coOrd

A Distributed Computing Platform for Large-scale Data & Computations

Woohyun Kim

The creator of open source “Coord”

(http://www.coordguru.com)

2009-10-27

Contents

Introduction to Coord

- Background
- A Glance at Coord
- Thrust Coord into Mainstream
- Coord Internals

HOWTO Coord

- HOWTO Coord
- HOWTO Coord MapReduce

Web Crawling

- Web Crawling Practice - NCDC Dataset

Relational Algebra Operations

- A Great Debate between MapReduce & RDBMS
- Parallelize RDBMS
- Parallel Join Practice: MovieLens Dataset
- Further Study in Parallel Join using MapReduce
- Improvements in Parallel Join

Large-scale Graph Search

- Graph Search on Coord

Conclusion

- Coord Footprint
- Future Works

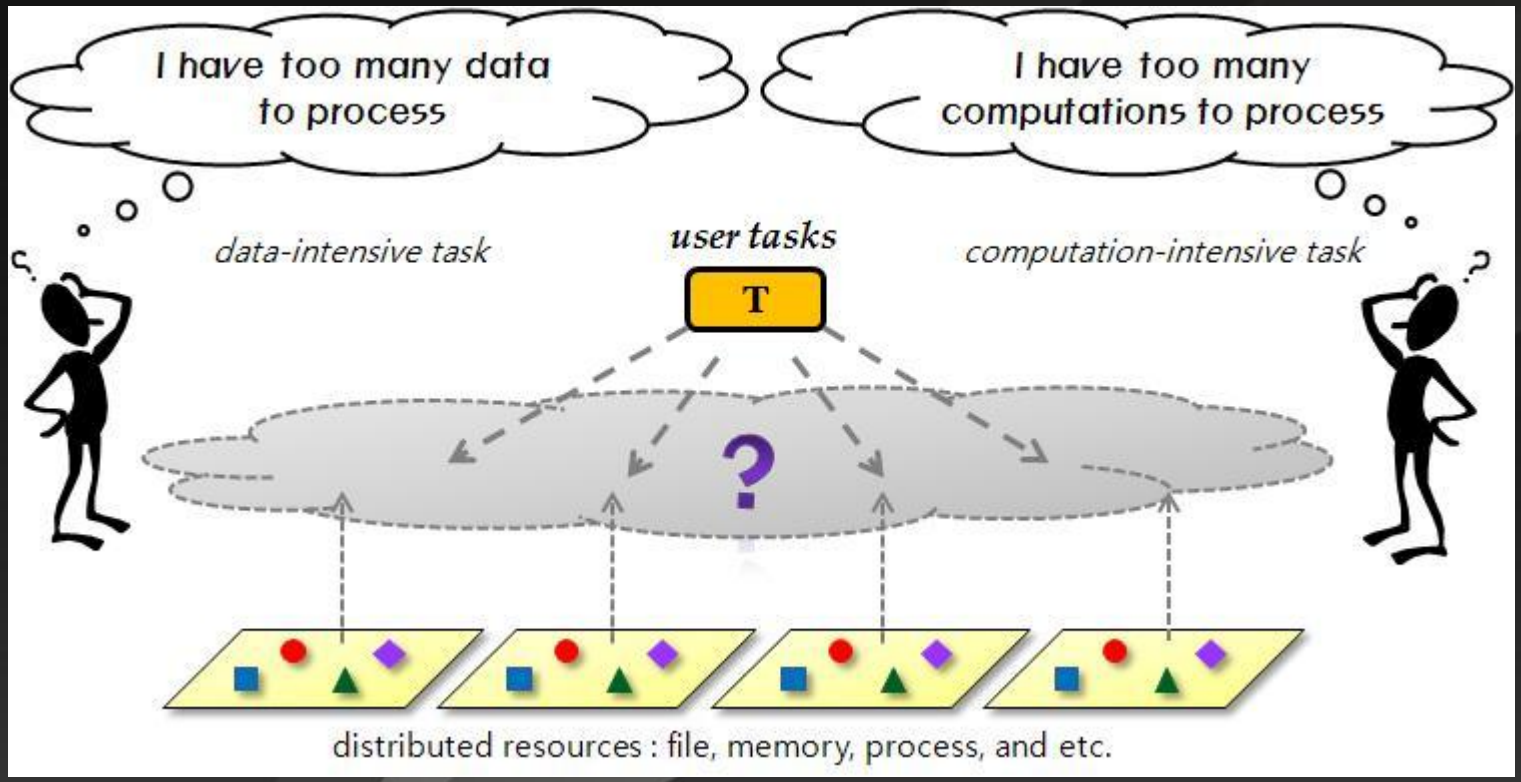
coord

Introduction to coOrd

Background

Nagging Problems in Distributed Environments

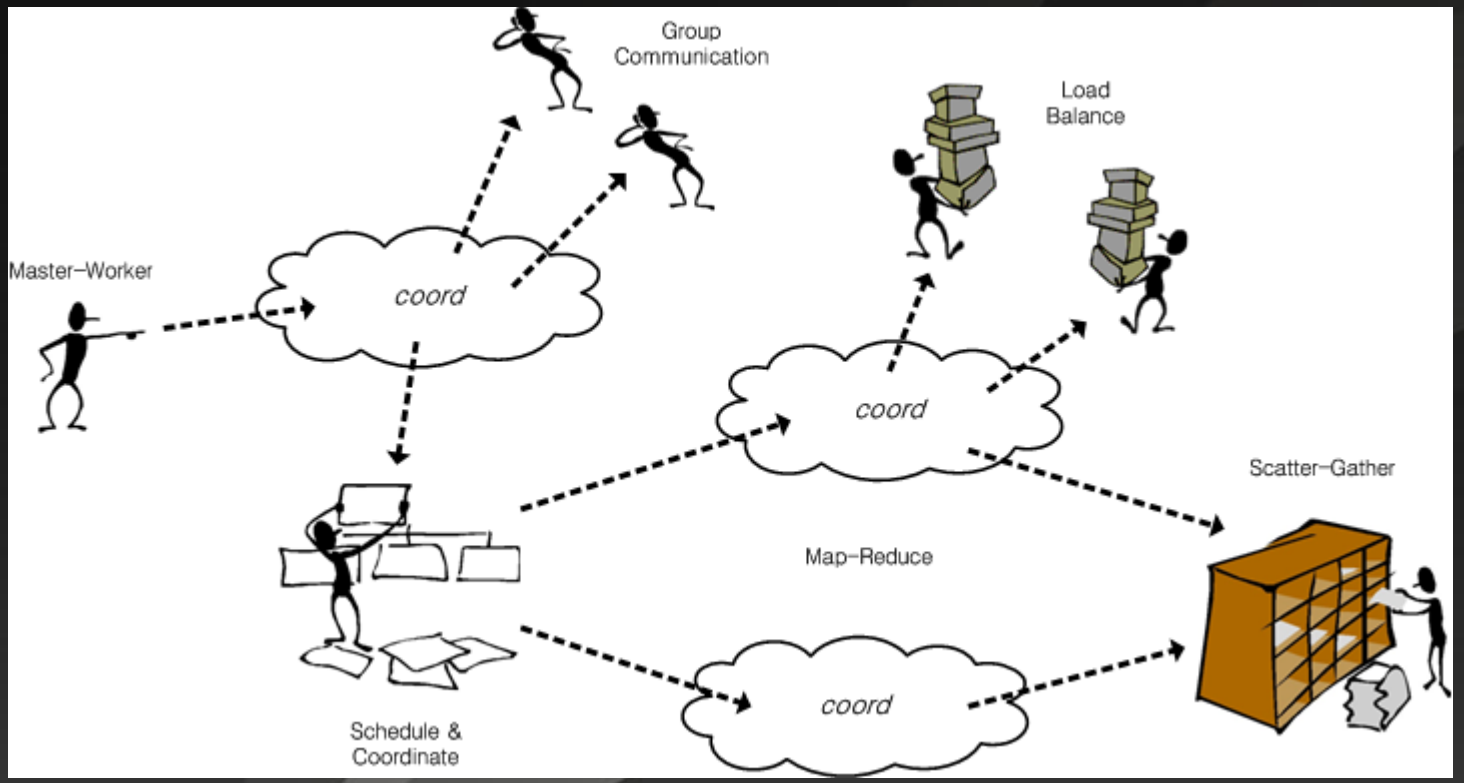
- Data-intensive jobs
- Computation-intensive jobs



A Glance at Coord

Coord

- A Coordinating System for Large-scale Distributed Resources
- A Distributed Computing Platform for Large-scale Data Analyses

