中国科学院
CHINESE ACADEMY OF SCIENCES

SciSpace

Towards a Serverless-Style Cloud Service for Scientific Data Lifecycle Management

(Part of the project SCIBDS)

Huajin Wang, Zhihong Shen, Yuepeng Li
wanghj@cnic.cn

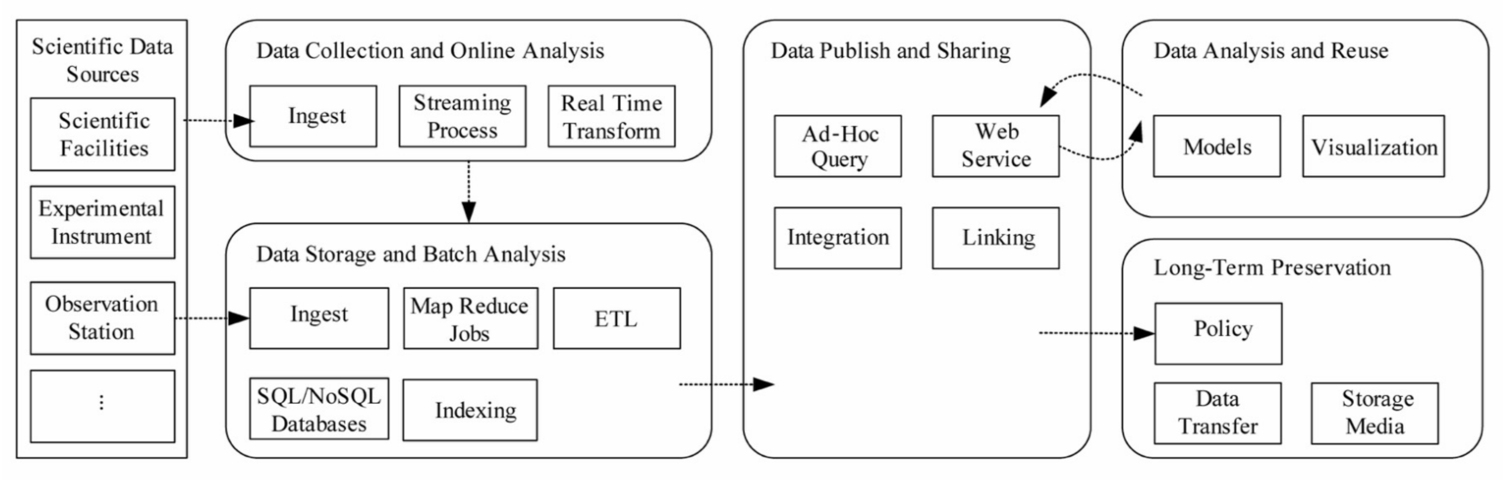
Nov. 30, 2018
BigSDM 2018, Beijing

中国科学院
计算机网络信息中心
Computer Network Information Center,
Chinese Academy of Sciences

Outline

- Data Lifecycle Management
  - What is
  - Traditional Approach
  - Serverless Approach
- SciSpace: Our Serverless Solution
  - Space: The Unified Serverless UI
  - Packone (Part I): Make Engine Elastic
  - Packone (Part II): Make URI Highly Available
- Conclusion

