queries in a number of formats and standards. It is considered nowadays to be the standard triple store server to be used in many Semantic Web applications and is characterized by its ease of setup and well written API. The server has been configured accordingly to be more scalable. Fuseki also allows reasoning by using rules. For the purposes of the present work reasoning may be performed to obtain relations between stored arguments. The main task is to obtain arguments that are related to specific entities, e.g. Chemopreventive Agents.

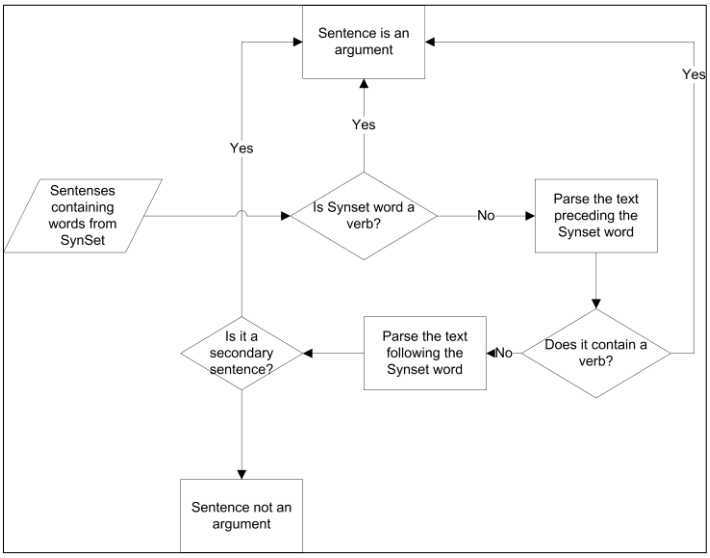

Figure 2. Grammar rules diagram

7. PROTOTYPE APPLICATION

In the context of this work, we have created an easy to use web interface (Figure 3) that allows users to upload a document as PDF and perform automatic extraction on the fly (apart from having a series of publications to be inserted in the pipeline). This application is provided in the form of a widget and can be located in the GRANATUM platform (http://www.granatum.org/bscw). The platform is open, but a registration is needed. The extraction results are presented in facets: we categorized the argument sentences in terms of associated entities that have been identified (chemoprevention agents, molecules etc). We also allow a user to perform search of terms in the RDF triplestore. If the retrieved terms in the argument sentence contain a term for which we stored a Linked Data URL we allow the user to click on that term to start browse the Linked Data Space. A functionality of retrieving similar arguments to the current ones being displayed (in terms of having links to similar Linked Data entities URI's), allows the user to check bibliography in a way that can assess what the current status on specific drugs is.

8. EXPERIMENTAL RESULTS

In order to evaluate the performance of the architecture we have taken time measurements regarding core functions of the solution. The functions that were evaluated were a) the pipeline responsible for argument and entities extractions and b) the triple store query engine.

In order to measure the time performed for argument and entities extraction, the GATE pipeline was run in batch mode directly from the GATE Developer Environment on a corpus of papers taken from PubMed's Open Access Subset in XML format. A total of 400 publications were selected. The publications were mainly published in journals related to cancer research. The machine hosting the pipeline featured an INTEL i7-2670QM processor and a total of 6GB of RAM. The total time for processing the corpus was 602.3 seconds. The time measured does not take into account the extra time needed to query the PubMed Database for extracting this extra information.