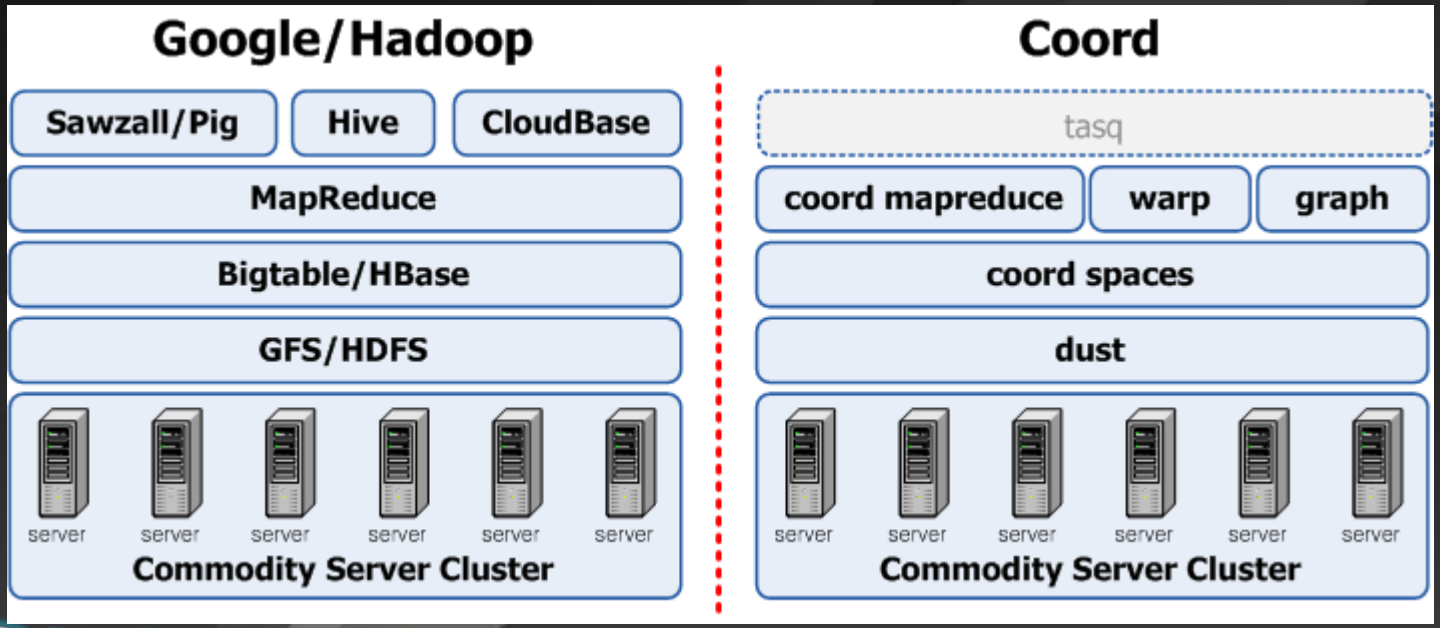


coord

Thrust Coord into mainstream

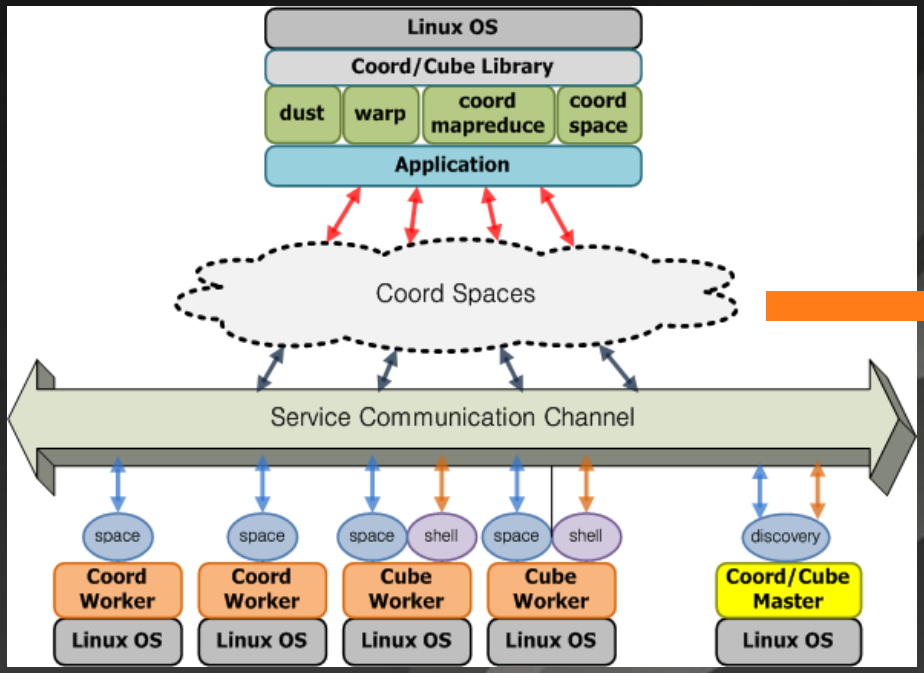
Architectural Comparison

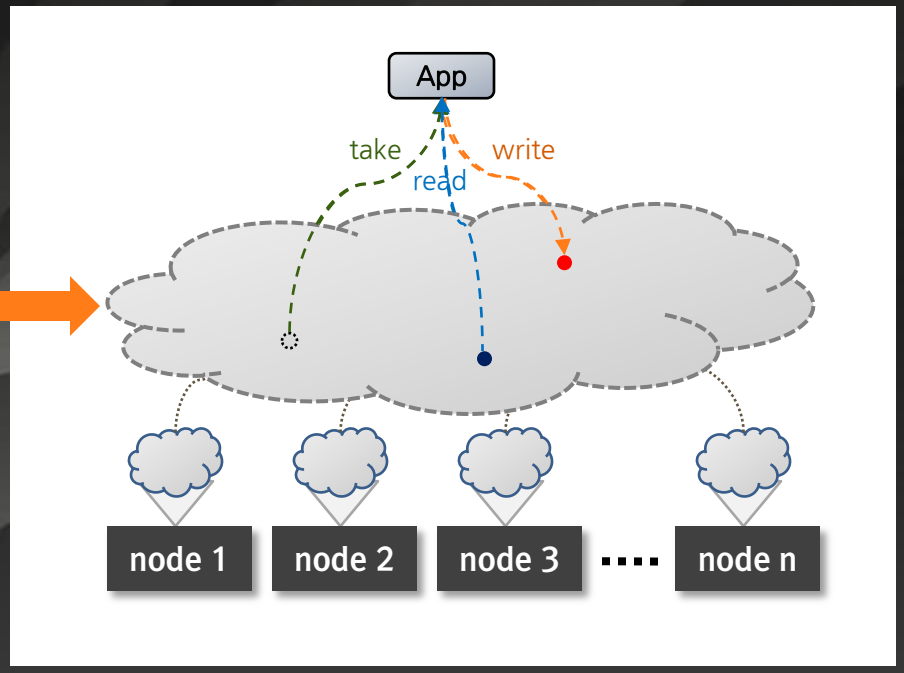
- dust: a distributed file system based on DHT
- coord spaces: a resource sharable store system based on SBA
- coord mapreduce: a simplified large-scale data processing framework
- warp: a scalable remote/parallel execution system
- graph: a large-scale distributed graph search system



Coord Internals

- A space-based architecture built on distributed hash tables
  - SBA(Space-based Architecture)
    - processes communicate with others thru. only spaces
  - DHT(Distributed Hash Tables)
    - data identified by hash functions are placed on numerically near nodes
- A computing platform to project a single address space on distributed memories