What is The Data Lifecycle Management



Scientific Data Sources

Scientific Facilities

Experimental Instrument

Observation Station

⋮

Data Collection and Online Analysis

Ingest

Streaming Process

Real Time Transform

Data Storage and Batch Analysis

Ingest

Map Reduce Jobs

ETL

SQL/NoSQL Databases

Indexing

Data Publish and Sharing

Ad-Hoc Query

Web Service

Integration

Linking

Data Analysis and Reuse

Models

Visualization

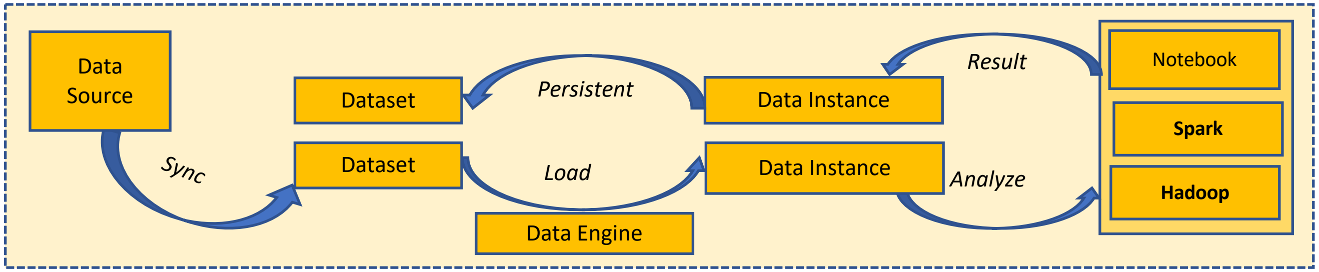
Long-Term Preservation

Policy

Data Transfer

Storage Media

Rethink:



Data Source

Sync

Dataset

Dataset

Persistent

Load

Data Engine

Data Instance

Data Instance

Result

Analyze

Notebook

Spark

Hadoop

中国科学院
计算机网络信息中心
Computer Network Information Center,
Chinese Academy of Sciences

Data Lifecycle Management: The Traditional Approach

- Data sources and scientists are scattered in different locations/clouds
  - data analysis always need to access remote data stores
  - data trans between Heterogenous Infrastructure are complicated
- No unified data lifecycle management environment
  - Scientists need to using different level UI to access data or computing powers
    - SSH/FTP/HDFS/S3/
  - Dataset or analysis results can hardly be referenceable by peers

Data Lifecycle Management: The Serverless Approach

What

- A unified data lifecycle management environment
  - Data Sync/Load/Share
  - Data Query/Analysis
- Server location is insignificant
- Server login is not required

