# Performance Improvement and Workflow Development of Virtual Diffraction Calculations


Shawn P. Coleman
University of Arkansas
NANO 213B
Fayetteville, AR 72701
479-575-7205
sxc033@uark.edu

Sudhakar Pamidighantam
NCSA
1205 West Clark Street
Urbana, IL 61801
217-333-5831
spamidig@ncsa.uiuc.edu

Mark Van Moer
NCSA
2102D NCSA Bldg
Urbana, IL 61801
(217) 333-2268
mvanmoer@ncsa.illinois.edu

Yang Wang
Pittsburg Supercomputing Center
4400 Fifth Ave.
Pittsburgh, PA 15213
412-268-2795
ywg@psc.edu

Lars Koesterke
Texas Advanced Computing Center
10100 Burnet Rd.
Austin, TX 78758
512-232-5190
lars@tacc.utexas.edu

Douglas E. Spearot
University of Arkansas
NANO 213
Fayetteville, AR 72701
479-575-2040
dspearot@uark.edu



## ABSTRACT

Electron and x-ray diffraction are well-established experimental methods used to explore the atomic scale structure of materials. In this work, a computational algorithm is presented to produce electron and x-ray diffraction patterns directly from atomistic simulation data. This algorithm advances beyond previous virtual diffraction methods by utilizing an ultra high-resolution mesh of reciprocal space which eliminates the need for *a priori* knowledge of the material structure. This paper focuses on (1) algorithmic advances necessary to improve performance, memory efficiency and scalability of the virtual diffraction calculation, and (2) the integration of the diffraction algorithm into a workflow across heterogeneous computing hardware for the purposes of integrating simulations, virtual diffraction calculations and visualization of electron and x-ray diffraction patterns.



## Categories and Subject Descriptors

*F.2.1 Numerical Algorithms and Problems*

## General Terms

Algorithms, Performance

## Keywords

Materials Science, Diffraction, Workflow, Visualization.




*XSEDE '14*, July 13 - 18 2014, Atlanta, GA, USA



http://dx.doi.org/10.1145/2616498.2616552


## 1. INTRODUCTION

Diffraction is a common experimental method used to study the atomic scale structure of a material [1]. In x-ray diffraction, for example, a sample is exposed to a beam of monochromatic x-rays with a wavelength on the same order as the spacing between atomic planes in the sample. X-rays scatter upon interacting with the atoms in the sample and the constructive interference of the scattered x-rays is collected and analyzed to determine the crystal structure and lattice constants of the material. Similarly, a beam of electrons can be used within a transmission electron microscope (TEM) to produce a selected area electron diffraction (SAED) pattern. Both x-ray and electron diffraction can be used to study lattice distortion due to defects within crystalline solids (cf. [2,3]).

Calculations of diffraction patterns based on kinematic diffraction theory were first developed in the 1980s to characterize atomistic simulations of large-angle symmetric twist grain boundaries [4-6]. In kinematic models, the diffraction intensity, $I$, is computed for N atoms as the product of the structure factor, $F(\mathbf{K})$, with its complex conjugate, $F^*(\mathbf{K})$,

$$I(\mathbf{K}) = F^*(\mathbf{K})F(\mathbf{K}) \quad (1)$$

where

$$F(\mathbf{K}) = \sum_{j=1}^{N} f_j \exp\left(2\pi i\, \mathbf{K} \cdot \mathbf{r}_j\right) \quad . \quad (2)$$

Here, $\mathbf{K}$ is the location of the diffraction peak in reciprocal space, $\mathbf{r}_j$ is the position of the atom in real space, and $f_j$ is the atomic scattering factor. In [4-6], computational limitations inherent to the time restricted both the number of atoms in the calculation and the range of reciprocal space explored. To

improve computational efficiency, diffraction intensities were computed over a limited region of reciprocal space known to be important based on *a priori* knowledge of the grain boundary unit cell. Using these techniques, Bristowe and Sass [4] showed that small displacements of the atoms within the grain boundary unit cell can create identifiable changes to the diffraction patterns. This breakthrough motivated researchers [4-8] to incorporate virtual diffraction patterns in their subsequent studies to verify predicted grain boundary structures.

