, 2009, Accessed 01/18/2019

[9] Matthew Roughan, Albert Greenberg, xperience in Measuring Backbone Traffic Variability: Models, Metrics, Measurements and Meaning, 2002 ACM, W'02, Nov. 6-8, 2002, Marseille, France

[10] K. McKusick and S. Quinlan, 'GFS: Evolution on Fast-forward', Commun. ACM, vol. 53, no. 10, pp. 42-49, March 2010.

[11] Mor Harchol-Balter Performance Modeling and Design of Computer Systems: Queueing Theory in Action, Cambridge University Press, 978-1-107-02750-3, 2013

[12] M.Bertoli, G.Casale, G.Serazzi. JMT: performance engineering tools for system modeling. ACM SIGMETRICS Performance Evaluation Review, Volume 36 Issue 4, New York, US, March 2009, 10-15, ACM press

[13] Jason Baker, Chris Bond, James, Corbett, 2011. 'Megastore: Providing Scalable, Highly AvailHarchol-Balterable Storage for Interactive Services', 5 th Biennial Conference on Innovative Data Systems Research

[14] MongoDB, 'MongoDB Multi-Data Center Deployments A MongoDB Whitepaper', November 2017, url:$http://s3.amazonaws.com/info-mongodb-com/MongoDB_Multi_Data_Center.pdf$, accessed 1/25/2019

[15] Diego Ongaro and John Ousterhout, In Search of an Understandable Consensus Algorithm (Extended Version), Stanford University, 2014

[16] H. Madhyastha, J. McCullough, G. Porter, A. Vahdat. 'scc: Cluster Storage Provisioning Informed by Application Characteristics and SLAs', FAST'12 10th USENIX Conference on File and Storage Technologies, 2012

[17] J. Corbett, J. Dean, M. Epstien, et.al, 'Spanner: Googles Globally Distributed Database' ACM Trans. Comput. Syst., vol. 31, ACM New York: Academic, August 2013, pp. 8:1–8:22.

[18] Google Cloud Platform, https:cloud.google.comspannerdocsinstances#available-configurations-multi-region, accessed 1/30/2018

[19] Tanenbaum, Andrew S. and Wetherall, David J., 'Computer Networks 5th ed.', 2010, isbn = 0132126958, 9780132126953, publisher = Prentice Hall Press, address = Upper Saddle River, NJ, USA

[20] Kaggle: Web Traffic Time Series Forecasting, Forecast Future traffic to Wikipedia Pages, url:https://www.kaggle.com/c/web-traffic-time-series-forecasting (Accessed 12/09/2018)

[21] https://en.wikipedia.org/wiki/Wikipedia:Size_comparisons, (Accessed 12/22/2018)

[22] https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia, (Accessed 12/22/2018)

[23] Wikipedia Analytics https://analytics.wikimedia.org/dashboards/vital-signs/#projects=eswiki,itwiki,enwiki,jawiki,dewiki,ruwiki,frwiki/metrics=Pageviews, (Accessed 12/22/2018)

[24] Wikimedia servers https://wikitech.wikimedia.org/wiki/Network_design, (Accessed 12/21/2018)

[25] Wikimedia servers https://meta.wikimedia.org/wiki/Wikimedia_servers, (Accessed 12/20/2018)

[26] https://www.mediawiki.org/wiki/Manual:$wgMaxArticleSize, (Accessed 12/22/2018)

[27] Wikipedia Statistics - Bytes per Article https://stats.wikimedia.org/EN/TablesArticlesBytesPerArticle.htm, (Accessed 12/30/2018)

[28] Xtools, https://xtools.wmflabs.org/articleinfo, (Accessed 12/30/2018)

[29] Luiz Barroso, Urs Hölzle, and Parthasarathy Ranganathan, 2018. The Datacenter as a Computer: Designing Warehouse-Scale Machines, Third Edition, Morgan & Claypool Publishers. https://www.morganclaypool.com/doi/pdf/10.2200/S00874ED3V01Y201809CAC046

[30] H. Zhang et.al, "Guaranteeing Deadlines for Inter-DataCenter Transfers", IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 25, NO. 1, FEBRUARY 2017

[31] Zhenyun Zhuang, Haricharan Ramachandra et al, Capacity Planning and Headroom Analysis for Taming Database Replication Latency-Experiences with LinkedIn Internet Traffic, ICPE15, Jan. 31Feb. 4, 2015, Austin, Texas, USA

Eric is a Product Area Network Manager for Google Product Infrastructure at Google Inc. He provides WAN capacity planning and management for Search, Photos, Chrome, Geo-Local, Hardware, and Mobile Products along with a long tail of other services. He is also a self funded Doctors of Professional Practice Candidate at Carnegie Mellon University's School of Architecture with a focus on sustainability of internet scale data-center infrastructure. His research curiosity is driven by the systems inter-actions that are enabling the connected world.