  - As if users worked in a single computing environment





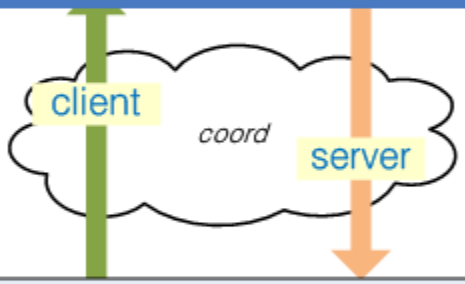
coord

HOWTO coOrd

HOWTO Coord

client

```
Coord myspace("myspace");
myspace.autoclean(false);

pair<string,string> msg = make_pair("server", "I'm a client");
myspace.write(msg);

msg = make_pair("client", "");
myspace.take(msg);

cout << "[client]" << msg.second << endl;
```

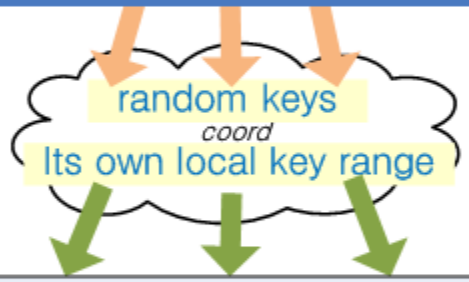



```
Coord myspace("myspace");
myspace.autoclean(false);

pair<string,string> msg = make_pair("server", "");
myspace.take(msg);

cout << "[server]" << msg.second << endl;

msg = make_pair("client", "I'm a server");
myspace.write(msg);
```

server

scatter

```
Coord scatter("myspaces");
scatter.autoclean(false);

pair<string,string> keyval;

for(int i=0;i<1000;i++) {
   keyval = get_kv(keyval); // generate random keys

   scatter.write(keyval);
}
```