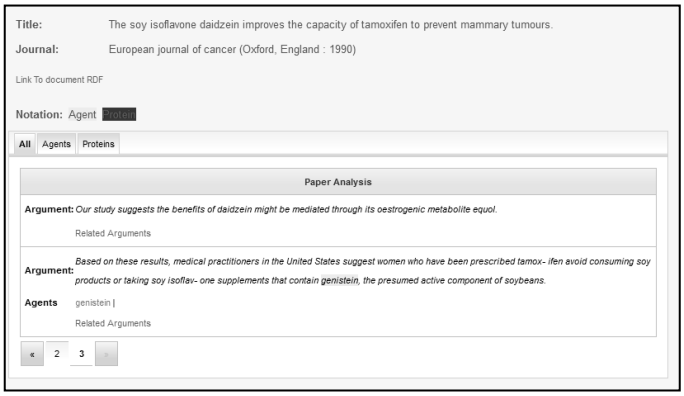

Figure 3. On-the-fly named entity extraction results

For the query engine sample queries were performed against a set of arguments already extracted in the context of the GRANATUM project. The number of arguments were a total of 13,853 and the database resided in the same local network with the machine requesting the data. In the first scenario, the total number of arguments was selected along with their associated entities. The query depicted in Table 2 was executed in 9.3 seconds.

Table 2. Query for selection of all arguments and their referred entities

```
SELECT DISTINCT * WHERE {
  ?arg onto:argumentSentence ?o .
  ?arg onto:appearsIn ?s .
  ?arg onto:refersTo ?mol .
}
```

For the second scenario we performed the above search query with the extra condition that the text of the argument contained the text "ER β" corresponding to ER-beta proteins (Table 3). The query was completed in 8.7 seconds.

Table 3. Extraction of arguments that contain text references to ER-beta proteins

```
SELECT DISTINCT * WHERE {
  ?arg onto:argumentSentence ?o .
  ?arg onto:appearsIn ?s .
  ?arg onto:refersTo ?mol .
  FILTER(regex(<http://www.w3.org/2001/XMLSchema#string>(?o),"ER β","is")).
}
```

9. CONCLUSION

In this paper we discussed the importance of automatically extracting information about biomedical entities from scientific publications. We have presented the approaches that were followed up until now in the area of text mining. We introduced the importance of Semantic Web and Linked Open Data, as presented an application that can automatically extract entities from the publications and connect them to Linked Data instances. The system was based on GATE text mining tool as well as open source tools for handling the triples produced. Finally, we have presented a prototype we application that allows a user to interface with the structured data and browse the data.

The work that has been done can receive some extensions that will elevate the knowledge that was gathered. First of all, efforts can be made to provide a ranking to the results page. At the moment results are

returned in an unordered fashion. By providing a ranking mechanism (while taking into consideration other factors such as owl:sameAs links between instance data, how many authors provide the same arguments etc) we can ensure that the most prominent results to be returned to the end user. Also, a social aspect of the system can be provided allowing a user to connect and share to other social network platforms and provide a way for people to discuss these results and give feedback.

ACKNOWLEDGEMENT

This work has been co-funded by the GRANATUM project, a European Commission research program under Contract Number FP7-270139.

REFERENCES

[1]. A. Passant et al., 2009. SWAN/SIOC: Aligning scientific discourse representation and social semantics, Proceedings of the Workshop on Semantic Web Applications in Scientific Discourse (SWASD 2009), 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA

[2]. Belleau F. et al., 2008. Bio2RDF: towards a mashup to build bioinformatics knowledge systems. Journal of biomedical informatics, pp. 706-716

[3]. Bizer,C. et al., 2009. Linked data-the story so far, International Journal on Semantic Web and Information Systems (IJSWIS),5(3), pp. 1-22

[4]. Cunningham H. et al., 2002. GATE: an Architecture for Development of Robust HLT Applications, ACL '02 Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 168-175

[5]. Cunningham H. et al., 2013. Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. PLoS computational biology 9.2

[6]. Doms, A. and Schroeder M., 2005. GoPubMed: exploring PubMed with the gene ontology. Nucleic acids research

[7]. Granatum Consortium, 2009. D1.3 – GRANATUM Biomedical Semantic Model

[8].Handschuh, S. et al., 2002. S-CREAM-semi-automatic creation of metadata. Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, pp. 165-184

[9]. Jena Fuseki, 2013. https://jena.apache.org/documentation/serving_data

[10]. Klyne G. et al., 2004. Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation

