On Data Placement in Distributed Systems

LADIS’14

João Paiva, Luís Rodrigues
{joao.paiva, ler}@tecnico.ulisboa.pt

Instituto Superior Técnico / INESC-ID, Lisboa, Portugal

October 23, 2014
What is Data Placement?

- Deciding how to assign data items to nodes in a distributed system in such way that they can be later retrieved.
Data Placement Affects

**Data Access Locality**
Placing correlated data together can reduce latency of operations

**Load Balancing**
By knowing the workload, data can be placed in a way to even out the load across all nodes

**Availability**
Data can be replicated depending on probability of node failure
Constraints to data placement practicality

- Lack of flexibility limits data placement improvements
- Scalability imposes limits on the flexibility of placement

Example

- Using a centralized directory is flexible, but not scalable
- Using consistent hashing is scalable, but not flexible
Constraints to data placement practicality

- Lack of flexibility limits data placement improvements
- Scalability imposes limits on the flexibility of placement

Example

- Using a centralized directory is flexible, but not scalable
- Using consistent hashing is scalable, but not flexible
Main Goal

Provide better options between

- Strong flexibility, limited scalability
- Limited flexibility, good scalability
Two Scenarios

Internet Scale
- Millions of nodes
- Short term connections
- Asymmetric, inconstant network

Datacenter Scale
- Thousands of nodes
- Stable connections
- Controlled network infrastructure
Two Scenarios: Previous state of the art

Internet Scale
  ▶ Scalable solutions with little flexibility, concerned with churn

Datacenter Scale
  ▶ Very flexible solutions, concerned with workload changes
Summary of Findings

Improvements for both scenarios:

- More flexible solution for Internet-Scale
- More scalable solution for Datacenter-Scale
Outline

Introduction

Internet Scale

Datacenter Scale

Conclusion
Data assigned to node groups
Variable replication degree
Nodes have no fixed position
Variable Replication Degree
Variable Replication Degree
Variable Replication Degree
Variable Replication Degree
Variable Replication Degree
Variable Replication Degree
Variable Replication Degree
Nodes have no fixed position
Nodes have no fixed position
Nodes have no fixed position
Internet Scale: Implementation

Rollerchain

- Gossip-based and structured overlay
- Better churn resilience than state of the art
- Decreased replication costs

Data Placement Policies

- **Avoid-Surplus**: Reducing monitoring costs
- **Resilient Load-Balancing**: Improving load balancing
- **Supersize-me**: Reducing replication costs

Read the paper to know the best policies:

Data Placement Policies

- **Avoid-Surplus**: Reducing monitoring costs
- **Resilient Load-Balancing**: Improving load balancing
- **Supersize-me**: Reducing replication costs

Read the paper to know the best policies:

Outline

Introduction

Internet Scale

Datacenter Scale

Conclusion
Datacenter scale: AutoPlacer

System where data placement is defined by combining:

- Consistent hashing for *most* items
- Precise placement for *selected* items

Locality-improving round-based algorithm for in-memory data grids


Datacenter scale: AutoPlacer

System where data placement is defined by combining:
  ▶ Consistent hashing for *most* items
  ▶ Precise placement for *selected* items

Locality-improving round-based algorithm for in-memory data grids

"AutoPlacer: scalable self-tuning data placement in distributed key-value stores”,
J. Paiva, P. Ruivo, P. Romano and L. Rodrigues, International Conference on
Autonomic Computing (*USENIX ICAC*), June 2013. (*Best paper finalist*)

"AutoPlacer: scalable self-tuning data placement in distributed key-value stores”,
and Adaptive Systems (*ACM TAAS*)
Online, round-based approach:

1. Statistics: Monitor data access to collect hotspots
2. Optimization: Decide placement for hotspots
3. Lookup: Encode / broadcast data placement
4. Move data
Algorithm overview

Online, round-based approach:

1. **Statistics**: Monitor data access to collect hotspots
2. Optimization: Decide placement for hotspots
3. Lookup: Encode / broadcast data placement
4. Move data
Key concept: Top-K stream analysis algorithm

- Lightweight
- Sub-linear space usage
- Inaccurate result... But with bounded error
Key concept: Top-K stream analysis algorithm

- Lightweight
- Sub-linear space usage
- Inaccurate result... But with bounded error
Statistics: Data access monitoring