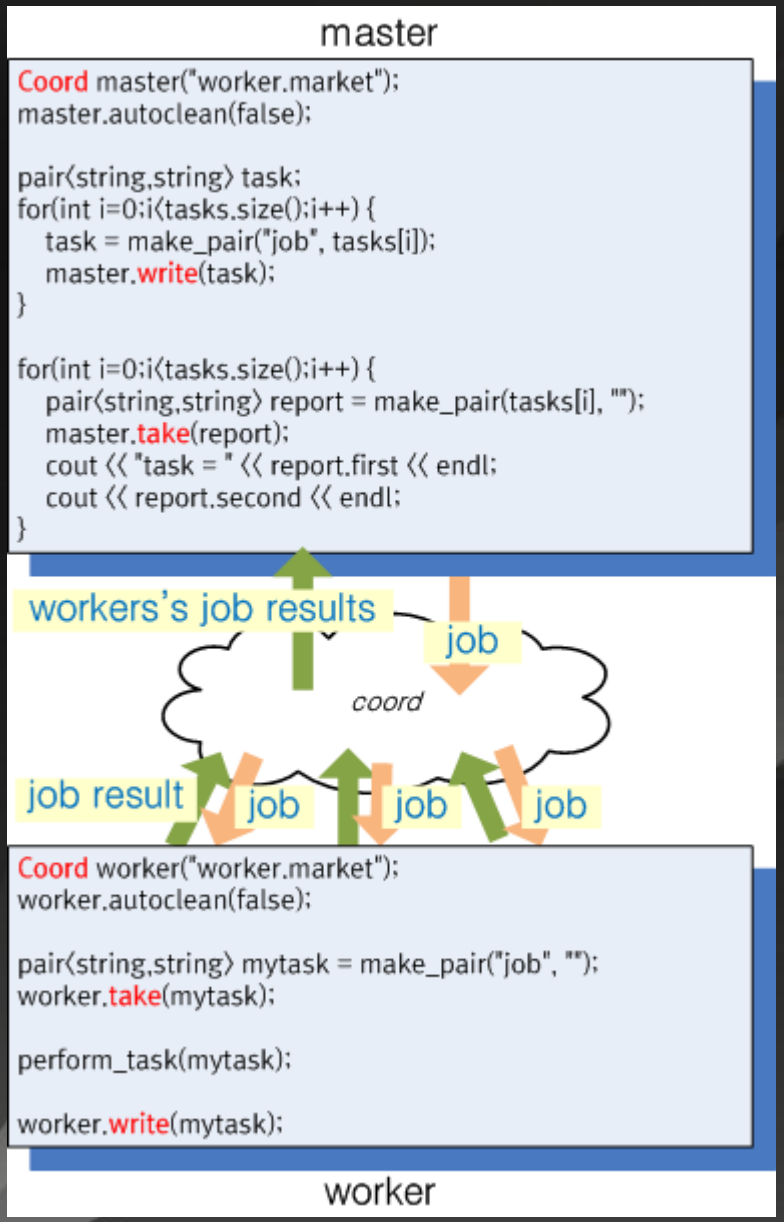



```
Coord gather("myspaces");
gather.autoclean(false);

for(int i=0;gather.size()>0;i++) {
   pair<string,string> keyval = make_pair("", "");

   gather.take(keyval);

   cout << keyval.first << "=" << keyval.second << endl;
}
```

gather

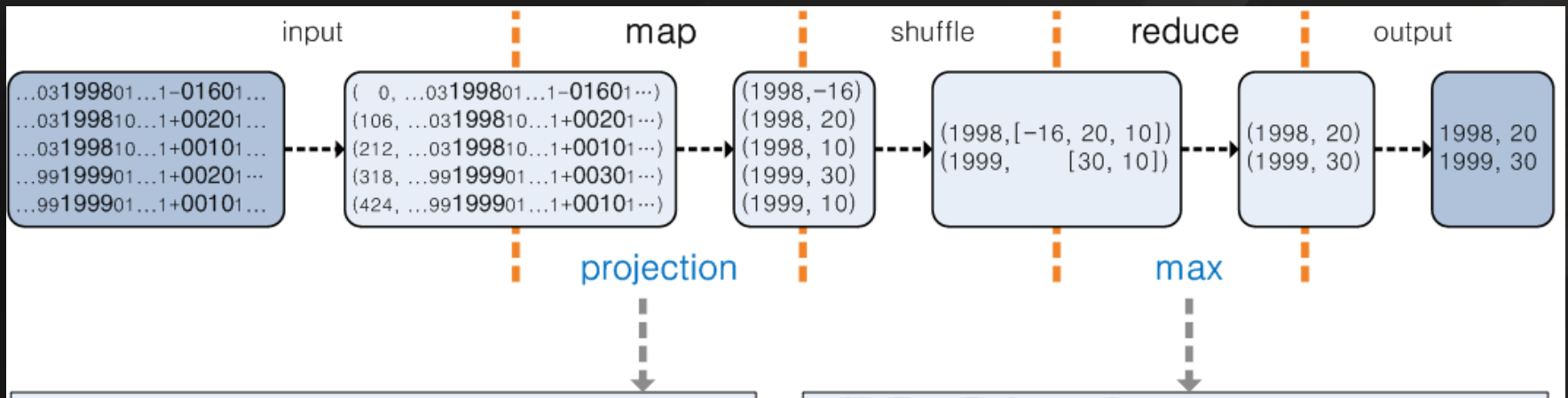
coord

HOWTO Coord (cont'd)



coOrd

HOWTO Coord MapReduce




```
int YearTempMap::map(const char *key, const char *value)
{
        string missing = "9999";
        string raw_data = value;
        if(raw_data.length()<93) return 1; // skip wrong recordings

        string year  = raw_data.substr(15,4);

        string temperature;
        if(raw_data[87]=='+') temperature = raw_data.substr(88,4);
        else temperature = raw_data.substr(87,5);

        string quality = raw_data.substr(92,1);

        if(temperature!=missing&&
             quality.find_first_of("01459")!=string::npos)
                        emit(year.c_str(), temperature.c_str());

        return 1;
}
```

```
int MaxTempReduce::reduce(const char *key, vector<const
char *> &values)
{
        int max_temp = INT_MIN;

        for(int i=0;i<values.size();i++) {
                max_temp = max(max_temp, atoi(values[i]));
        }

        char final_max_temp[50];
        sprintf(final_max_temp, "%d", max_temp);

        emit(key, final_max_temp);

        return 1;
}
```

coord

Demo

Web Crawling

Web Crawling Practice: NCDC Dataset

NCDC Dataset

- NCDC(National Climatic Data Center) is the world's largest active archive of weather data.
- This data is stored using a line-oriented ASCII format, in which each line is a record.
- A sample dataset format as follows:

