A Grid-based Cyber infrastructure for High Performance Chemical Dynamics Simulations

Khadka Prashant and Yu Zhuang
Department of Computer Science, Texas Tech University
Lubbock, Texas 79409-3104, USA

and

Upakarasamy Lourderaj and William L. Hase
Department of Chemistry and Biochemistry, Texas Tech University
Lubbock, Texas 79409-1061, USA

ABSTRACT

Chemical dynamics simulation is an effective means to study atomic level motions of molecules, collections of molecules, liquids, surfaces, interfaces of materials, and chemical reactions. To make chemical dynamics simulations globally accessible to a broad range of users, recently a cyber infrastructure was developed that provides an online portal to VENUS, a popular chemical dynamics simulation program package, to allow people to submit simulation jobs that will be executed on the web server machine. In this paper, we report new developments of the cyber infrastructure for the improvement of its quality of service by dispatching the submitted simulations jobs from the web server machine onto a cluster of workstations for execution, and by adding an animation tool, which is optimized for animating the simulation results. The separation of the server machine from the simulation-running machine improves the service quality by increasing the capacity to serve more requests simultaneously with even reduced web response time, and allows the execution of large scale, time-consuming simulation jobs on the powerful workstation cluster. With the addition of an animation tool, the cyber infrastructure automatically converts, upon the selection of the user, some simulation results into an animation file that can be viewed on usual web browsers without requiring installation of any special software on the user computer. Since animation is essential for understanding the results of chemical dynamics simulations, this animation capacity provides a better way for understanding simulation details of the chemical dynamics. By combining computing resources at locations under different administrative controls, this cyber infrastructure constitutes a grid environment providing physically and administratively distributed functionalities through a single easy-to-use online portal

Keywords: cyber infrastructure, chemical dynamics simulations, grid computing, seamless integration, online animation

1. INTRODUCTION

Chemical dynamics simulation is a powerful computing tool to study atomic level motions of molecules, collection of molecules, liquids, surfaces, interfaces of materials, and chemical reactions. These simulations have important applications in many research fields, including nanoscience, material and environmental sciences, molecular biology, and biochemistry. The Hase research group has developed the VENUS chemical dynamics computer program [1] for performing such simulations, which they use for their research as well as distribute to others. In an effort to make VENUS globally accessible to a broad range of users, a cyber infrastructure [2] was developed for accessing VENUS, which, through a website, allows a user to run a chemical dynamics simulation by selecting or building a simulation model and submitting the simulation execution job, and the modified or selected model gets executed on server machine, and users can download the output file. This web portal has been proven to be a valuable resource for research and education, as evidenced by visits of the website from researchers, educators, and students within the US and abroad.

In the original implementation of the VENUS-accessing cyber infrastructure, the simulation jobs were executed in web server machine. Computational time for a simulation varies from a few seconds to hours, sometimes even to days. For medium and large simulation jobs, the server machine is obviously not powerful enough — a more powerful cluster is needed to cut days-long jobs into hours-long and hours-long to minutes-long by running these time-consuming simulations on the cluster in parallel. Another drawback of doing everything on the server machine is that, as the number of user requests increases, response time gets poorer even if the requested simulation jobs take only several seconds for each. Thus, for a better response time, web application is interfaced with a cluster and the simulation work is sent to the cluster for execution. This enables VENUS cyber infrastructure to serve more users simultaneously with better response time.

The original version of the VENUS-accessing cyber infrastructure also lacks tools for animating a chemical dynamics simulation, which is very important for understanding the complex atomic motions of the simulation. In a chemical dynamics simulation, the positions (i.e. the Cartesian coordinates) of all the atoms of the molecular system are calculated for a sequence of time steps by some numerical integration. Such a sequence of atom positions as a function of time is a classical trajectory that gives a detailed "picture" of the complex, often correlated motions of the atoms. And animation of the trajectories is an effective way for helping understand these atomic-level motions. One of the motivations behind the work presented in this paper is to provide an easy-to-use online tool for animating chemical dynamics simulations. Steps involved in a chemical dynamics simulation and animation can be summarized as below:

- A simulation with a selected simulation model is run through simulation software to get trajectories (simulation results).
- Scene description files are obtained from a selected trajectory of the simulation results.
- Scene description files are run through ray tracer program to get individual frames (images).
- Finally all images are combined to get an animated file.