In the mid 2000s, a second method to compute diffraction patterns that utilized assumptions based on powder diffraction conditions became popular among researchers modeling nanocrystalline materials [9-14]. These researchers used formulations of the Debye scattering equation to compute diffraction intensity based on the interatomic distance between atoms, $r_{ij}$, via [1],

$$I(k) = \sum_{i=1}^{N}\sum_{j=1}^{N} f_i f_j \frac{\sin(2\pi k r_{ij})}{2\pi k r_{ij}}. \quad (3)$$

Here, $k = 2\sin(\theta)/\lambda$ represents a spherically averaged position in reciprocal space that is related to the diffraction angle, $\theta$, and monochromatic radiation of wavelength, $\lambda$. By spherically averaging the positions in reciprocal space, all orientations of the simulated crystals are mathematically represented mimicking the random distribution associated with powder diffraction. Using this relationship, researchers [9-14] constructed x-ray diffraction line profiles to investigate peak shift and peak broadening in order to extract data on the mean grain size and lattice strain in nanocrystalline models with different grain diameters.

The virtual diffraction algorithm in this work advances beyond these previous methods to create both SAED and x-ray diffraction line profiles using the same algorithm without any *a priori* knowledge of the crystal structure. The algorithm is sufficiently generic for all atomic species and is integrated into the Lammps atomistic simulation package [15] as a user-defined compute; it can also be implemented into other atomistic simulation packages. Following a concise discussion of the computational algorithm, this article presents advancements made to increase performance and scalability, and the integration of this algorithm into a workflow across heterogeneous computing hardware including visualization. A complete discussion of the virtual diffraction algorithm is provided in [16,17] along with application of this method to study grain boundary structure, nanocrystalline materials and interfaces between complex solids.

# 2. VIRTUAL DIFFRACTION METHOD

## 2.1 Diffraction Intensity

The diffraction algorithm generates a high-resolution, three-dimensional mesh of points filling a volume of reciprocal space constructed using the entire domain of the atomistic simulation cell. Each point on the reciprocal space mesh is associated with a reciprocal lattice vector $\mathbf{K}$ describing the deviation between the diffracted and incident wave vectors $\mathbf{k_D}$ and $\mathbf{k_I}$ [1],

$$\mathbf{K} = \mathbf{k_D} - \mathbf{k_I} \quad . \quad (4)$$

The mesh of reciprocal space points is built on a rectilinear grid with resolution defined by the user [16,17]. By constructing a high-resolution reciprocal space mesh, strong intensity peaks associated with the constructive interference between the incoming x-ray or electron beam and the crystal lattice are intuitively captured without *a priori* knowledge of the crystal structure of the material. The interplanar distances $d_{hkl}$ associated with each reflection can be computed at each reciprocal lattice point utilizing the geometric relationships described by Bragg's Law [1],

$$\frac{2\sin(\theta)}{\lambda} = \frac{1}{d_{hkl}} = |\mathbf{K}| \quad . \quad (5)$$

The diffraction intensity at each reciprocal lattice point is computed using the structure factor equation, Eq. (1), with the modifications described below to differentiate between electron and x-ray diffraction. In the diffraction algorithm, atomic scattering factors are explicitly computed for each atomic species to account for the reduction in diffracted intensity from an individual atom due to Compton scattering [18]. Specifically, at each diffraction angle, the atomic scattering factors are computed using analytical approximations parameterized for each atomic species. For electron diffraction, the analytical approximation of the atomic scattering factor is taken as the summation of five Gaussian functions of the form [19],

$$f_j\left(\frac{\sin\theta}{\lambda}\right) = \sum_{i}^{5} a_i \exp\left(-b_i \frac{\sin^2\theta}{\lambda^2}\right) \quad , \quad (6)$$