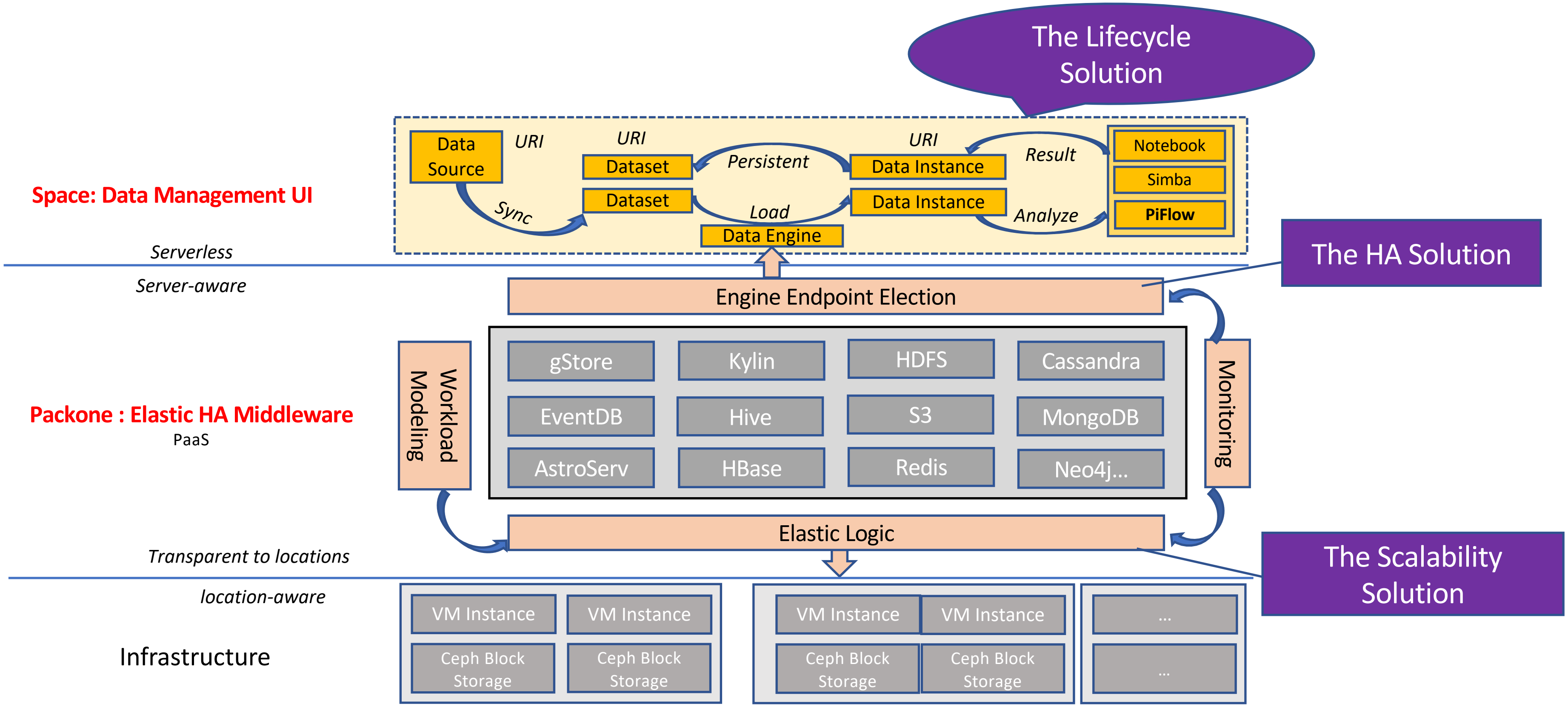
Advantages

- Liberate Scientists

Challenges

- The lifecycle challenge
  - How to takeover data lifecycle management
    - without direct interactions with underlying servers
- The scalability challenge
  - How to automatically scale computing infrastructure
    - to meet the performance constraints
- The HA challenge
  - How to tolerant server failures
    - to keep service highly available

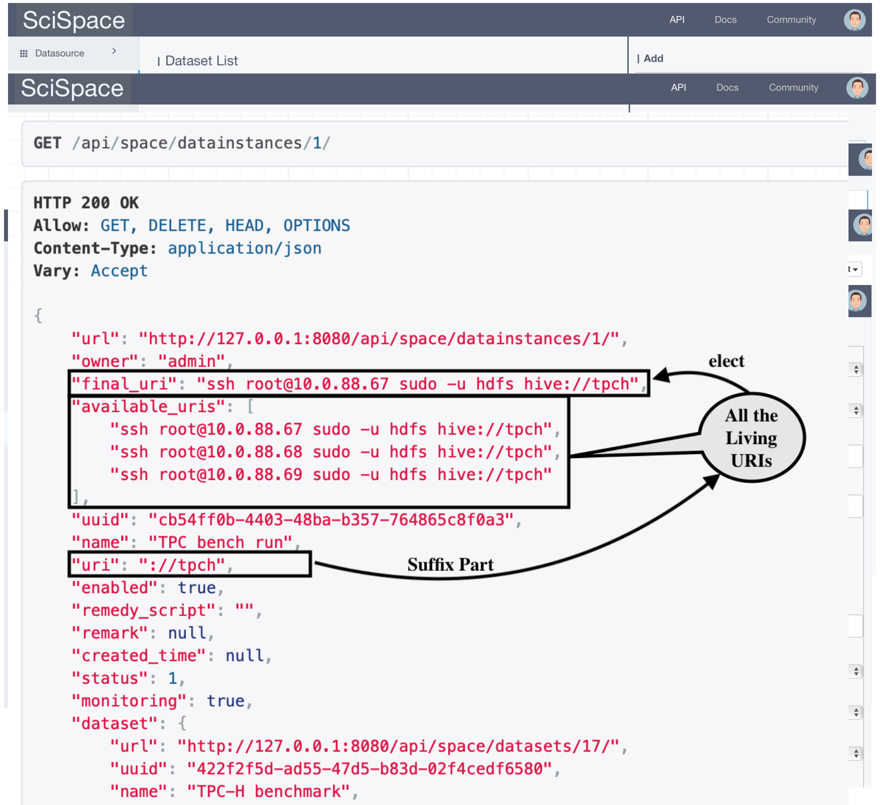
SciSpace: Our Serverless Data Lifecycle Management Service