Currently there are numerous tools for performing above steps and whole processes usually involve integration of two or more software packages, and users need to install all required software and hence need to have working knowledge of these software packages. To make our online animation tool of chemical dynamics simulations widely accessible to users, including animation software experts and non-experts, the animation tool is incorporated into the cyber infrastructure in such a way that multiple software packages for animation are integrated, and provide user a single access portal requiring only mouse-clicking for data selection or upload for animation, and the ensuing animation-creation processes are automated by the our system, thus hiding the complexity of creating an animation for non-expert users and freeing users from the worries to install in their computers and also to obtain working knowledge of animation software.

In addition to ease-of-use, we also optimized the animation tool to achieve high processing speed by choosing only a calculated subset of simulation data from the entire set of data of a selected trajectory, where the subset of data is calculated based on our empirical rules to minimize the amount of data for animation while also attain a good visual quality of the animation. By using only the minimum amount data for adequately good visual quality, we not only achieve high processing speed for the creation of animation file, but also reduce data transmission time over the internet, an essential factor in the quality of service for any online animation tools since data transmission over any network — internet, grid network, or inter-processor connection network — is far slower per unit amount of data than data processing speed inside a computer.

To provide the flexibility to serve different users with different needs, the cyber infrastructure is designed to have three access modes for users: 1. users perform dynamics simulation and animation on our computing resources; 2. users run the dynamics simulations on our computing resources and download simulation results onto their machine which they can perform their own post-simulation analyses like animation if the users have such software installed on their machines; 3. users run dynamics simulations on their own machines and use our online animation tool to run the animation.

By combining hardware and software computing resources at two locations under different administrative controls, this cyber infrastructure constitutes a grid environment providing physically and administratively distributed functionalities through a single, easy-to-use online portal for chemical dynamics simulations, and post-simulation analyses including animation of calculated trajectories. While there exist many chemical dynamics simulation software packages nowadays, to the best of our knowledge, our cyber infrastructure is the only online portal to grid environment for chemical dynamics simulations and its animation-included post-simulation analyses. And this infrastructure is designed to achieve both ease-of-use and high performance in terms of utilizing parallel computing power for simulations and minimizing data amount for animation and file transferring.

The remaining of the paper is organized as follows. Section 2 gives the overviews of the cyber infrastructure, Section 3 contains the description of the interface to the cluster for simulation execution for the web server machine, Section 4 is the description of the integration and optimization of animation software, and the conclusion is given in Section 5.

2. THE STRUCTURAL OVERVIEWS OF THE CYBER INFRASTRUCTURE

This online tool has a three tiered architecture. In the Figure 1, which shows the physical architecture, Client is a web browser running in any platform, and Web Server is responsible for handling requests that come from Client. Application that includes Servlet, JSP pages, and Java class files resides in Web Server. Cluster is for running simulation model.

