WEB SERVICE INTEGRATED MATLAB PARALLEL COMPUTING

Dejan Vujičić, Željko Jovanović, Dijana Stojić, Uroš Pešović, Marina Milošević, Siniša Ranđić
University of Kragujevac, Faculty of Technical Sciences in Čačak, Serbia

Abstract
The possibility of program code parallelization has been studied intensively since the multi-core processors have hit the mainstream scene. Matlab as the programming environment possesses the necessary tools for parallelization and optimization of the code, if it is applicable. However, in the terms of web service native support, it lacks some functionalities, but they have been overcome by third party solution. This paper presents the possibility of web service integration with Matlab, in order to parallelize the program code and provide better results.

Keywords: Matlab, parallel computing, web service.

INTRODUCTION

The problem of the parallel execution of the code has been thoroughly studied in recent times, since the multi-core processors firstly appeared. Nowadays the programmers have to be aware of the parallelism possibilities, whether it is realized in code with special parallel directives, or in compiler. Matlab as a programming tool has support for parallel execution via Parallel Computing Toolbox. It possesses special directives and statements that help in parallelizing loops or making distributed arrays.

Making local parallel programs is fine, but if you need to access a computer with greater processing power remotely, then web services come in handy. They provide the environment for communication between server and clients. In our case, we used third party web service with full support in Matlab. The web service was initiated and started in Matlab, communicating with Apache Tomcat web server. Server application requires some parameters, that are supplied via GET HTTP request, and the processing times of sequential and parallel code is returned to the client.

The paper is organized as follows. The second chapter brings theoretical background on web services. The third chapter describes parallel computing in Matlab. The fourth chapter gives details about web service integration with Matlab. The fifth chapter brings results and discussion. At the end, there are concluding remarks.

WEB SERVICES

Web services are client and server applications that communicate over the World Wide Web's (WWW) HyperText Transfer Protocol (HTTP). They provide a standard means of interoperating between software applications implemented in any language running on a variety of platforms and frameworks. They are classified at the two main types [1]: StateFull and RestFull Web services. StateFull Web services are usually called SOAP services, while RestFull are usually called REST services. SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) are both web service communication protocols. SOAP was long the standard approach to web service interfaces, although it's been dominated by REST in recent years, with REST now representing more than 70% of public APIs.

StateFull Web services require several technologies and protocols to transport and to transform data between consumer and a service. The main ones are [2], [3]:

- Universal Description Discovery and Integration (UDDI) is a registry and discovery mechanism, that is used for storing and categorizing web services interfaces.

- Web Services Description Language (WSDL) defines the web service interface, data and message types, interactions, and protocols.
- Simple Object Access Protocol (SOAP) is a message-encoding protocol based on XML technologies, defining an envelope for web services communication. SOAP messages are exchanged using HTTP.
- Extensible Markup Language (XML) is the basic foundation on which web services (SOAP, WSDL, and UDDI) are built and defined.

With this characteristics web services can be integrated into MVC (Model View Control) or SOA (Service Oriented Architecture) application architecture.

Representational State Transfer (REST) Web services are even easier for integration since they do not need to be registered or described. That's why they are more represented nowadays. Usually, REST services are used for exposing a public API over the Internet, while SOAP exposes even part of logic as a service description.

Some of REST benefits over SOAP are [4], [5]:

- Greater variety of data formats, whereas SOAP only allows XML.
- Thanks to JSON support, REST offers faster parsing and better browser support.
- Faster and uses less bandwidth.

Some of SOAP benefits over REST are:

- Built-in retry logic to compensate for failed communications. REST, on the other hand, doesn't have a built-in messaging system. If a communication fails, the client has to deal with it by retrying.
- SOAP is a standardized protocol, and is provides better support for other standards and is easier for it to operate across firewalls and proxies.

PARALLEL COMPUTING WITH MATLAB

Matlab as a programming tool has native support for running parallel computations [6], [7]. Parallel execution of the code is possible on multicore processors, GPUs, and computer clusters. This toolbox helps in parallelizing Matlab applications with the following high-level constructs [8]:

- Parallel for-loops;
- Special array types, and
- Parallelized numerical algorithms.

Other than this, Matlab has support for distributed computing, as well as interactive Big Data processing tools [9]. Parallel Computing Toolbox can be used to execute applications on multicore processors, creating workers that work in parallel, or to harness the possibilities of modern GPUs with large number of processing units. Also, Matlab with its Distributed Computing Server can execute applications on a computer cluster (Fig. 1 [8]).