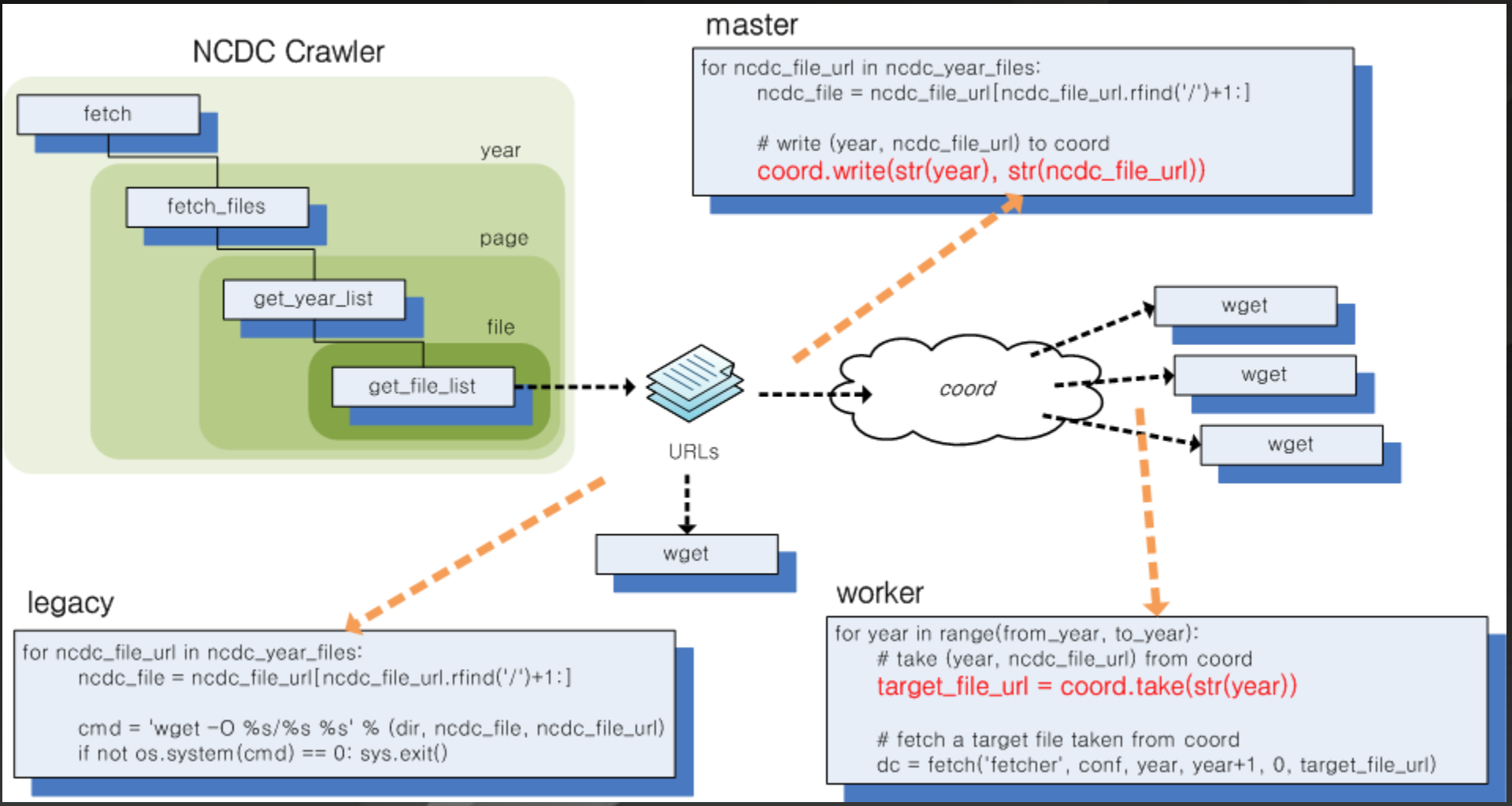
0093726813999991999010107364+43630-116630FM-15+0740KEUL V0203301N002612200019N0160931N1+00301-
00101999999ADDAG11999GF10099199999999999999999MA1102101999999MW1001REMMET008NEW
AO1;EQDQ01+000000PRSWM2
0093726813999991999010108364+43630-116630FM-15+0740KEUL V0203001N003112200019N0160931N1+00201-
00201999999ADDAG11999GF10099199999999999999999MA1102241999999MW1001REMMET008NEW
AO1;EQDQ01+000000PRSWM2
0093726813999991999010109364+43630-116630FM-15+0740KEUL V0209999C000012200019N0160931N1+00101-
00201999999ADDAG11999GF10099199999999999999999MA1102301999999MW1001REMMET008NEW
AO1;EQDQ01+000000PRSWM2
0093726813999991999010110364+43630-116630FM-15+0740KEUL V0203201N002112200019N0160931N1+00101-
00201999999ADDAG11999GF10099199999999999999999MA1102401999999MW1001REMMET008NEW
AO1;EQDQ01+000000PRSWM2

Observation Date

Air Temperature

Web Crawling Practice: NCDC Dataset (cont'd)

How to Parallelize NCDC Crawlers



Demo

Relational Algebra Operations

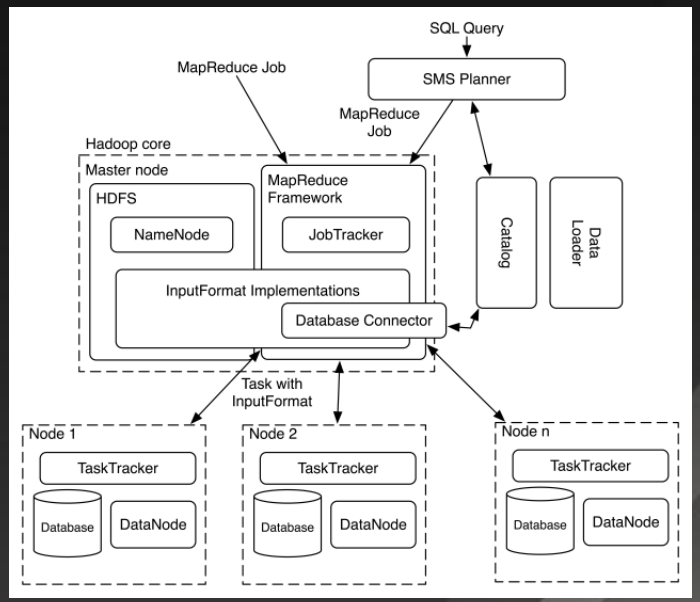
A Great Debate between MapReduce & RDBMS

RDBMS compared to MapReduce

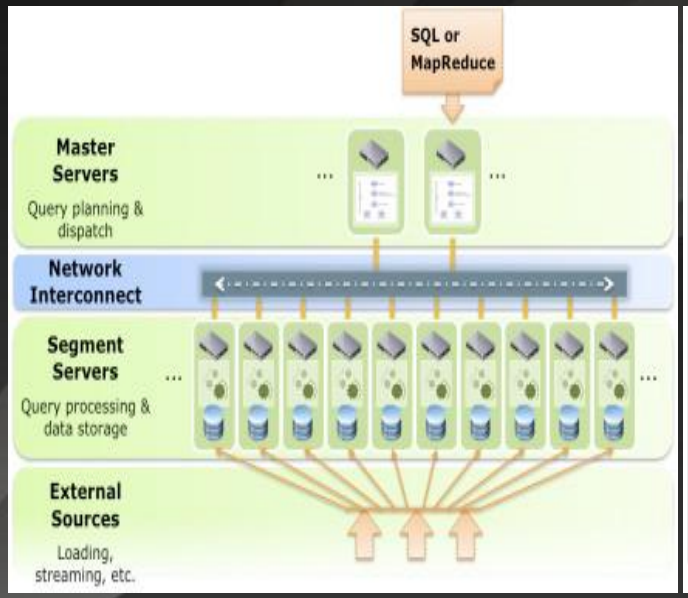
	Traditional RDBMS	MapReduce
Data size	Gigabytes	Petabytes
Access	Interactive and batch	Batch
Updates	Read and write many times	Write once, read many times
Structure	Static schema	Dynamic schema
Integrity	High	Low
Scaling	Nonlinear	Linear