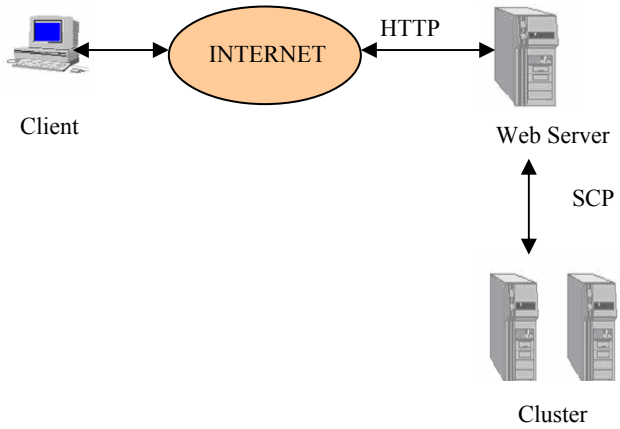

Figure 1: Physical Architecture

This online tool has been developed using JavaServer Pages (JSP) and Java Servlet technology. Figure 2 below shows the Software Structure. Only authenticated users will be able to access the system. Login module provides functionality for user registration and authentication. Users can choose between two programs VENUS96C, VENUS05. For each of these program types, there are three ways to provide input. Users can use already available simulation model as it is; they can modify the already available model; and they can upload their simulation model or trajectory file. Model/Program Type Selection Module provides functionality for choosing the program and input method type. Model Modify Module provides functionality for modifying the simulation model. Existing Model Module provides functionality for selecting the simulation model. File Upload Module allows users to upload the simulation model and trajectory file. Execute Module handles the execution of simulation model in cluster. Animation Setup Module allows users to change different parameters of animation like coloring method, the drawing method used for coloring atoms, etc. Animation Maker Module provides functionality for generating an animated file. Animation Viewer Module allows users to view and download the animated file.

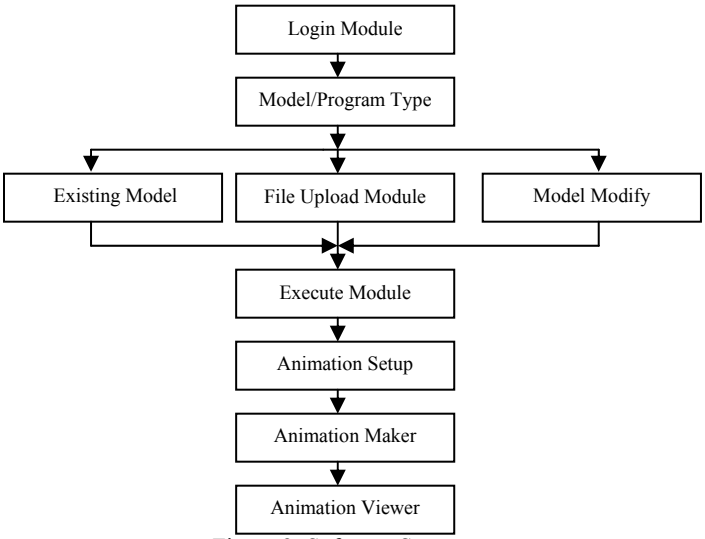

Figure 2: Software Structure

3. CLUSTER INTERFACE

When users connect to a web server, a session is created. Object "ClusterSession," which implements "HttpSessionListener" interface, gets notified when a session is created. Object "ClusterSession" has two methods: "sessionCreated" and "sessionDestroyed".

Method "sessionCreated" is called at the startup of a session, and method "sessionDestroyed" is called at the end of a session. Method "sessionCreated" creates object "ClusterExecutable", calls its "connect" method, and saves this object as a session object. "Connect" method of object "ClusterExecutable" connects to the cluster and creates a connection session with the cluster. Other methods of "clusterexeectuble" object include "execute," "put," and "get". The "execute" executes the command in cluster; "put" copies files from web server to cluster; and "get" copies files from cluster to web server.

Method "sessionDestroyed" calls "closesession" and "closeconncetion" method of "clusterexecutable" object, which closes the session and connection with the cluster. Figure 3 below shows Cluster Connection Sequence Diagram.

Existing Model Module, File upload Module, and Model Modify produce a simulation model and transfer it to the cluster for execution. In the execution of simulation model, two objects are involved, viz. "SimulationScmaker" and "SimulationScexecuter." Object "SimulationScmaker" creates a script file called "cluster_venus.sh" and transfers it to the cluster. For transferring the script file, Object "SimulationScmaker retrieves object "ClusterExecutable" from the session and calls its "put" with the script file as parameter. Script file "cluster_venus.sh" has a command to execute the simulation model.