[11]. Popov, B. et al, 2003. KIM–semantic annotation platform. The Semantic Web-ISWC

[12]. Rebholz-Schuhmann,D. et al., 2007. EBIMed—text crunching to gather facts for proteins from Medline Bioinformatics, 23(2), e237-e244

[13]. Tempich C. et al., 2005. An argumentation ontology for distributed, loosely-controlled and evolving engineering processes of ontologies (DILIGENT). The Semantic Web: Research and Applications. Springer Berlin Heidelberg, pp. 241-256

[14]. Vargas-Vera et al., 2002. MnM: Ontology driven semi-automatic and automatic support for semantic markup. Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web, pp. 213-221.

[15]. WordNet, 2013, http://wordnet.princeton.edu

CLOUDPDE: A WEB SOLVER OF DIFFERENTIAL EQUATIONS BY FINITE DIFFERENCE DISCRETIZATION

Fabio Takeshi Matsunaga, José Luiz Vilas Boas, Neyva Maria Lopes Romeiro
and Jacques Duílio Brancher
Departamento de Computação, Universidade Estadual de Londrina (UEL) – Rod. Celso Garcia Cid, Cx. Postal 10011, Londrina-PR Brazil

ABSTRACT

The Internet and Web technologies have been widely applied on many scientific fields, such as mathematic, physic, chemistry and engineering. These scientific applications are migrating to cloud and web services, which are becoming a new paradigm of computational applications. The aim of this work is to develop a cloud service named CloudPDE that uses the concept of cloud computing to solve problems involving partial differential equations, such as heat and wave equations, with 1 and 2 dimensions by finite difference discretization method to solve large problems. The system consists of a front-end and back-end client-server communication, a database of results and user/request information storage and a graphical user interface for outputs visualization. The CloudPDE service can be executed on devices independent on their performance, since the high-order matrixes computations and high-cost calculations required for accurate simulations are made on a remote server.

KEYWORDS

Back-end, Front-end, Linear Algebra, Scientific Applications, Web Service

1. INTRODUCTION

Cloud computing and Web Services aim to provide services which applications, platforms or infra-structures are located on remote servers and data-centers, deliver and manage services over the internet, also allowing users to access remote files and applications. These computer models have characteristics such as accessibility, scalability, coherency, virtualization and interoperability and (Moghe et al., 2012). Thus, many internet services have been developed in many scientific applications, such as mathematic, physic, chemistry and engineering.

Most specifically, in many works this technology has been focused to numerical simulations and partial differential equations solutions (Ari and Muhtaroglu, 2012; Jorissen et al., 2012). Such scientific applications often require complexes and high-cost calculation, large amount of data, high performance environment and several computational resources. Furthermore, cloud services applications have being developed for real-time interaction and communication with users (Villegas-Puyod, 2012), which requires high performance algorithms and methods.

Problems involving mathematical applications are difficulty for conventional means in classrooms, because of the extensive theory and experimental knowledge. Some developed web tools (Grande et al., 2010; Basak, 2012) for mathematical and physical phenomena visualization uses video recording and interactive tools which require a large computer memory, ideal hardware configuration and local softwares or plug-ins installations for ideal design and responses.

There are many advantages and motivations of web technologies for numerical simulations (Perus et al., 2013), such as the automatic licensing, system maintenance, upgrade and version control made from a web server for global use and facility of deployment. Furthermore, finite difference discretization method is a straightforward method and for refined mesh, it is necessary high cost linear algebra operations, such as linear systems resolution (Reimer and Cheviakov, 2013) or sparse matrix-vector multiplication and a huge amount of data requiring a remote database for information storage and server applications which are maintained and easily accessible from anywhere and anytime.

Considering the above, this paper presents the development of a service called CloudPDE to solve 1D and 2D problems solutions involving time dependent partial differential equations (PDE), such as heat conduction and wave propagation equation. The method used to solve these equations is the finite difference discretization considering Cartesian coordinates. For this purpose, a real time front-end client-server communication and a back-end to process all the calculations in remote server independent on user's machine or device performance, were developed in order to help the understanding of heat transfer phenomena and interaction with surfaces and boundary conditions.

