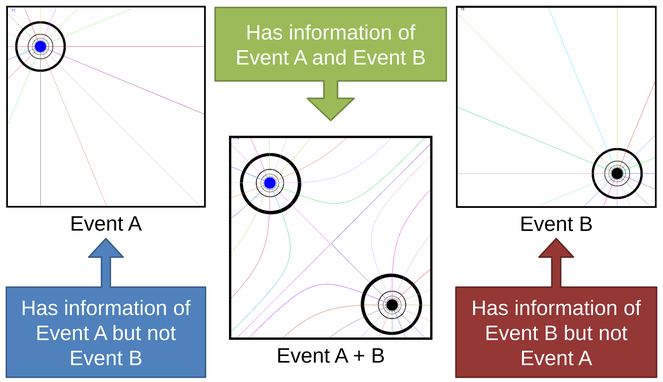
OSDC BigDataBus: Towards an Unified Architecture for Scientific Data Aggregation

Pedro D. Bello-Maldonado[1], Ana Oprescu[2], Paola Grosso[2], Heidi Alvarez[1], Indira Gutierrez[1]
[1]Florida International University, [2]Universiteit van Amsterdam

Introduction

Cross-disciplinary discoveries are brought by inter-disciplinary research and exchange of data/experiments. The Open Science Data Cloud (OSDC) platform is a growing venue for such data. However, one obstacle is the lack of a tool to discover data content and to allow data exchange, independent of the store format. To this end, we want to provide two types of tools: one that deals with categorizing the data content, by using generic semantics about it (what field it comes from, how it was obtained: experiment, benchmark, measurement, simulation); and another which deals with non-functional aspects of the data, e.g. type of database, geo-location.

Problem Description



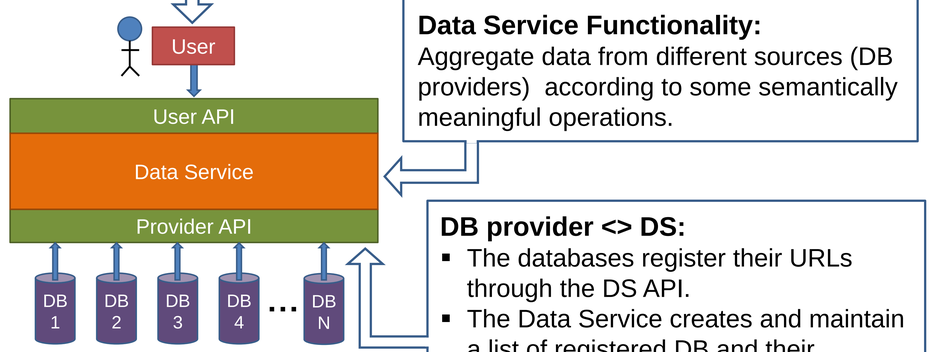
Research Questions

How can we create a tool that allow users throughout the world to *easily share* their *data* within *research communities*?

Can this tool unify the different data management alternatives (databases) so the users need not be concerned about the details of the data management but rather about the data itself?

Architecture

DS <> User:
Through the User API, the Data Service serves the respective data without bothering the user with the details of *how* the data is stored or *where* it is.



Data Service Functionality:
Aggregate data from different sources (DB providers) according to some semantically meaningful operations.

DB provider <> DS:
- The databases register their URLs through the DS API.
- The Data Service creates and maintain a list of registered DB and their information.

Conclusion

Two different parallelization approaches were applied here to solve the data aggregation process among different database paradigms. Using single threads to aggregate the data in each database led to performance decrease. However, dividing the data inside a single database, then using threads to aggregate it, and then repeating the process for other databases worked much better than the initial approach.

Acknowledgments

The authors of this work would like to thank the Partnership for International Research (PIRE) program, the Open Science Data Cloud (OSDC), and the National Science Foundation for their support of this work. We would like to extend our special thanks to Dr. Heidi Alvarez, Vasilka Chergarova, and all the members of the Center for Internet and Augmented Research and Assesment (CIARA) at FIU.

Parallelization and Results

One Thread Per Table Parallel V 1.0

MySQL Database

ID	Coord X	Coord Y	Intensity
S1	10	25	60
S2	15	21	53
S3	19	18	73
S4	21	36	64
S5	12	26	56

HBase Database

Name	A	B	C	D	E	F
Pos X	53	50	48	50	42	58
Pos Y	59	63	51	63	49	57
Value	59	45	63	70	66	62

Name	X	Y	Value
S1	10	25	60
S2	15	21	53
S3	19	18	73
S4	21	36	64
S5	12	26	56
A	53	59	59
B	50	63	45
C	48	51	63
D	50	63	70
E	42	49	66
F	58	57	62

MySQL Database

ID	Coord X	Coord Y	Intensity
S1	10	25	60
S2	15	21	53
S3	19	18	73
S4	21	36	64
S5	12	26	56

HBase Database

Name	A	B	C	D	E	F
Pos X	53	50	48	50	42	58
Pos Y	59	63	51	63	49	57
Value	59	45	63	70	66	62

One Thread Per Partition Parallel V 2.0