After script is created, "SimulationScexecuter" object is created. Object "SimulationScexecuter" is responsible for executing the simulation model in the cluster. Object "SimulationScexecuter" retrieves object "ClusterExecutable" from the session and call its "execute" method with the following as parameter:
<qsub "Path of cluster_venus.sh"/cluster_venus.sh>.

The above command submits cluster_venus.sh script to the cluster queue with the help of qsub utility. After "cluster_venus.sh" gets executed, it makes "end.sh" file. When "SimulationScexecuter" object encounters "end.sh," it then copies output file to web server by calling "get" method. After the execution of model is finished, the user is forwarded to "Final.jsp" JSP page, which has a link to download the output file. Figure 4 below shows Execution Sequence Diagram.

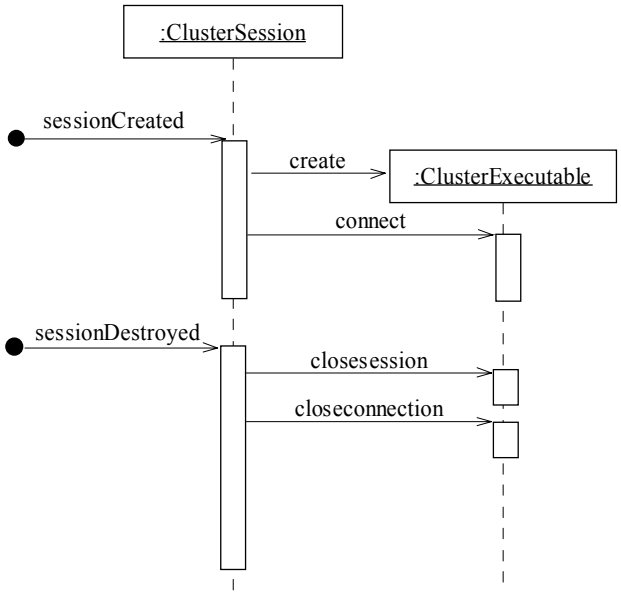

Figure 3: Cluster Connection Sequence Diagram.

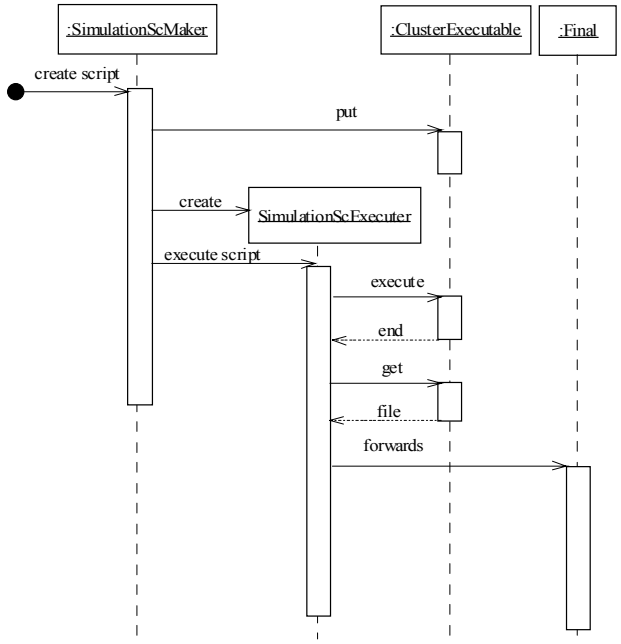

Figure 4: Execution Sequence Diagram.

4. ANIMATION PROCESS

Figure 5 below shows the steps involve in generating animation from simulation result.