中国科学院
计算机网络信息中心
Computer Network Information Center,
Chinese Academy of Sciences

Space: The Unified Serverless UI

- Easy to share dataset
- Dataset can be loaded into data instances multiple times
- Using the URI field to talk with each other.
- Monitor underlying computing resource located anywhere

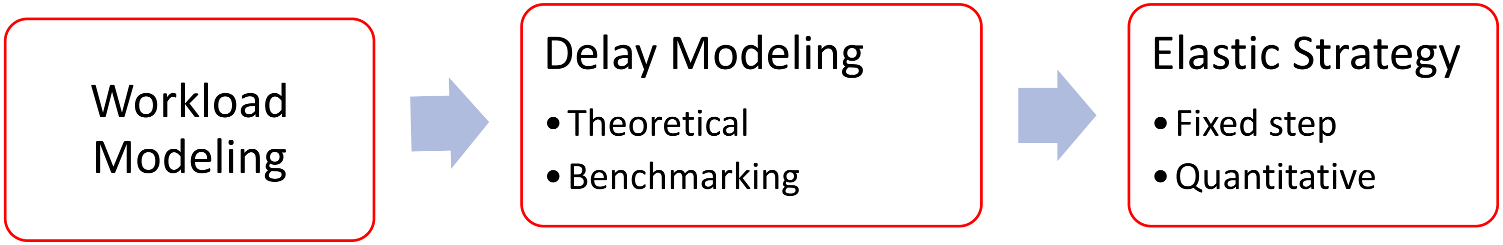


```
GET /api/space/datainstances/1/

HTTP 200 OK
Allow: GET, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "url": "http://127.0.0.1:8080/api/space/datainstances/1/",
    "owner": "admin",
    "final_uri": "ssh root@10.0.88.67 sudo -u hdfs hive://tpch",
    "available_uris": [
        "ssh root@10.0.88.67 sudo -u hdfs hive://tpch",
        "ssh root@10.0.88.68 sudo -u hdfs hive://tpch",
        "ssh root@10.0.88.69 sudo -u hdfs hive://tpch"
    ],
    "uuid": "cb54ff0b-4403-48ba-b357-764865c8f0a3",
    "name": "TPC bench run",
    "uri": "://tpch",
    "enabled": true,
    "remedy_script": "",
    "remark": null,
    "created_time": null,
    "status": 1,
    "monitoring": true,
    "dataset": {
        "url": "http://127.0.0.1:8080/api/space/datasets/17/",
        "uuid": "422f2f5d-ad55-47d5-b83d-02f4cedf6580",
        "name": "TPC-H benchmark",
```

