PNUTS: Yahoo!’s Hosted Data Serving Platform

Presented by: Gaurav Vaidya

Some of the slides in this presentation have been taken from http://www.cse.iitb.ac.in/dbms/Data/Courses/CS632/Talks/pnuts-vldb08.ppt
INTRODUCTION
How do I build a cool new web app?

• Option 1: **Code it up! Make it live!**
  – Scale it later
  – It gets posted to slashdot
  – **Scale it now!**
  – Flickr, Twitter, MySpace, Facebook, …
How do I build a cool new web app?

- Option 2: **Make it industrial strength!**
  - Evaluate scalable database backends
  - Evaluate scalable indexing systems
  - Evaluate scalable caching systems
  - Architect data partitioning schemes
  - Architect data replication schemes
  - Architect monitoring and reporting infrastructure
  - *Write application*
  - Go live
  - **Realize it doesn’t scale as well as you hoped**
  - Rearchitect around bottlenecks
  - 1 year later – ready to go!
Example: social network updates

What are my friends up to?

Sonja:

Brandon:
Example: social network updates

- 6 Jimi
- 8 Mary
- 12 Sonja
- 15 Brandon
- 16 Mike
- 17 Bob
Consistency Example

Photo Sharing List

- Mom
- John

Remove

Photo Sharing

Album:
Spring Break Party
Timeline consistency

Node 1
- Share photos
- Remove user

Node 2
- Remove user
- Share photos
Features needed for web-apps

- Scalability
- Response Time and Geographic Scope
- High Availability and Fault Tolerance
- Relaxed Consistency Guarantees
PNUTS in a nutshell

It is a
  - massively parallel
  - geographically distributed
  - database system for Yahoo!’s web applications.

It is a hosted & centrally managed service
PNUTS in a nutshell

- Data storage organized as hashed or ordered tables
- Low latency for large numbers of concurrent requests including updates and queries
- Per-record consistency guarantees
Contributions

- Record-level, asynchronous geographic replication
- A consistency model that offers applications transactional features but stops short of full serializability.
- A careful choice of features
  - include (e.g., hashed and ordered table organizations, flexible schemas) or
  - exclude (e.g., limits on ad hoc queries, no referential integrity or serializable transactions).
- Data management as a hosted service
FUNCTIONALITY
PNUTS Specifications

- Data Model and Features
  - Simple relational model
- Fault Tolerance
- Topic-based pub/sub system
  - Yahoo! Message Broker (YMB)
- Record-level Mastering
- Hosting
Data and Query Model

- Data is organized into tables of records with attributes
  - hashed / ordered tables
- The query language of PNUTS supports selection and projection from a single table.
- **point access**: A user may update her own record.
- **range access**: Another user may scan a set of friends in order by name.
- PNUTS also does not enforce constraints such as
  - referential integrity
  - complex ad hoc queries (joins, group-by, etc.).
Hiding the Complexity of Replication

per-record timeline consistency: all replicas of a given record apply all updates to the record in the same order

The sequence number

- generation of the record (each new insert is a new generation)
- version of the record (each update of an existing record creates a new version).

Note that we (currently) keep only one version of a record at each replica
API calls

- Read-any
  - Stale versions
- Read-critical (required version)
- Read-latest
- Write
  - Single ACID operation
- Test-and-set-write (required version)
  - Concurrent writes
Bundled updates

Relaxed consistency: Allow applications to indicate, per-table, whether they want updates to continue in the presence of major outages, potentially branching the record timeline
Notifications

- Trigger-like notifications are important for applications e.g.: Ad – Serving
- allow the user to subscribe to the stream of updates on a table
SYSTEM ARCHITECTURE
Architecture

Clients

REST API

Tablet controller

Routers

Message Broker

Storage units

Data-path components
Each storage unit has many tablets (horizontal partitions of the table). Storage unit may become a hotspot.