from Hadoop - The Definitive Guide

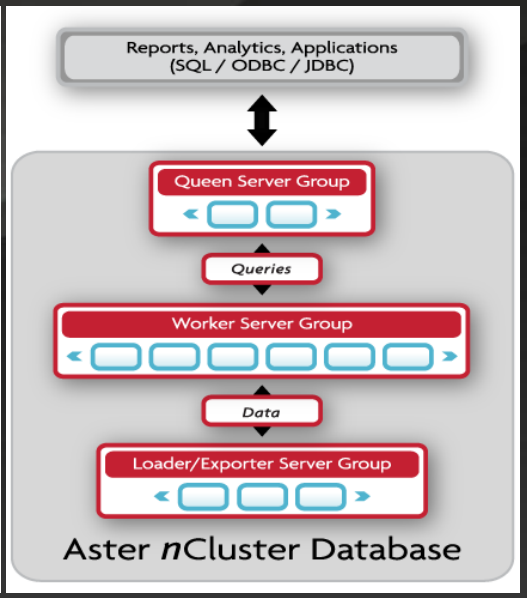
Hybrid Systems of MapReduce & RDBMS



HadoopDB



Greenplum

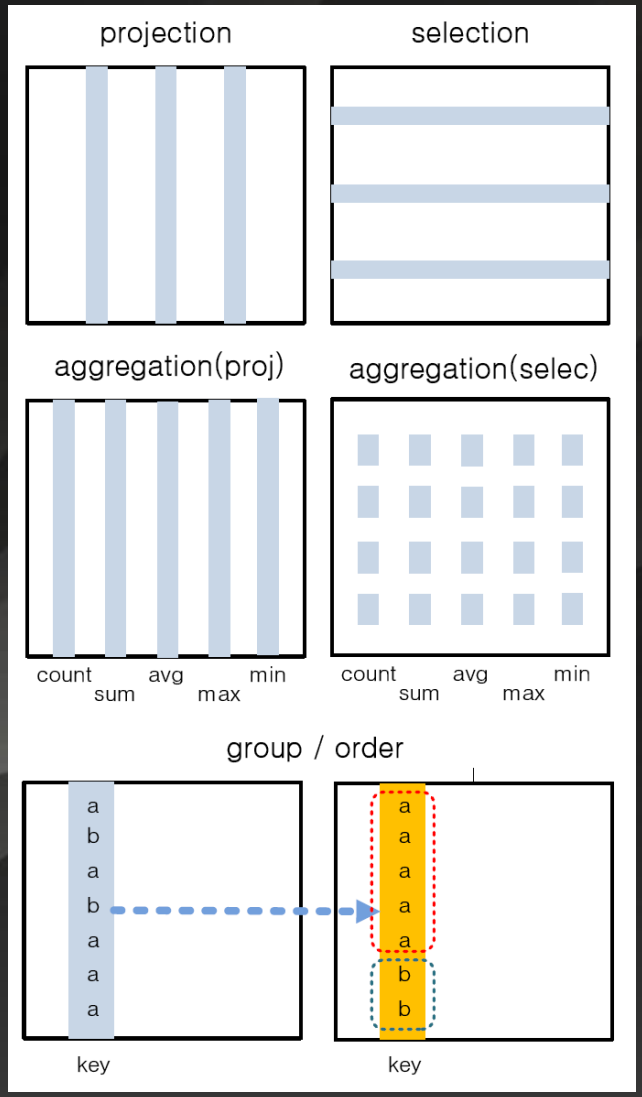


Aster Data

Parallelize RDBMS

Relational Algebra Operations

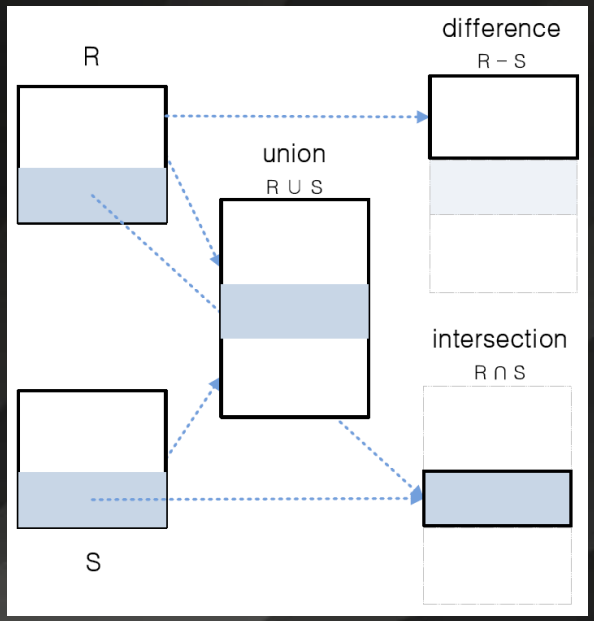
- Unary Operations
  - Projection
  - Selection
  - Aggregation
    - Count
    - Sum
    - Avg
    - Min
    - Max
  - Group & Order

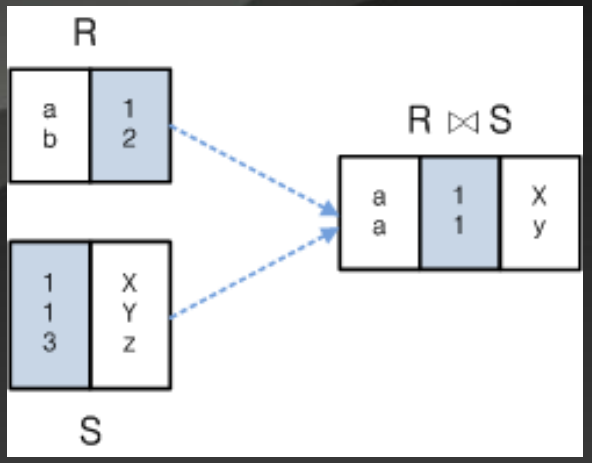


Parallelize RDBMS (cont'd)

Relational Algebra Operations

- Binary Operations
  - Set Operations
    - Union
    - Intersection
    - Difference
  - Cartesian Product
  - Join: selection + cartesian product
    - Theta join
    - Equijoin
    - Outerjoin
    - Natural join
    - Semijoin
    - Division