Thus, the structure of this paper is as follows: Section 2 presents a review of literatures about related works; Section 3 presents the finite difference discretization method for 2D heat conduction equation and 1D wave propagation equation. In Section 4, we described the CloudPDE architecture to support the demand of finite difference calculation. Section 5 shows some results examples of both cases of equations. Section 6 is the conclusions and some future works.

2. LITERATURE REVIEW

Several works have been developed to solve differential equation by finite difference discretization method. Russel and Probert (2004) for example developed the Fdiff3 educational software to solve heat conduction by finite difference method, aiming to facilitate the understanding of the physical phenomena. Reimer and Cheviakov (2013) developed a Matlab simulator to solve problem involving Poisson Equation, a time-independent PDE to calculate the static heat distribution, in rectangular, cylindrical and spherical coordinates, with mesh refinement control in a local application. However, many scientific applications are migrating to the Internet, providing a new paradigm of softwares and computing services.

Considering the advents of this new computing paradigm, several web and cloud services for scientific applications have been developed recently. One of the most relevant and recent, we can cite the Ari and Muhtaroglu (2012) work. On this work, the authors implanted a cloud service for linear and non-linear mechanical structures, using finite elements method to solve PDEs. In the same research line, Xiao-ping and Hu (2010) developed a web service for remote finite elements analysis focused on ideal design of complex products aiming the local fast modeling.

Another work with the same approach was developed to support scientific applications, such as a PaaS that provides some material sciences and quantum chemistry applications source-codes and benchmarks and interface tools (Jorissen et al., 2012). This work developed a platform using high performance resources and the scalability was tested in cloud clusters distributed on Amazon Web Services virtual machines.

In this same line of research, Costa et al., (2012) for example, established an information management, data analysis and processing technique to numerical simulations of natural phenomena in scientific cloud applications. The authors mentioned that an accurate simulation is performed considering a high scale simulation with more level of detail and a fine grained mesh representation of a physical domain. The main work challenges were the simulation data storage, the physical data representation and data management.

3. FINITE DIFFERENCE DISCRETIZATION METHOD

The numerical method chosen to solve PDEs was the finite difference discretization. This method has been applied to solve many PDEs in several scientific areas (Yang and Sen, 2009). The main idea is to solve PDEs using finite difference equations obtained by Taylor series to approximate the partial derivatives. Then, this method consists on spatial domain discretization in a computational mesh using the finite difference equations to approximate the values of each mesh point. The next sections will present the kinds of PDEs to be discretized and considered on this work.

3.1 One Dimensional Wave Equation

On the one dimensional wave equation (1), different from 2D heat equation, we considered the mesh dimension of x axis $[x_0,x_f]$, the final time simulation ($Tmax$), the boundaries conditions given by $u(x_0,t)$ and

$u(x_f,t)$, the initial condition $u_0(x,0)$, the initial speed condition $\frac{\partial u}{\partial t}(x,0) = g(x)$ and a C constant. The mesh is discretized according of N_x and N_t partitions, which refers to number of partitions in x and t respectively, and the spacing of each mesh node $\Delta x = (x_f\text{-}x_0)/(N_x\text{-}1)$ and $\Delta t = (T_{max})/(N_t\text{-}1)$.

$$\frac{\partial^2 u}{\partial t^2} = C^2 \frac{\partial^2 u}{\partial x^2} \quad (x_0 \leqslant x \leqslant x_f) \quad (1)$$

The Equation 1 discretization consists on substitute the second order partial derivatives on time-domain and the second order partial derivative on spatial-domain finite difference equation (2), and the initial speed condition application $g(x)$ for $u_{i,1}$ (3).