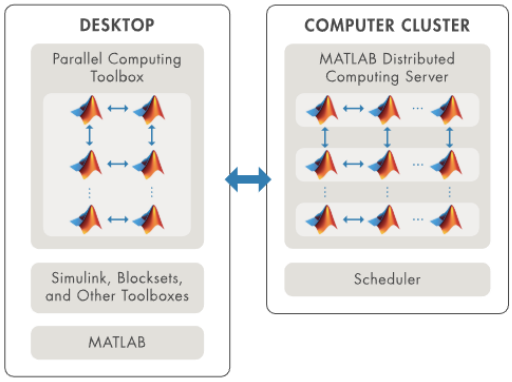

Fig. 1. *Parallel computing with Matlab [8]*

MATLAB WEB SERVICE INTEGRATION

Although Matlab has native support for calling web services, in order to make requests and create web service instances, a third party solutions have to be used. In our case, we used Modelit Matlab Webservice Toolbox [10]. Other than basic functionalities of creating web services, Modelit Matlab Webservice Toolbox provides utilities for [11]:

- Implementation of web services;
- Creation of XML output;
- Encapsulation of Matlab figures in HTML;
- Conversion of JSON strings to Matlab;
- Serialization and deserialization of encoded ASCII;
- Parallelization of tasks by making asynchronous calls to web services.

The deployment options for the Matlab Webservice Toolbox is shown in Fig. 2 [11].

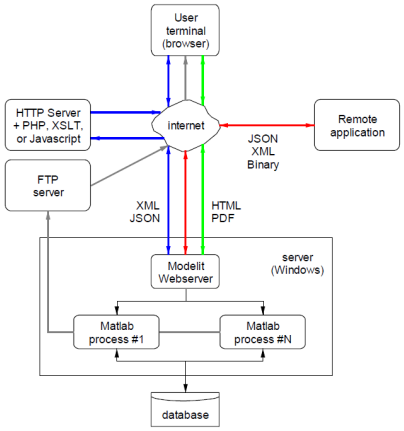

Fig. 2. *Matlab Webservice Toolbox deployment options [11]*

In order to make web service to work with Matlab, the installation of Apache Tomcat is necessary [12]. Before starting Tomcat, in its installations directory, the corresponding JAR file and Matlab executable code is necessary, in order to make a connection between Matlab and Tomcat.

A XML file has to be made in order to establish a connection. This XML file includes servlet name, port, and URL address to the server.

From Matlab command line, the **createMatlabServer()** function has to be called, with port number passed as an argument. After that, the server has to be started and it will automatically execute **HTMLCallback()** function, if there isn't any other argument set. In this function, we implemented code to call the function that runs sequential and parallel code and returns the time required for their completion. The code of the function is as follows:

```
function [time1,time2] = example(n,A)

a = zeros(1,n);

%% Sequential
tic
for i = 1:n
    a(i) = max(abs(eig(rand(A))));
end
time1 = toc;
```

```
%% Parallel
a = zeros(1,n);
tic
parfor i = 1:n
    a(i) = max(abs(eig(rand(A))));
end
time2 = toc;

end
```

As we can see from this code, the function receives two arguments, **n** (number of iterations) and **A** (size of the matrix). It then calculates maximum of the absolute values of eigenvalues of the random matrix in **n** iterations. The parallel code is realized with **parfor** statement, and it automatically creates default number of Matlab parallel workers (usually equal to the number of processor cores). The parameters **n** and **A** are provided via URL in the GET request, for example as follows:

http://localhost:8080/matlabserver/myExample?n=1&A=2

RESULTS AND DISCUSSION

In order to test web service implementation with Matlab, the function for finding eigenvalues of random matrices was called in both sequential and parallel way. The system specifications are given in Table 1.

Table 1. *The system specifications*

CPU:	Intel Core i7-4790K @ 3.6GHz (4 cores, 8 threads)
RAM:	16GB
OS:	Windows 10 Pro x64

With given parameters of **n = 200** and **A = 1000**, the Matlab produces results that are given back to the Tomcat web server. This is shown in Fig. 3. The default number of workers for the used processor is 4.

17-Oct-2018 17:47:01

Query: n=200&A=500

Sequential time: 26.8054 sec

Parallel time: 107.4163 sec

Fig. 3. *Response from Matlab web service*

If the URL GET request is not in the required format, the error message is displayed (Fig. 4).