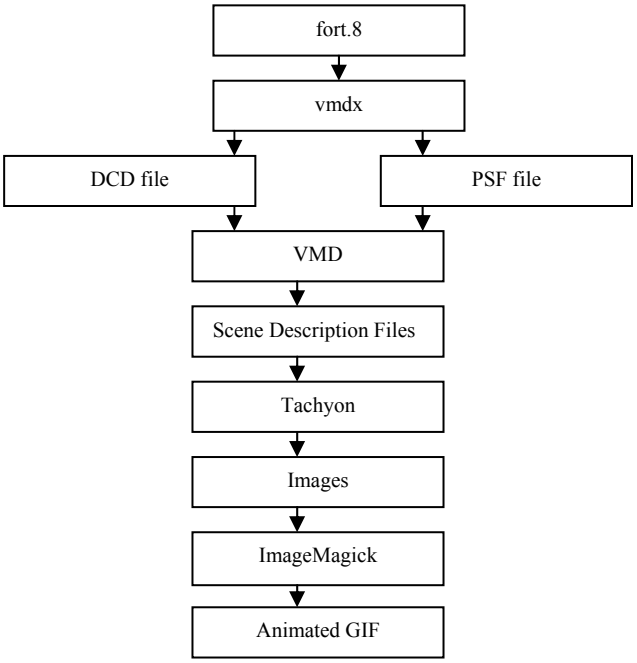

Figure 5: Animation Process

The "fort.8" file contains information of atoms like atom name, atomic number, and coordinates of each trajectory calculated during execution of simulation model. The "vmdx" is a C program, which converts "fort.8" file into DCD and PSF files. The PSF file, also called protein structure file, contains molecular specific information such as atoms, bonds, angles, dihedrals, etc. DCD is a binary file that contains coordinates of trajectory. VMD [3] takes DCD and PSF file as input and generates scene description files. Those scene description files are fed through Tachyon [4] ray tracer, which converts them into images. Finally, ImageMagick [5] converts all these images into an animated GIF file.

This application automates all these processes of generating an animation. Modules described below are involved in animation processes.

4.1. Animation Setup Module

Not every simulation model is valid for animation. When executed, the valid simulation model generates "fort.8" file. Object "FortfileChecker" performs checking of fort.8 file. If file is not found, the user is forwarded to "AnimationError.jsp" page, which displays an error. If the file is found, the user is forwarded to "AnimationSetup.jsp" page. This JSP page has an option for setting different parameters of animation. Trajectory for which animation is generated can be specified. Both the coloring method used to color atoms and the drawing method used to draw atoms can be changed. Additionally, the orientation of atoms can be changed by specifying degree in x, y and z axes. Delay between frames i.e. the delay between consecutive images in an animated file can be set. This parameter is important when the animation has a few frames. By adding delay between frames, the user can slow down the frames transition, thus all the details of animated file will be visible. Figure 6 below shows Animation Setup Sequence Diagram.

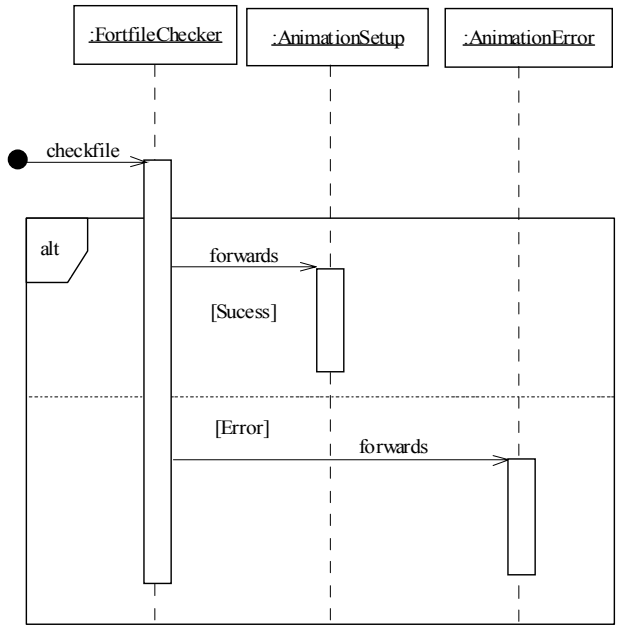

Figure 6: Animation Setup Sequence Diagram

4.2. Animation Maker Module

After animation parameters has been set on "AnimationSetup.jsp," the users clicks "submit" button. It calls "AnimationMaker" servlet. This servlet is responsible for executing the animation script, which generates the animated file. Object "AnimationMaker" creates object "AnimationscriptMaker." This object creates six script files, viz. "vmdx.sh," "vmd.tcl," "vmd_script.sh," "tachyon.tcl," "convert.sh," and "delete.sh."