Packone: Make Engine Elastic



Workload Modeling

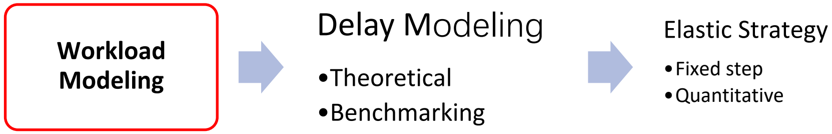
Delay Modeling

- Theoretical
- Benchmarking

Elastic Strategy

- Fixed step
- Quantitative

Packone: Make Engine Elastic

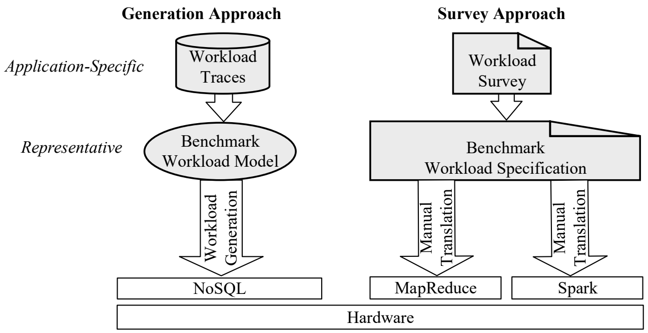


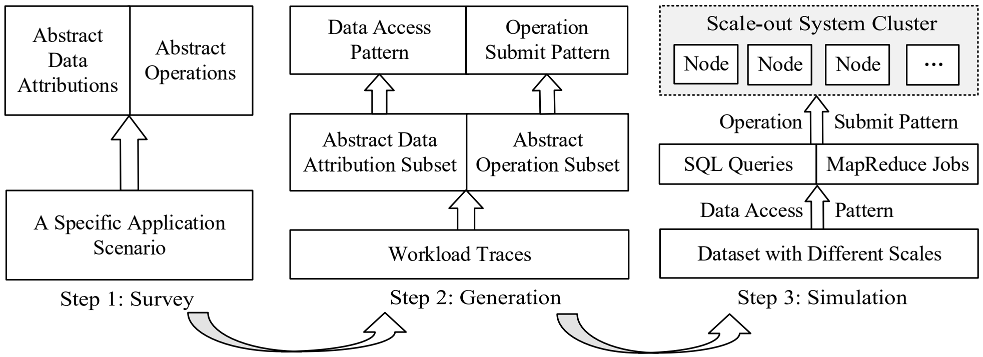
Workload Modeling

Delay Modeling
- Theoretical
- Benchmarking

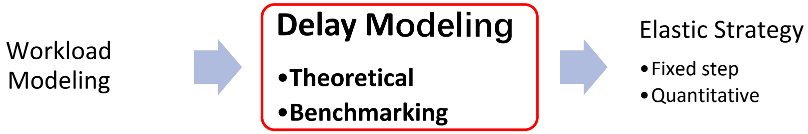
Elastic Strategy
- Fixed step
- Quantitative

- Survey Approach
  - Portable but easy to be retired
- Generation Approach
  - Up to date but engine dependent
- Our Evolutionary Approach
  - Portable
  - Up to date
  - Engine independent





Packone: Make Engine Elastic

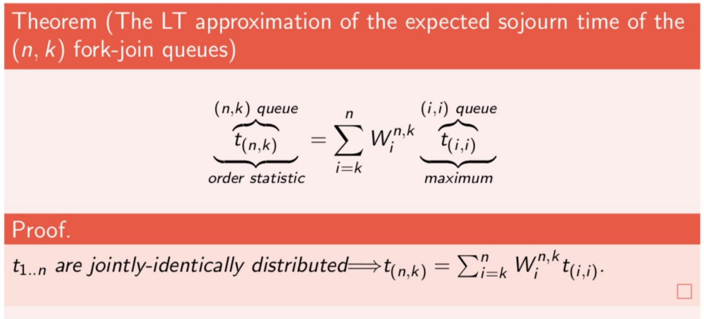


Workload Modeling

Delay Modeling
- Theoretical
- Benchmarking

Elastic Strategy
- Fixed step
- Quantitative

- **Theoretical**: LT approx. depicted how node qty. replication factor & consistency level affect the operation delay