$$\frac{\partial^2 u}{\partial t^2} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{\Delta t^2} \quad \frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\Delta x^2} \quad (2)$$

$$u_{i,1} = (1-\lambda^2)u_0(x_i,0) + \frac{\lambda^2}{2}u_0(x_{i+1},0) + \frac{\lambda^2}{2}u_0(x_{i-1},0) + C\,g(x_i) \quad (3)$$

Considering $u_{i,j}$ the time-step mesh approximation in a time t_j, a spatial x_i and $\lambda=(C.\Delta t)/\Delta x$, the scheme of the discretized equation is showed in (4). It is necessary that $\lambda \leq 1$ for the stability condition and the non-oscillation during the simulation.

$$u_{i,j+1} = 2(1-\lambda^2)u_{i,j} + \lambda^2 u_{i+1,j} + \lambda^2 u_{i-1,j} - u_{i,j-1} \quad (4)$$

The equation above (4) leads to a matrix-vector multiplication, where the matrix A is a tridiagonal formed of a main diagonal [$2(1\text{-}\lambda^2)$, $2(1\text{-}\lambda^2)$, $2(1\text{-}\lambda^2)$ … $2(1\text{-}\lambda^2)$], a subdiagonal and a superdiagonal formed of [λ^2 ,λ^2,λ^2…λ^2] elements, and the vector is $u_{i,j}$ elements. As we can see on (4), the discretized equation finds the t_{j+1} time step of wave propagation, then the vector resulted from the multiplication is subtracted from $u_{i,j-1}$, which is found by the initial speed condition equation (3). For these operations we used routines from BLAS/LAPACK (2013) library, which has high-performance linear algebra functions that execute vector operations efficiently on shared memories and parallel processors. The routines used to perform these operations are SBMV (matrix-vector operation for banded matrix) for tridiagonal matrix-vector multiplication and a SAXPY (vector addition with scalar multiplication) operation. for vectors subtractions. Then, the operation to find $u_{i,j+1}$ is SAXPY(SBMV(A, $u_{i,j}$), $u_{i,j-1}$, -1), where -1 indicates that the SAXPY performs a vector subtraction.

3.2 Two Dimensional Heat Equation

On the two dimensional heat equation (5), we consider the mesh dimensions $[x_0,x_f]$ e $[y_0,y_f]$, the final time simulation ($Tmax$), the boundaries conditions given by $T(x_0,y,t)$, $T(x_f,y,t)$, $T(x,y_0,t)$ e $T(x,y_f,t)$ and the initial condition $T_0(x,y,0)$. The mesh is discretized according of N_x, N_y and N_t partitions, which refers to number of partitions in x, y and t respectively, and the spacing of each mesh node $\Delta x=(xf\text{-}x0)/(Nx\text{-}1)$, $\Delta y=(y_f\text{-}y0)/(N_y\text{-}1)$ and $\Delta t=(Tmax)/(N_t\text{-}1)$. The K is the thermal diffusivity constant (cm²/s) and the value depend on the material.

$$\frac{\partial T}{\partial t} = K\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) \quad (x_0 \leqslant x \leqslant x_f\,;\, y_0 \leqslant y \leqslant y_f) \quad (5)$$

The Equation 1 discretization consists on substitute the first order partial derivatives on time-domain by the regressive difference equation (6) and the second order partial derivative on spatial-domain by the central difference equation (7).

$$\frac{\partial T}{\partial t} = \frac{T_{i,j}^{k} - T_{i,j}^{k-1}}{\Delta t} \quad (6)$$

$$\frac{\partial^2 T}{\partial x^2} = \frac{T_{i-1,j}^{k} - 2T_{i,j}^{k} + T_{i+1,j}^{k}}{\Delta x^2} \quad \frac{\partial^2 T}{\partial y^2} = \frac{T_{i,j-1}^{k} - 2T_{i,j}^{k} + T_{i,j+1}^{k}}{\Delta y^2} \quad (7)$$