The script file "vmdx.sh" converts "fort.8" into PSF and DCD files. DCD file contains the coordinates of trajectory, as specified by users in "AnimationSetup.jsp" page.

The script file "vmd.tcl" first loads DCD and PSF files; then, it sets parameters as users choose in the "AnimationSetup.jsp" page. It, then, calls procedure "makemovie." The procedure "makemovie" calls "buildmovie" procedure of VMD, which is responsible for generating scene file descriptions.

The script file "vmd_script.sh" runs VMD with input file as "vmd.tcl". After execution of this script, VMD generates scene description files. The number of scene description files depends on number of frames in DCD file, which in turns depends on NS and NIP parameters of the simulation model.

The script file "tachyon.tcl" loops through each frame and calls Tachyon executable with input as scene description file for that frame. The Tachyon generates .rgb files. To speed up the process of ray tracing, mediumshade is used, and antialiasing is turned off.

The script file "convert.sh" executes ImageMagick convert utility, which adds delay as specified by the users and combines all the images to create animated GIF file.

The fifth script file "delete.sh" deletes the scene description files and images generated during animation processes. After necessary scripts have been created, "AnimationMaker" servlet executes these scripts. The order in which these scripts are executed is "vmdx.sh", "vmd_script.sh," "tachyon.tcl," "convert.sh," and "delete.sh."

Page "AnimationSetup.jsp" also informs users about progress of animation processes. It informs users about completed stage(s) and stage(s) in progress. In an animation process, there are four stages: psf/DCD file creation, scene files creation, images creation and GIF file creation. The time it takes to execute the "vmdx.sh" script file, corresponds to psf/DCD file creation stage. Similarly, execution time of "vmd.tcl" and "vmd_script.tcl" corresponds to scene files creation stage. The execution time of the "tachyon.tcl" corresponds to images creation stage, and the execution time of the "convert.sh" corresponds to GIF file creation stage. This progress mechanism has been created with the help of Dynamics Web Remoting (DWR) [6]. The servlet "AnimationMaker" creates "MovieStatusListener" object. As "AnimationMaker" servlet executes the script, it gives the name of script it is executing to "MovieStatusListener" object, which passes this information to "MovieStatusInfo" object. The page "AnimationSetup.jsp" gets the "MovieStatusInfo" object by calling the method "getMovieInfo" of "MovieStatusMonitor" object. Then, it extracts the name of the script being executed from "MovieStatusInfo" object, and the name is displayed in the page. Figure 7 below shows the Animation Maker Module Sequence Diagram.