Parallel Join Practice: MovieLens Dataset

MovieLens Dataset

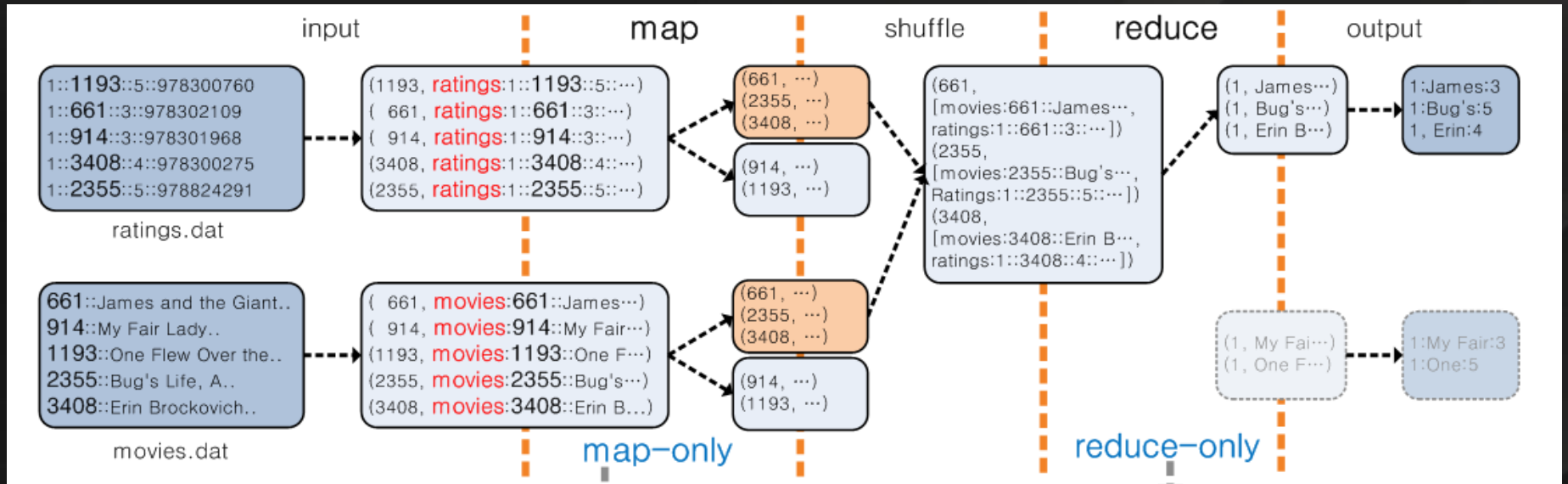
- MovieLens data sets are provided by the GroupLens Research Project at the University of Minnesota
- This data set contains 10,000,054 ratings and 95,580 tags applied to 10,681 movies by 71,567 users of the online movie recommender service MovieLens
- The data are contained in three files:
  - movies.dat
  - ratings.dat
  - tags.dat
  - users.dat
- GroupLens Research operates a movie recommender based on collaborative filtering, MovieLens, which is the source of these data

Parallel Join Practice: MovieLens Dataset (cont'd)

MovieLens Dataset Format

- Movies.dat
  - MovieID::Title::Genres
    - 1::Toy Story (1995)::Adventure|Animation|Children|Comedy|Fantasy
    - 2::Jumanji (1995)::Adventure|Children|Fantasy
- Ratings.dat
  - UserID::MovieID::Rating::Timestamp
    - 1::616::5::838984941
    - 2::110::5::868245777
- Tags.dat
  - UserID::MovieID::Tag::Timestamp
    - 15::4973::excellent!::1215184630
    - 20::1747::politics::1188263867
- Users.dat
  - UserID::Sex:Age
    - 1::F::1::10::48067
    - 2::M::56::16::70072

Parallel Join Practice: MovieLens Dataset (cont'd)




```
int RatingsMap::map(const char *key, const char *value)
{
        string tag = "ratings";
        string tagged_value = tag + delim + value;

        emit(key, tagged_value.c_str());

        return 1;
}
```

```
int MoviesMap::map(const char *key, const char *value)
{
        string tag = "movies";
        string tagged_value = tag + delim + value;

        emit(key, tagged_value.c_str());

        return 1;
}
```

```
int RatingsMoviesJoinReduce::reduce(const char *key, vector< vector<string> > &tokenized_values)
{
        … (omitted)

        map<string,int> movies;
        for(int i=0;i<tokenized_values.size();i++) {
                string tag = tokenized_values[i][tag_pos];
                if(tag==movies_tag) movies[tokenized_values[i][movies_mid_pos]] = i;
        }

        stringstream joined_value;
        for(int i=0;i<tokenized_values.size();i++) {
                string tag = tokenized_values[i][tag_pos];
                if(tag==ratings_tag) {
                        joined_value <<
tokenized_values[movies[tokenized_values[i][ratings_mid_pos]]][movies_mname_pos] << delim;
                        joined_value << tokenized_values[i][ratings_score_pos] << delim;

                        emit(tokenized_values[i][ratings_uid_pos].c_str(), joined_value.str().c_str());
                        joined_value.str("");
                }
        }

        return 1;
}
```

Demo

Further Study in Parallel Join using MapReduce

Problems

- Need to sort
- Move the partitioned data across the network
  - Due to shuffling, must send the whole data
- Skewed by popular keys
  - All records for a particular key are sent to the same reducer
- Overhead by tagging

Alternatives

- Map-side Join
  - Mapper-only job to avoid sort and to reduce data movement across the network
- Semi-Join
  - Shrink data size through semi-join(by preprocessing)

Improvements in Parallel Join

Map-Side Join

- Replicate a relatively smaller input source to the cluster
  - what if use Coord?
- Join - a relatively larger input source with the replicated dataset
  - Mapper: do Mapper-side Join