Considering $T_{i,j}^{k}$ the time-step mesh approximation in a time t_h and a spatial point (x_i,y_j) and $\alpha = (K.\Delta t)/\Delta x^2$ and $\beta = (K.\Delta t)/\Delta y^2$, we obtain the implicit scheme of the discretized equation (8).

$$T_{i,j}^{k} = (1 + 2\alpha + 2\beta)T_{i,j}^{k+1} - \alpha T_{i+1,j}^{k+1} - \alpha T_{i-1,j}^{k+1} - \beta T_{i,j+1}^{k+1} - \beta T_{i,j-1}^{k+1} \quad (8)$$

The equation above (8) refers to a block tridiagonal linear system with multiple diagonals (Reimer and Cheviakov, 2013), which solution are the values of $T_{i,j}^{k}$ in a set of interior points of the mesh. These linear systems are solved by LU factorization adapted to special matrixes, provided by the LAPACK (Linear Algebra Package) library. In this case, the routine SGBSV were selected, because it is optimized to banded matrixes with multiple diagonals, which is stored as an array for each column j: $AB(KU+KL+i-j, j) = A(i,j)$ em que $Max(0,j-KU) \leq i \leq Min(n,j+KL)$, where KU is the number of superdiagonals, KL is the number of subdiagonals and n the banded matrix A order. This storage means make the SGBSV routine efficient to solving linear systems, considering only the non-zero values of matrixes.

4. CLOUD SERVICE ARCHITECTURE

This section presents a general architecture of the CloudPDE (Figure 1) composed of a front-end of client-server communication and interaction, and a back-end that executes the applications of each requester user in a remote server.

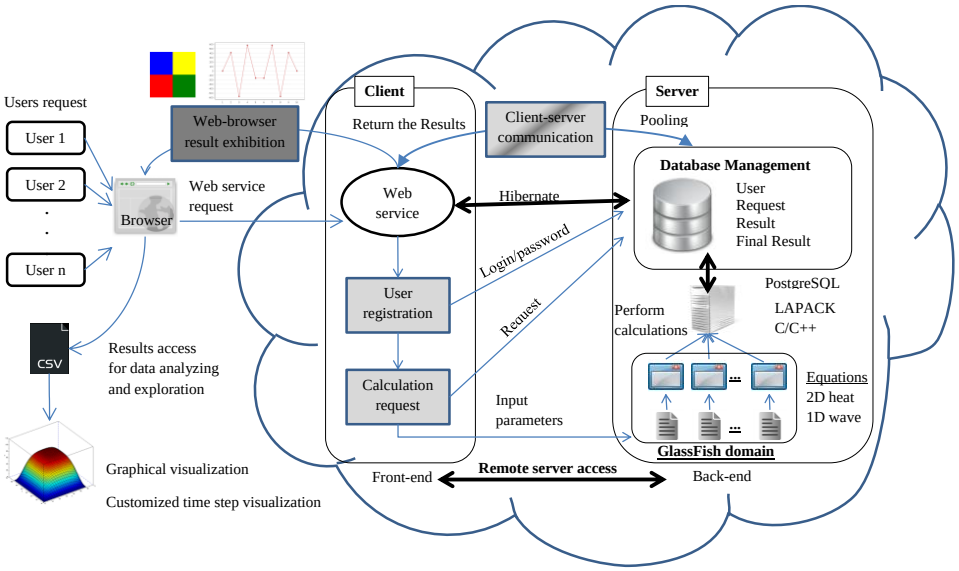

Figure 1. CloudPDE architecture (front-end, back-end and user interface) and main modules

The main modules presented on the cloud service architecture (Figure 1) are described below:

1. **User registration**: the user makes the registration and mounts the problem, entering the parameters.
2. **Calculation request**: Send the problem to the solver contained on a remote server.

3. **Client-server communication**: The server receives the problem and start solving it. For each iteration, the server sends the partial solution to the client.
4. **Web-browser result exhibition**: The client shows the results in the browser.