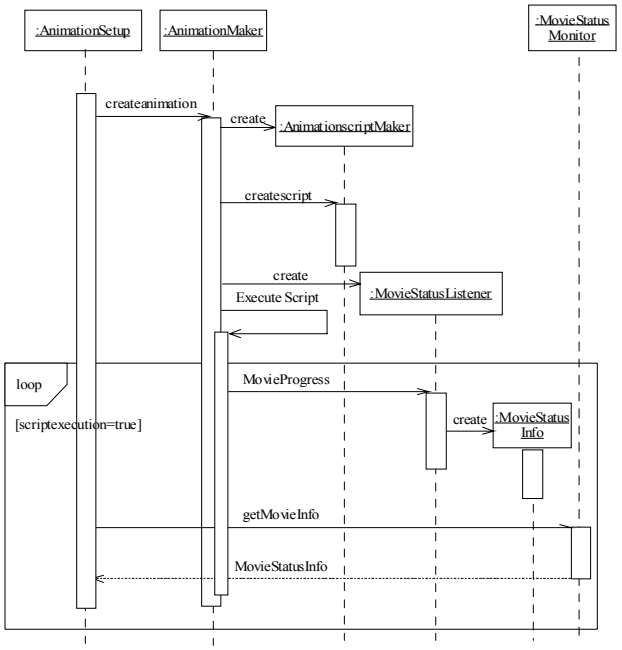

Figure 7: Animation Maker Sequence Diagram

4.3. The Optimization

The maximal number of frames in an animated file depends on NS (total number of integration steps per trajectory) and NIP (number of steps between intermediate printouts) parameters of simulation model. For example, for NS=4000 and NIP=20, we get 200 intermediate printouts of coordinates in "fort.8" file. Intermediate printouts of coordinates in "fort.8" file correspond to frames in an animated file. Thus, in this case, we have 200 frames. Greater number of frames generates a larger animated file, and it requires longer processing time and increases the communication time to transfer the animated file from a server to a client. To minimize communication overhead and to reduce processing time for animation file generation, the number of frames is optimized. The optimization is carried out in such a way that the number of image frames is minimized and yet the visual quality of the animation is not compromised.

In an animation of a trajectory, vibration of an atom is the fastest motion. From set of frames, of trajectory, if we pick frames so that the time between consecutive frames equals to at least half of the time period of the atom having the highest frequency, and generate an animation with such number of frames, this animation will capture the motion of the atom having the highest frequency. Since this animation includes the fastest motion, it will also include all other motion of atoms, for that trajectory. However, the number of frames required for this kind of result is large. Therefore, although the visual quality is very good, processing time is very high. Usually, the motions that interest chemistry researchers are not the highest frequency. If we generate animation with amount of frames, which is average of, viz, number of frames calculated by setting time between consecutive frames to time period of highest frequency, and that of lowest frequency, we will be able to reduce the number of frames and animation will still looks good.

To verify above statement, we did experiment on $F^- + CH_3OOH$ system. For this system, highest vibrational frequency is 4000 cm^{-1}, lowest frequency is 420 cm^{-1}, length of trajectory is 1750 fs and time step between consecutive frames is 1 fs. We calculate the time period using the formula as shown below.

$c=\nu\lambda$, c= light of speed, ν=vibrational frequency, λ=wave length
$\nu=c*1/\lambda$
$1/t=c*\lambda^{-1}$, vibrational frequency can be written as 1/t, t is time period and λ^{-1} is vibrational frequency in cm^{-1}
$t=1/(c*\lambda^{-1})$
$t=1/((2.99792458*10^{10}$ cm. sec.)$*\lambda^{-1})$, $c=2.99792458*10^{10}$ cm sec.
$t=(3.335640952*10^{-11}/\lambda^{-1})$ sec.
For our system, time period of highest frequency:
$t=3.335640952*10^{-11}/4000$
$t=8*10^{-15}$
t=8 fs.
Time period of lowest frequency:
$t=3.335640952*10^{-11}/420$
$t=80*10^{-15}$
t=80 fs.
Number of frames to be picked:
$1/2(1750/8+1750/80) \approx 100$

Time step between these 100 frames=1750/100≈18. Therefore, frames picked were in order of 1, 19, 37, and so on. We generate an animation with these100 frames and the animation was effective.

To implement this method described above, we need to have access to simulation output file, but since this online tool has also option to generate an animation from uploaded trajectory file, for this case, we cannot do such calculation. Therefore, we fix the number of frames based on above experiment and processing time, it takes to generate animation.