Semi-Join

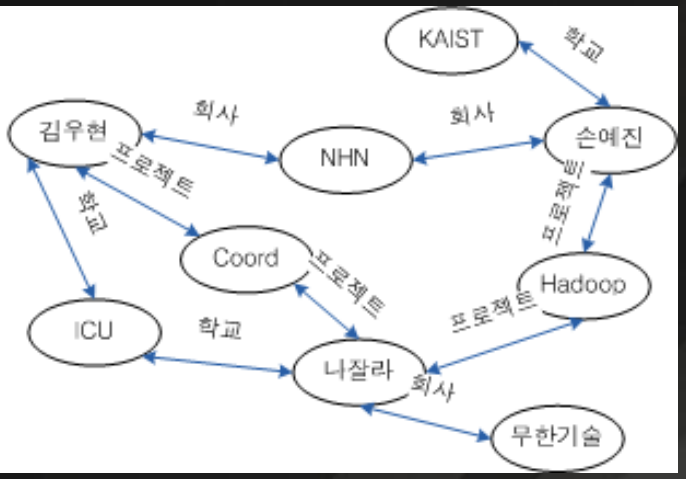
- Extract - unique IDs in the other larger input source referenced in a larger input source
  - Mapper: extract Movie IDs from Ratings records
  - Reducer: accumulate all unique Movie IDs
- Filter - the other larger input source with the referenced unique IDs
  - Mapper: filter the referenced Movie IDs from full Movie dataset
- Join - the larger input source with the filtered dataset
  - Mapper: do Mapper-side Join
    - Ratings records & the filtered movie IDs dataset

Large-scale Graph Search

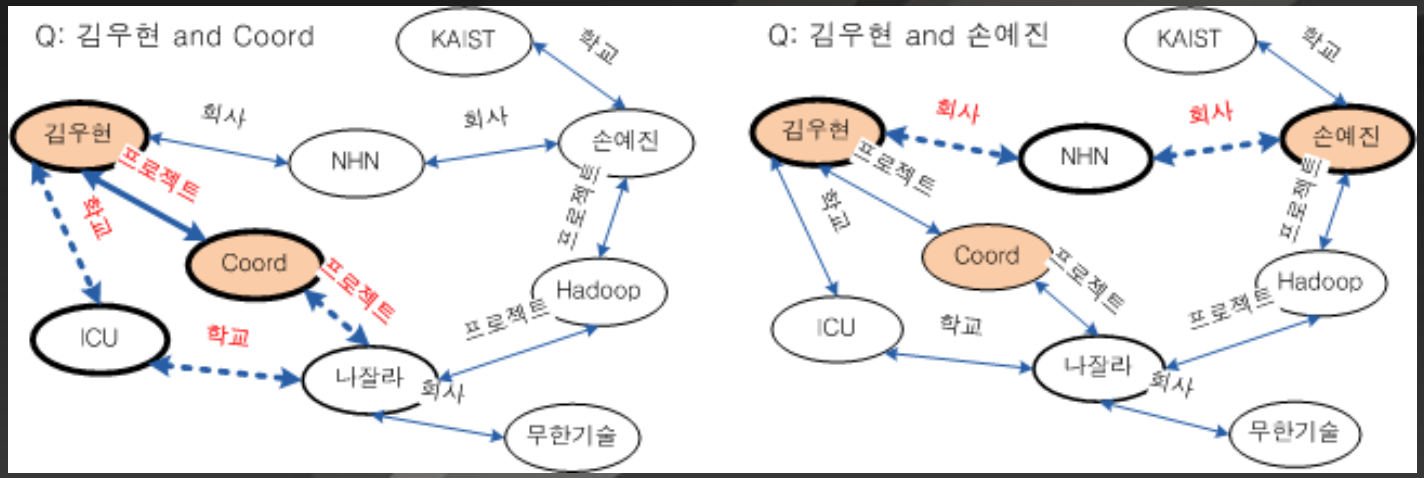
Graph Search on Coord

Sample Graph Dataset & Its Graph

(김우현, 회사, NHN)
(김우현, 프로젝트, Coord)
(김우현, 학교, ICU)
(손예진, 회사, NHN)
(손예진, 프로젝트, Hadoop)
(손예진, 학교, KAIST)
(나잘라, 회사, 무한기술)
(나잘라, 프로젝트, Coord)
(나잘라, 프로젝트, Hadoop)
(나잘라, 학교, ICU)



Graph Traversals



coord

Graph Search on Coord (cont'd)

Progress in Graph Search

```
[woorung@coord graph]$ ./graph_test -n relation -x "김우현" "Coord" -v
+<김우현>,,Coord
  +김우현,프로젝트,<Coord>
  -김우현,학교,ICU
  -김우현,회사,NHN
+<ICU>,,Coord
        ----김우현,학교,ICU
  -ICU,학교,나잘라
+<NHN>,,Coord
        ----김우현,회사,NHN
  -NHN,회사,손예진
+<나잘라>,,Coord
        ----김우현,학교,ICU
        ----ICU,학교,나잘라
  +나잘라,프로젝트,<Coord>
  |--김우현,학교,ICU
  -나잘라,프로젝트,Hadoop
  -나잘라,회사,무한기술
+<손예진>,,Coord
        ----김우현,회사,NHN
        ----NHN,회사,손예진
  -손예진,프로젝트,Hadoop
  -손예진,학교,KAIST
+<Hadoop>,,Coord
        ----김우현,학교,ICU
        ----ICU,학교,나잘라
        ----나잘라,프로젝트,Hadoop
  -Hadoop,프로젝트,손예진
+<무한기술>,,Coord
        ----김우현,학교,ICU
        ----ICU,학교,나잘라
        ----나잘라,회사,무한기술
+<Hadoop>,,Coord
        ----김우현,회사,NHN
        ----NHN,회사,손예진
        ----손예진,프로젝트,Hadoop
  -Hadoop,프로젝트,나잘라
+<KAIST>,,Coord
        ----김우현,회사,NHN
        ----NHN,회사,손예진
        ----손예진,학교,KAIST
+<손예진>,,Coord
```

Found a direct pattern & another indirect pattern

김우현-프로젝트-Coord
김우현-학교-ICU-학교-나잘라-프로젝트-Coord

Reduction Rule
-if (A, 학교, x) and (B, 학교, x) then (A, 동문, B)

Demo

Conclusion

Coord Footprint

Coord Presentations

- 2008/2009 NHN DeView
- 2009 Korea Hadoop Community
- 2009 Samsung SDS
- 2009 SKC&C

Coord Applications & Community Activities

- Coord+Lucene
  - 2008 WoC(Winter of Code) Award - Coord+Lucene
  - 2009 SNU Class - Coord+Lucene(2009 NHN DeView Presentation)
- Distributed C Compiler
  - 2009 BITComputer Project(by XID Team)
- NaverLab OCR
- UCS
- Spam Wars (on-going)

Future Works

Coord

- Use apache avro instead of gsoap
- Redesign & newly implement dynamic service framework
- Dynamic clustering
- Failover
- Integrate into filesystems, databases, and lucene as well as memory

Coord MapReduce

- Improve performance of dust, warp, mapreduce
- Failover
- Various machine learning algorithms

Coord DB

- Parallel RDBMS + MapReduce such as Greenplum, Aster, HadoopDB
- SQL support with query optimization

Thank you.