The client and server web services are built in NetBeans IDE as a Java Servlet application, and the deploy of them was made on GlassFish 3.1 server. The PostgreSQL 9.2 database was used as mean of calculation and output storage and queries and the client-server database communication are performed by Hibernate Framework. The finite difference server application was developed in C/C++ language with PostgreSQL integration to store the outputs on database and LAPACK integration to perform the optimized and efficient linear algebra calculations.

4.1 Front-end and Back-end

The front-end of the cloud service is responsible to information and inputs/outputs exchange between client and server, providing a real time interaction with the users. The main way to make this interaction is the database that stores all the related information (users, problems and solutions). The entity-relationship model of database is shown on Figure 2.

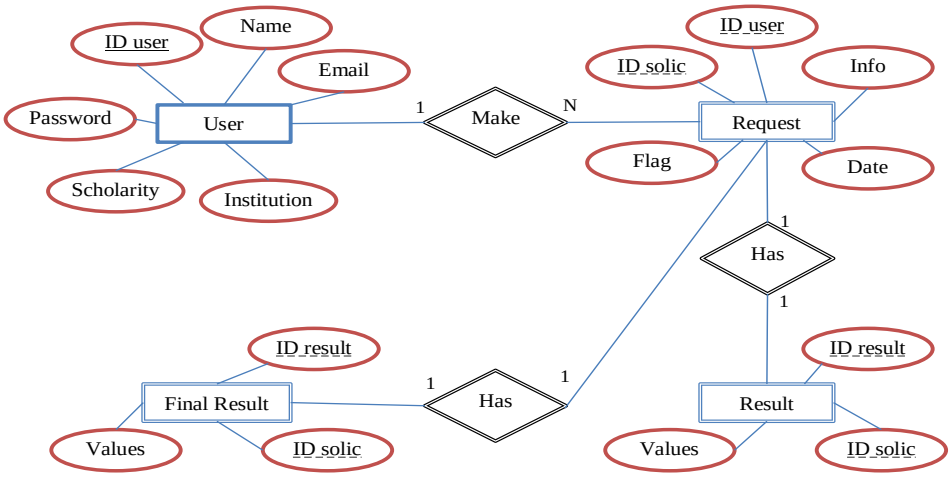

Figure 2. Entity-relationship model of client-server communication database.

As we can see on Figure 2, when a request is made, the front-end inserts the user information on ‘User’ table and the respective request information, with flag=’A’ on ‘Request’ table, indicating the calculation are being processed. The front-end also sends the users input to server as a text file format indexed with the user ID. This file is stored on the GlassFish server domain and is read by the back-end application, which is responsible to perform the calculations and updates the results on each time instance.

While the calculations are performed on each Δt instance, the results are inserted on ‘Result’ table as *varchar*(1000000) format and mirrored to ‘Final Result’ table performed by a trigger function until *Tmax* time and the flag=’A’ are maintained. When the calculation is terminated, the flag field is assigned as ‘F’, activating another trigger to remove all results information related to a determined ID request. This way makes the ‘Result’ a temporary table, aiming to not degrade the performance during the queries and real time results exhibition, especially when many simultaneous users are accessing the service.

The 'Final Result' table is a history of results of 'Result', utilized to future queries, in the case when a new user define a problem to solve, the cloud service makes a search on the database to verify their existence. If the return is true, the solved problem is presented to the users. This management database model allows the sharing of previously solved problems, existing solutions and also provides an economical use of remote server resources, so it is not necessary to solve a problem whose solution is already stored in the database. In addition, it is possible to available the solution to any users and organize the information by date of request, name, e-mail or ID of users and institutions that requested the processed problems, creating a collaborative system accessible from anywhere.