Overfull tablets split

Tablets may grow over time

Shed load by moving tablets to other servers
Architecture

Local region

Remote regions

Clients

Routers

REST API

YMB

Storage units
Accessing Data – Ordered tables

Key k divided into intervals

1. Get key k
2. Get key k
3. Record for key k
4. Record for key k
Accessing Data – Hash Tables

n bit Hash Function $H(k)$
$0 < H(k) < 2^n$

Divided into intervals

1. Get $H(k)$
2. Get $H(k)$
3. Record for $H(k)$
4. Record for $H(k)$

SU  SU  SU
Updates

1. Write key k
2. Write key k
3. Write key k
4. SUCCESS
5. Write key k
6. Write key k
7. Sequence # for key k
8. Sequence # for key k

Routers

Message brokers
Replication and Consistency

Yahoo Message Broker

- Data updates are considered “committed” when they have been published to YMB
- YMB guarantees message delivery
- Logs the updates
- PNUTS clusters saved from dealing with update propagation
- Provides partial ordering
Record Level mastering

- One replica becomes a master copy
- 85% writes to a record originate from the same datacenter
- Master propagates updates to other replicas
- Mastership can be assigned to other replicas as needed
  - Eg: When a change in user’s location is detected
- Every record has a hidden metadata field storing the identity of the master
Router Failure

- Routers contain only a cached copy of the interval mapping
- The mapping is owned by the tablet controller
- If a router fails, we simply start a new one
Recovery

- Involves copying lost tablets from another replica
- The tablet controller requests a copy from a particular remote replica
- "checkpoint message" is published to YMB, to ensure that any in-flight updates at the time the copy is initiated are applied to the source tablet.
- The source tablet is copied to the destination region
Other Database System Functionality

- Query Processing
  - Multi-record requests
  - Range Queries

- Notifications
  - Notifying external systems on updating certain records
  - Subscribe to the topic for specific tablet
PNUTS APPLICATIONS

- User Database
- Social Applications
- Content Meta-Data
  - Eg: email attachments
- Listings Management
  - Eg: Comparison shopping
- Session Data
Experimental setup

- Production PNUTS code
  - Enhanced with ordered table type

- Three PNUTS regions
  - 2 west coast, 1 east coast
  - 5 storage units, 2 message brokers, 1 router
  - West: Dual 2.8 GHz Xeon, 4GB RAM, 6 disk RAID 5 array
  - East: Quad 2.13 GHz Xeon, 4GB RAM, 1 SATA disk

- Workload
  - 1200–3600 requests/second
  - 0–50% writes
  - 80% locality
Scalability

![Graph showing scalability with average latency in milliseconds vs storage units. The graph compares hash table and ordered table performance.]
Request skew

![Graph showing the relationship between Zipf parameter and average latency for hash table and ordered table.](image-url)

- **Y-axis**: Average latency (ms)
- **X-axis**: Zipf parameter
- **Legend**:
  - Hash table (diamonds)
  - Ordered table (squares)

The graph illustrates how the average latency changes with varying Zipf parameters for both hash tables and ordered tables.
Size of range scans

![Graph showing the relationship between the fraction of table scanned and average latency in milliseconds for 30 clients and 300 clients. The graph demonstrates a linear increase in latency as the fraction of the table scanned increases, with a steeper slope for 300 clients compared to 30 clients.](image)
Related work

- Distributed and parallel databases
  - Especially query processing and transactions
  - BigTable, Dynamo, S3, SimpleDB, SQL Server Data Services, Cassandra

- Distributed filesystems
  - Ceph, Boxwood, Sinfonia

- Distributed (P2P) hash tables
  - Chord, Pastry, ...

- Database replication
  - Master–slave, epidemic/gossip, synchronous...
Conclusions and ongoing work

- PNUTS is an interesting research product
  - Research: consistency, performance, fault tolerance, rich functionality
  - Product: make it work, keep it (relatively) simple, learn from experience and real applications

- Ongoing work
  - Indexes and materialized views
  - Bundled updates
  - Batch query processing