Figure 8, below shows the processing time versus no of frames, for different number of atoms. It shows processing time for 8, 729, 1290 and 2013 number of atoms with respect to 10, 50,100,200,300 and 400 frames. As seen in figure, for each frame, as number of atoms increases there is only slightly increase in processing time <1 min. On basis of this, we can conclude that, processing time for each frame is same, independent of number of atoms. In our web server for 10 frames, processing time is approximately 12 seconds, for 50 frames it is approximately 1 minutes, for 100 frames it is approximately 2 minutes, for 200 frames it is approximately 4 minutes, for 300 frames it is approximately 6 minutes, and for 400 frames it is approximately 7 minutes.

For F^- + CH_3OOH system, with 100 frames processing time is approximately 2 minutes and animation with this number of frames is effective. So whatever large no of frames fort.8 contains, only 100 frames will be picked and transferred to DCD file. Of course, this empirical criterion could be system-dependent, so people who prefer high animation quality may impose more than 100 frames. As future work, we are planning to add functionality that provides people with choices of high animation visual quality and medium visual quality.

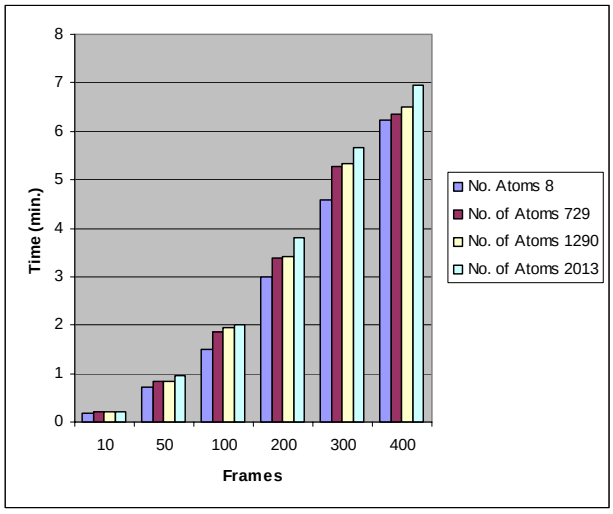

Figure 8: Animation Generation Time versus No. of Frames

5. CONCLUSION

In this paper, we presented new developments of a cyber infrastructure for high performance chemical dynamics simulations through an online portal to the chemical dynamics simulation software VENUS and post-simulation analyses tools including easy-to-use optimized animation creation software. This infrastructure integrates multiple software packages for simulation and post-simulation analyses into one web portal where all processes of software interactions are automated by the system, thus freeing users from having to obtain working knowledge of all necessary but complex software packages. By dispatching, different, computational tasks to physically and administratively distributed computing resources that are available to us, this cyber infrastructure attain high computation performance by utilizing all computing power of a computational grid it has access to. VENUS-web portal can be accessed using http://cdssim.chem.ttu.edu URL.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant Nos. CHE-0412677 and CHE-0615321, and the Robert A. Welch Foundation under Grant No. D-0005.

7. REFERENCES

[1] Hase, William L., Ronald J. Duchovic, Xiche Hu, Andrew Komornicki, Kieran F. Lim, Da-hong Lu, Gilles H. Peslherbe, Kandadai N. Swamy, Scott R. Vande Linde, Antonio Varandas, Jacobin Wang, and Ralph J. Wolf. "VENUS 96, A General Chemical Dynamics Computer Program." *Quantum Chemistry Program Exchange* 16, no. 671 (1996).

[2] Chawla, Navdeep. "EVALUATION OF TECHNOLOGIES FOR WEB-ENABLED CHEMICAL DYNAMICS SIMULATIONS." M.S. thesis, Texas Tech University, 2005.

[3] Humphrey, William, Andrew Dalke, and Klaus Schulten. "VMD - Visual Molecular Dynamics." *Journal of Molecular Graphics* 14 (1996): 33-38.

[4] *Tachyon*. Accessed 1 jan 2007. Available from http://jedi.ks.uiuc.edu/~johns/raytracer

[5] *ImageMagick*. Accessed 1 jan 2007. Available from http://www.imagemagick.org/script/index.php

[6] *DWR*. Accessed 1 jan 2007. Available from http://getahead.ltd.uk/dwr