4.2 Graphical User Interface and Mesh Visualization

While the calculations are performed, the respective results instances are exhibited to the user. The results queries and real time exhibition are performed by a component called pooling, from Prime Faces framework. The pooling makes asynchronous calls every three seconds, which is considered the ideal maximum time response for real time tasks (Villegas-Puyod, 2012), to verify if there is any new line inserted in a 'Result' table, that's it, if there is a new state of the 1D bar (wave equation) or 2D surface (heat equation) in a time instance. If the pooling process detects a new line inserted, the values of discretized mesh points are written on browser on CSV (Comma Separated Values) format. On the end of processing, all the mesh points values on all t time instances ($0<t<Tmax$) for both cases of equations are available to the user.

Furthermore, there is an option where the user can view the graphics correspondent to each time instance $0<t<$Tmax. For the 1D visualization, we used the JFreeChart framework to plot line graphics to represent the temperature distribution and variation along the time. The visualization process is performed by taking the tuple of 'Result' table, which values are CSVs represented as string, inserted in a DataTable (from PrimeFaces) grid and inserted on a JFreeChart dataset.

For the 2D discretized mesh visualization, it is used the chain colors of 15 shades for positive and negative temperatures values, where the largest and the smallest values of temperature in a t is divided by 15 (number of shades), from which the interval colors is constructed and a HTML table is built to represent the surface heat distribution. In this table mesh, the user can access the temperature values on each discretized region, as well as having access to meta-information, such as (x,y) position, the influential boundary condition and the temperature variation with respect to the previous t value.

Despite the simplicity of our graphical output visualization, the visualization of discretized mesh of heat distribution can be performed on any device with HTML and Java support browsers, such as personal computers, mobile devices (tablets and smartphones). The real-time visualization of heat conduction phenomena is possible due to the simple configuration of the mesh generated by colored HTML table.

5. RESULTS

To check the viability of the system, we solved two problems. The first one is a 1D wave equation (Figure 3) with: $C = 1$, $T(x_0,t)=0$, $T(x_f,t)=0$, the initial condition $T_0(x,0) = sen(\pi x)$ and the initial speed condition $g(x)=0$. The mesh dimension specified was [0,1] and the mesh size $N_x=1000$, implying in a resolution of a tridiagonal linear system of order 998. The final time instance was $T_{max}=2$ with $N_t=100$ time steps.

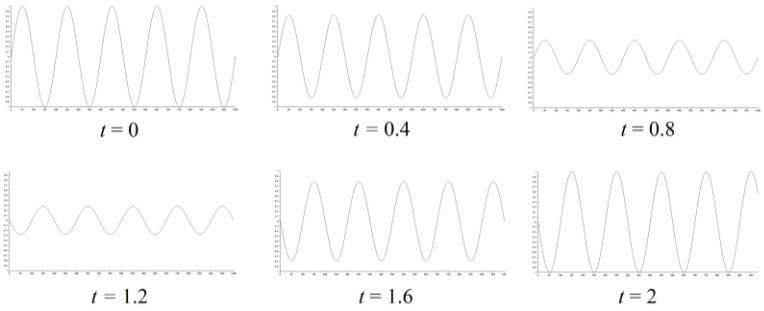

Figure 3. Line-charts of 1D wave equation outputs extracted in some time instances.

As we can see on Figure 3, the sinusoidal wave amplitude decreases over the time, until when approximately $t=1$ with the wave length constant. After that, the amplitude increases until the amplitude of the initial time step, however with displacement in x. Then, the wave equation can simulate this amplitude changing phenomenon periodically.

For the 2D heat equation (Figure 4), we considered $K=10^{-3}$, $T(x_0,y,t)=e^y-cos(y)$, $T(xf,y,t)=e^ycos(4)-e^4cos(y)$, $T(x,y_0,t)=cos(x)-e^x$ and $T(x,y_f,t)=e^4cos(x)-e^xcos(4)$ and the initial condition $T_0(x,y,0)=100$. The mesh dimension specified was $x=[0,4]$, $y=[0,4]$ and the mesh size $N_x=N_y=101$, which implies the resolution of a linear system of order 10,000 (matrix order of 10,000x10,000). The final time instance considered was $T_{max}=60$ and the number of time steps specified was $N_t=5$.