17-Oct-2018 17:48:30

Query: m=200&A=500

Error: wrong arguments

Fig. 4. *Error message due to wrong URL GET request format*

We can see from Fig. 3 that time for parallel execution is significantly greater than time for sequential execution. This is due to the overhead resulted in the process of starting Matlab parallel pool with desired number of workers. If we run the query again, with parallel pool already created, we get the following response (Fig. 5).

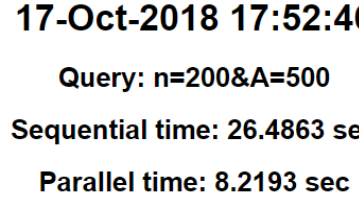

Fig. 5. *Response from Matlab web service with parallel pool already running*

We can see big improvements over time for parallel execution, because now there is not any delay due to parallel pool starting procedures.

Next, we decided to try different number of workers to see how much the speed of execution is dependent on the number of processor cores. So, we started parallel pool with 3, 2, and 1 worker and got the times of parallel execution. The time for sequential execution is approximately the same in all cases. These results are given in Table 2.

Table 2. *Correspondence between number of parallel workers and time of execution*

Number of workers	Time of execution (seconds)
4	8.2
3	9.7
2	13.2
1	25.1

As we can see from Table 2, there are significant improvements in execution times when the number of workers is bigger. Interesting thing is that parallel execution with one worker was less than 1s faster than sequential execution.

Of course, improvements due to parallelization are possible because the data dependencies are not present, i.e. individual cores can get the almost equal portions of the loop iterators to execute, and they don't need to communicate or wait for each other along the way. Only communication is done after the calculations. With different algorithms, data dependencies can occur in greater manner, so this may be an issue.

CONCLUSION

This paper demonstrated successful integration of web service functionalities with Matlab. Other than that, we showed performance improvements of the parallel code execution over sequential, using Parallel Computing Toolbox in Matlab.

Although the example we used was trivial and used just in demonstration purposes, nevertheless it showed significant improvements with parallel code execution and possibility to pass the data via web service.

We also tested the performance of the parallel execution with different number of workers and showed that the processing time is lower if there are more workers involved. This is a natural assumption, but only with the programs that don't possess data dependencies between different parts of the parallel pools. If that is not the case, i.e. there are data dependencies and low possibility of parallelism, then the possible solution lays in the different approaches in parallelization, primarily by implementing different algorithms that support parallelism.

ACKNOWLEDGMENT

The work presented in this paper was funded by grant TR32043 for the period 2011-2018, by the Ministry of Education, Science, and Technological Development of the Republic of Serbia.

REFERENCE

[1] Pautasso, C., Zimmermann, O., & Leymann, F. (2008, April). Restful web services vs. big'web services: making the right architectural decision. In *Proceedings of the 17th international conference on World Wide Web* (pp. 805-814). ACM.

[2] Wagh, K., & Thool, R. (2012). A comparative study of soap vs rest web services provisioning techniques for mobile host. *Journal of Information Engineering and Applications*, 2(5), 12-16.

[3] Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., & Weerawarana, S. (2002). Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. *IEEE Internet computing*, 6(2), 86-93.

[4] Richardson, L., & Ruby, S. (2008). *RESTful web services*. "O'Reilly Media, Inc.".

[5] Rodriguez, A. (2008). Restful web services: The basics. *IBM developerWorks*, 33.

[6] *Sharma, G., & Martin, J. (2009). MATLAB®:* a language for parallel computing. *International Journal of Parallel Programming*, 37(1), 3-36.

[7] Choy, R., & Edelman, A. (2005). Parallel MATLAB: Doing it right. *Proceedings of the IEEE*, 93(2), 331-341.

[8] *Perform parallel computations on multicore computers, GPUs, and computer clusters*, available at: https://www.mathworks.com/products/parallel-computing.html (accessed on: October 2018)

[9] Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues. *Information Systems*, 47, 98-115.

[10] Modelit Webserver Toolbox for Matlab, available at: https://www.modelit.nl/index.php/webserver-toolbox-for-matlab (accessed on: October 2018)

[11] Van der Zijpp, N. J., Hoogland, K. J. (2016). User Guide for the Modelit Matlab Webservice Toolbox. *Modelit*

[12] *Apache Tomcat*, available at: http://tomcat.apache.org/ (accessed on: October 2018)