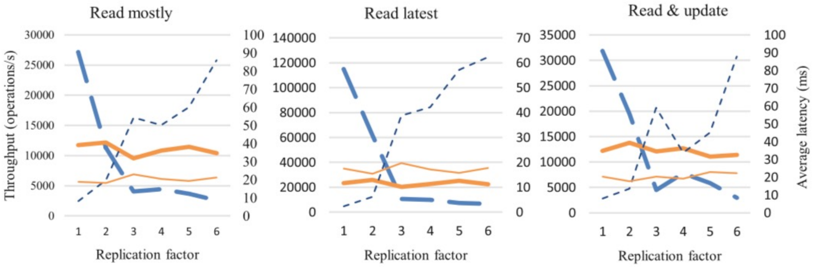


Theorem (The LT approximation of the expected sojourn time of the (n, k) fork-join queues)

$$\underbrace{\overbrace{t_{(n,k)}}^{(n,k)\text{ queue}}}_{\text{order statistic}} = \sum_{i=k}^{n} W_i^{n,k} \underbrace{\overbrace{t_{(i,i)}}^{(i,i)\text{ queue}}}_{\text{maximum}}$$

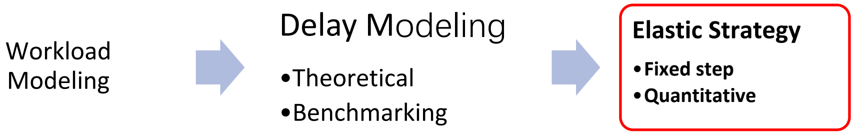
Proof.

$t_{1..n}$ are jointly-identically distributed $\Longrightarrow t_{(n,k)} = \sum_{i=k}^{n} W_i^{n,k} t_{(i,i)}$. □



- **Benchmarking** gives the real delay of the target engine
- **Real-life delay ≈ {Benchmarking@Workload Model} × Theoretical**

Packone: Make Engine Elastic



Workload Modeling

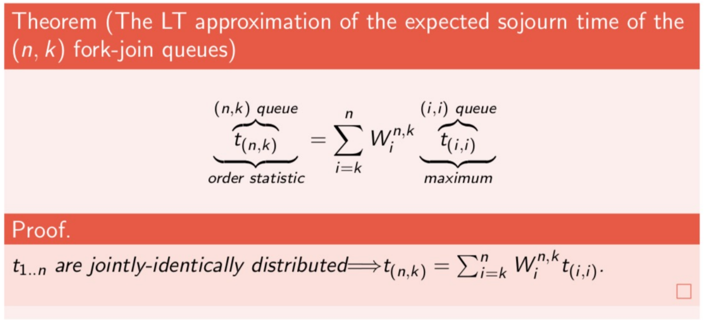
Delay Modeling

- Theoretical
- Benchmarking

Elastic Strategy

- **Fixed step**
- **Quantitative**

- Elastic Strategies to keep a satisfactory delay
  - Fixed Step: Easy to employ but low-precision
  - Quantitative: Fast-forward but hard to calculate
  - Our Mixed Strategy
    - Beginning: Fixed Step
    - Ending: Quantitative



Theorem (The LT approximation of the expected sojourn time of the (n, k) fork-join queues)

