Enhancing VLAM Workflow Model with MapReduce Operations

Mikołaj Baranowski[a], Adam Belloum[a], Marian Bubak[a,b]

[a] *Informatics Institute, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands*

[b] *Departament of Computer Science, AGH University of Science and Technology, Mickiewicza 30, 30-059 Kraków, Poland*

{baranowski, a.s.z.belloum}@uva.nl, bubak@agh.edu.pl

Objective

Provide an easy to use and efficient Domain Specific Language for defining MapReduce operations in workflows.

Motivation

- Importance of MapReduce in processing big data
- Pig Latin and Sawzall – solutions based on Domain Specific Languages that provide simple and user-friendly access to MapReduce resources
- To get access to MapReduce resources, users have to use different environments for specifying and running MapReduce jobs along with other application models like workflows

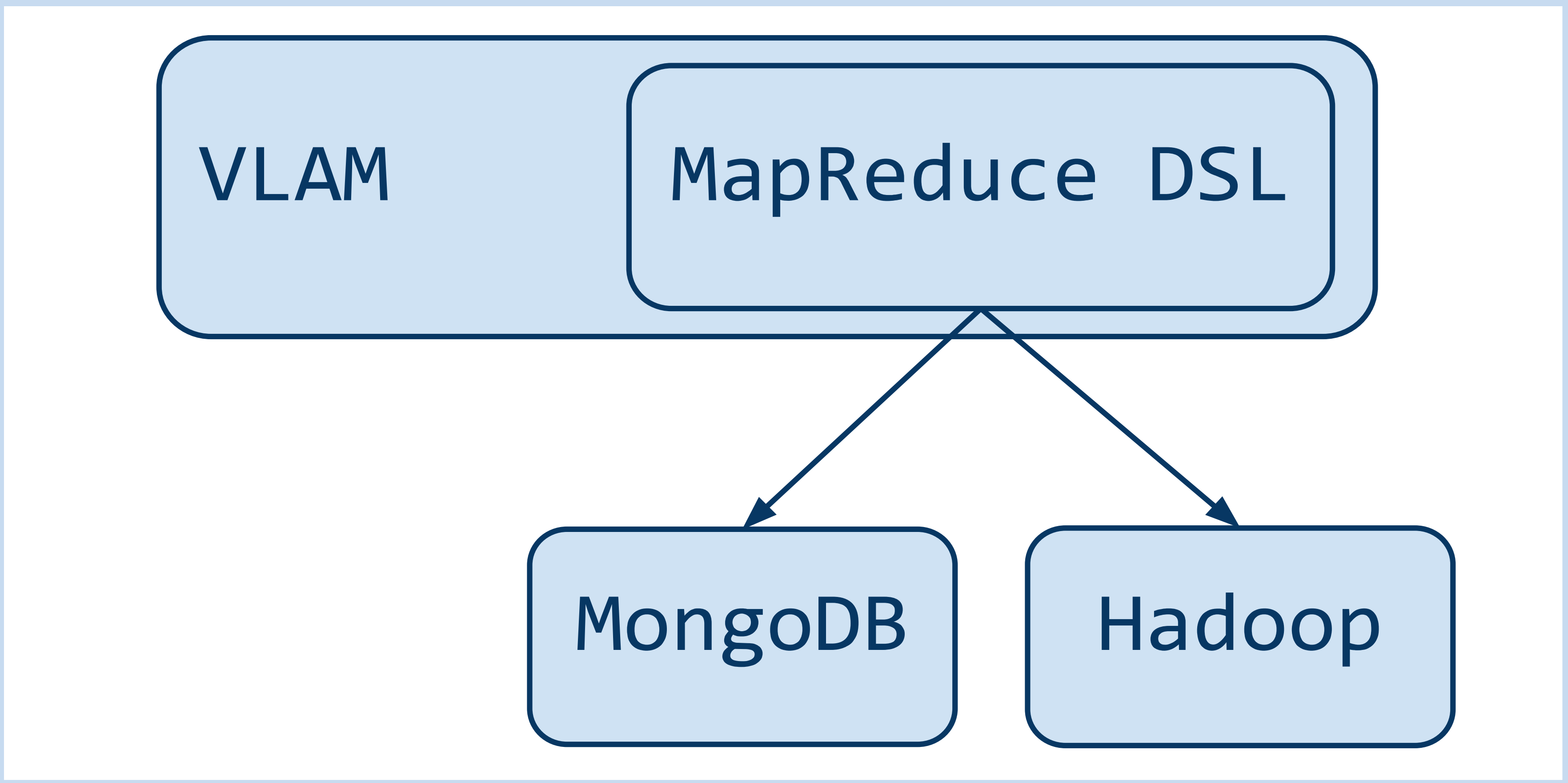

Figure 1: DLS can be used within an existing application (VLAM) to define operations for MapReduce framework

Design and Implementation

- Designed DSL describes only Map operation
  - Map operation is changed many times during the implementation process and the most of the execution time is spend on waiting for I/O operations
  - Users rarely change reduce and aggregate operations and they use a small number of them
  - The execution time strongly depends on reduce phase
- DSL translates Map operations to many platforms
  - Specifies types of processed data (required statically typed Hadoop reducers)
  - Defined with Ruby programming language which allows to choose an appropriate implementation

	Hadoop	MongoDB	Sawzall
Map operation	Java or any executable (streaming interface)	Javascript	Sawzall DSL
Reduce operations	Java or any executable (streaming interface)	Javascript	C++
Statically typed	Yes	No	Yes
User can define Reduce functions	Yes	Yes	No

Table 1: Features of MapReduce frameworks

Example application (word count)

- DSL was used to define Map operation (Listing 1)
- Special routines (`map`, `c.string`, `c.number`) were designed to simplify development of Map operations
- Sum reducer (`c.sum`) is included in the Hadoop distribution

```
map do |c, v|
  res = []
  v.split.each do |i|
    res << [c.string i, c.number 1]
  end
  c.sum(res)
end
```

Listing 1: Map operation from a word count application; analyzed value `v` is split into substrings and appended into `res` which is emitted at the end of the iteration

Conclusions

- Comparing to others, developed method provides a portable and pluggable solution
- The solution based on dynamic languages and DLS allows to define Map operation with a short, clear code
- It can be adapted to many existing applications thanks to the limited number of dependencies (Ruby)
- Map operations defined with the proposed DSL can be executed on many MapReduce platforms

Acknowledgements – This work was supported by the Dutch National Program COMMIT.

Bibliography

[1] A. Belloum, M. Inda, D. Vasunin, V. Korkhov, Z. Zhao, H. Rauwerda, T. Breit, M. Bubak, L. Hertzberger, Collaborative e-science experiments and scientific workflows, Internet Computing, IEEE 15 (2011) 39–47.

[2] C. Olston, B. Reed, U. Srivastava, R. Kumar, A. Tomkins, Pig Latin: a not-so-foreign language for data processing, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD '08, ACM, New York, NY, USA, 2008, 1099–1110.

[3] R. Pike, S. Dorward, R. Griesemer, S. Quinlan, Interpreting the data: Parallel analysis with Sawzall, Scientific Programming 13 (2005) 277.

[4] M. Baranowski, A. Belloum, M. Bubak, M. Malawski, Constructing workflows from script applications, Scientific Programming 20 (2012) 359–377

[5] M. Baranowski, A. Belloum, M. Bubak, MapReduce operations with WS-VLAM Workflow Management System, Procedia Computer Science 18 (2013) 2599-2602

University of Amsterdam

COMMIT/