**Key concept:** Top-K stream analysis algorithm

- Lightweight
- Sub-linear space usage
- Inaccurate result... But with bounded error
Algorithm overview

Online, round-based approach:

1. Statistics: Monitor data access to collect hotspots
2. **Optimization**: Decide placement for hotspots
3. Lookup: Encode / broadcast data placement
4. Move data
Optimization

Integer Linear Programming problem formulation:

\[
\min \sum_{j \in \mathcal{N}} \sum_{i \in \mathcal{O}} X_{ij} (cr^r r_{ij} + cr^w w_{ij}) + X_{ij} (cl^r r_{ij} + cl^w w_{ij}) \quad (1)
\]

subject to:

\[
\forall i \in \mathcal{O} : \sum_{j \in \mathcal{N}} X_{ij} = d \land \forall j \in \mathcal{N} : \sum_{i \in \mathcal{O}} X_{ij} \leq S_j
\]

Inaccurate input:

- Does not provide optimal placement
- Upper-bound on error
Accelerating optimization

1. ILP Relaxed to Linear Programming problem
2. Distributed optimization

LP relaxation

- Allow data item ownership to be in \([0 - 1]\) interval

Distributed Optimization

- Partition by the \(N\) nodes
- Each node optimizes hotspots mapped to it by CH
- Strengthen capacity constraint
Algorithm overview

Online, round-based approach:

1. Statistics: Monitor data access to collect hotspots
2. Optimization: Decide placement for hotspots
3. **Lookup:** Encode / broadcast data placement
4. Move data
Lookup: Encoding placement

Probabilistic Associative Array (PAA)

- Associative array interface (keys → values)
- Probabilistic and space-efficient
- Trade-off space usage for accuracy
Probabilistic Associative Array: Usage

Building

1. Build PAA from hotspot mappings
2. Broadcast PAA

Looking up objects

- If item is hotspot, return PAA mapping
- Otherwise, default to Consistent Hashing
Probabilistic Associative Array: Usage

Building

1. Build PAA from hotspot mappings
2. Broadcast PAA

Looking up objects

- If item is hotspot, return PAA mapping
- Otherwise, default to Consistent Hashing
PAA: Building blocks

- **Bloom Filter**
  Space-efficient membership test (is item in PAA?)

- **Decision tree classifier**
  Space-efficient mapping (where is hotspot mapped to?)
PAA: Building blocks

- **Bloom Filter**
  Space-efficient membership test (is item in PAA?)

- **Decision tree classifier**
  Space-efficient mapping (where is hotspot mapped to?)
PAA: Properties

Bloom Filter:
- **No False Negatives**: never return ⊥ for items in PAA.
- **False Positives**: match items that it was not supposed to.

Decision tree classifier:
- **Inaccurate values** (bounded error).
- **Deterministic response**: deterministic (item→node) mapping.
PAA: Properties

Bloom Filter:

- **No False Negatives**: never return ⊥ for items in PAA.
- **False Positives**: match items that it was not supposed to.

Decision tree classifier:

- **Inaccurate values** (bounded error).
- **Deterministic response**: deterministic (item→node) mapping.
PAA: Properties

Bloom Filter:

- **No False Negatives**: never return \( \perp \) for items in PAA.
- **False Positives**: match items that it was not supposed to.

Decision tree classifier:

- **Inaccurate** values (bounded error).
- **Deterministic response**: deterministic (item\(\rightarrow\)node) mapping.
Outline

Introduction

Internet Scale

Datacenter Scale
  Autoplacer
  Evaluation

Conclusion
  Conclusion
Evaluation: Throughput

Transactions per second (TX/s) vs Time (minutes) for different locality percentages: 100%, 90%, 50%, 0%, and baseline.
Evaluation: Optimization

The graph shows the objective function relative to unoptimized (%) over rounds for different configurations. The configurations include AutoPlacer with 50% locality, Centralized with 50% locality, AutoPlacer with 90% locality, Centralized with 90% locality, AutoPlacer with 100% locality, and Centralized with 100% locality.
Outline

Introduction

Internet Scale

Datacenter Scale

Conclusion
Conclusions

Internet Scale

- More flexible overlay for data placement
- Policies to improve metrics using added flexibility

Datacenter Scale

- Scalable mechanism for data placement
- Algorithm to improve locality through hotspot placement
Thank you

joao.paiva@tecnico.ulisboa.pt
web.tecnico.ulisboa.pt/joao.paiva