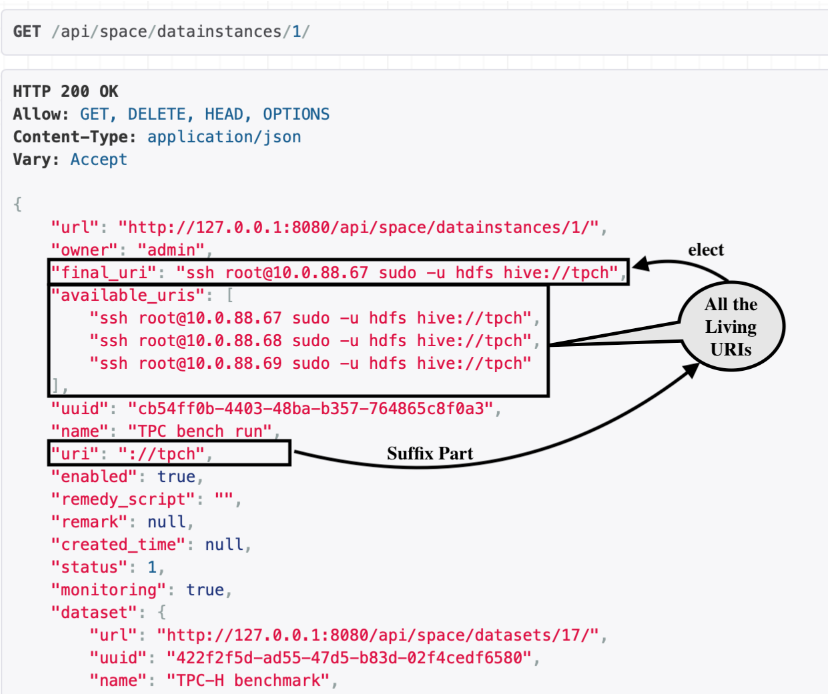
$$\underbrace{\overbrace{t_{(n,k)}}^{(n,k)\ queue}}_{order\ statistic} = \sum_{i=k}^{n} W_i^{n,k} \underbrace{\overbrace{t_{(i,i)}}^{(i,i)\ queue}}_{maximum}$$

Proof.

$t_{1..n}$ *are jointly-identically distributed* $\Longrightarrow t_{(n,k)} = \sum_{i=k}^{n} W_i^{n,k} t_{(i,i)}$. □

Packone: Make URI Highly Available

- Restful API
- Realtime available URI set
- URI Election Strategy
  - Naïve: Pick out the 1st living one
  - Load balance (TODO)
  - Round Robin (TODO)



```
GET /api/space/datainstances/1/
```

```
HTTP 200 OK
Allow: GET, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "url": "http://127.0.0.1:8080/api/space/datainstances/1/",
    "owner": "admin",
    "final_uri": "ssh root@10.0.88.67 sudo -u hdfs hive://tpch",
    "available_uris": [
        "ssh root@10.0.88.67 sudo -u hdfs hive://tpch",
        "ssh root@10.0.88.68 sudo -u hdfs hive://tpch",
        "ssh root@10.0.88.69 sudo -u hdfs hive://tpch"
    ],
    "uuid": "cb54ff0b-4403-48ba-b357-764865c8f0a3",
    "name": "TPC bench run",
    "uri": "://tpch",
    "enabled": true,
    "remedy_script": "",
    "remark": null,
    "created_time": null,
    "status": 1,
    "monitoring": true,
    "dataset": {
        "url": "http://127.0.0.1:8080/api/space/datasets/17/",
        "uuid": "422f2f5d-ad55-47d5-b83d-02f4cedf6580",
        "name": "TPC-H benchmark",
```

中国科学院
计算机网络信息中心
Computer Network Information Center,
Chinese Academy of Sciences

Conclusion

- Traditional approach
  - Scattered dataset and computing resources
  - Complicated and difficult
  - Poorly referenceable
- Serverless Approach
  - The lifecycle challenge
  - The scalability challenge
  - The HA challenge
- SciSpace: Our Serverless Data Lifecycle Management Service
  - Space: The Unified Serverless UI
  - Packone Make Engine Elastic and URI Highly Available

References

- J Li , Z Shen, and X Meng. Scientific Big Data Management: Concepts, Technologies and System.
- H Wang, J Li, H Zhang, Y Zhou. Benchmarking replication and consistency strategies in cloud serving databases: Hbase and cassandra
- H Wang, J Li, Z Shen, Y Zhou. Approximations and Bounds for (n, k) Fork-Join Queues: A Linear Transformation Approach. CCGrid 2018.
- H Wang, M Wan, R Han etc. Towards the Automatic Evolution of Workload Models in Large-scale Astronomical Data Management.

Thanks!

Q & A