which have been parameterized for the majority of neutral elements by Peng et al. [20]. For x-ray diffraction, the analytical approximation of the atomic scattering factor is the summation of four Gaussian functions plus a constant of the form [21],

$$f_j\left(\frac{\sin\theta}{\lambda}\right) = \sum_{i}^{4} a_i \exp\left(-b_i \frac{\sin^2\theta}{\lambda^2}\right) + c \quad , \quad (7)$$

which have been parameterized by Fox et al. [22] for most atom species. In addition, to compute x-ray diffraction intensities, the Lorentz-polarization factor, $Lp(\theta)$, is applied to account for the relative distribution of the reciprocal lattice points and the change in scatter intensity when using non-polarized incident radiation. The Lorentz-polarization factor is computed via [18],

$$Lp(\theta) = \frac{1+\cos^2(2\theta)}{\cos(\theta)\sin^2(\theta)} \quad . \quad (8)$$

## 2.2 Diffraction Pattern Generation

X-ray diffraction 2θ line profiles are created by virtually rotating the Ewald sphere around the origin of reciprocal space to all possible orientations, mimicking powder diffraction conditions [18]. By making all diffraction orientations equally probable, every reciprocal space point will intersect the surface of the Ewald sphere. Line profiles are constructed by collecting reciprocal space points into bins corresponding to their scattering angle, using Eq. (5), and summing the intensity data within each bin. The scattering angle bin size is optimized through trials to reduce the noise within the line profile while maximizing peak features. For example, Figure 1 shows the calculated powder diffraction line profile for bulk α-alumina with comparison to experimental data [23].

Virtual SAED patterns are created by examining the region in reciprocal space intersecting the Ewald sphere of radius $1/\lambda$. For a particular zone axis, the Ewald sphere is centered at the tail of the associated incident wave vector and intersects the origin of reciprocal space. To construct the electron diffraction pattern, a hemispherical slice of the reciprocal space mesh lying on the surface of the Ewald sphere is isolated and viewed along the zone axis. Using VisIt [24], intensity values are interpolated between points on the reciprocal space mesh allowing the slice to take on no thickness while maintaining a continuous intensity field along the hemisphere. For example, Figure 2 shows the calculated SAED pattern for a (001) γ-alumina surface with comparison to experimental data [25]. To enhance features within the diffraction pattern, the sampled reciprocal space nodes are colored by intensity on a $\log_{10}$ scale. To achieve higher computational efficiency when creating SAED patterns for a user-specified zone axis, the diffraction algorithm can be augmented to limit the intensity calculation to only those reciprocal space mesh points that lie near the surface of the Ewald sphere.

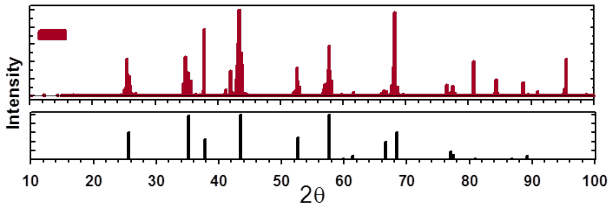


**Figure 1. Calculated powder diffraction pattern of α-alumina with comparison to experiment [23].**

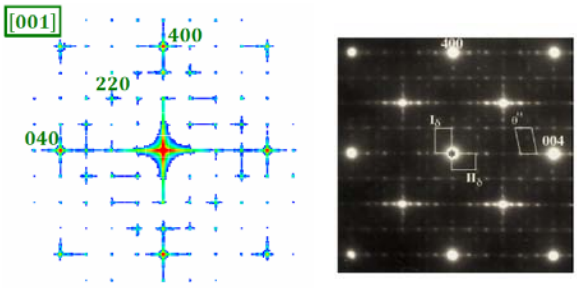


**Figure 2. Calculated selected area electron diffraction pattern for the γ-alumina surface with comparison to experiment [25].**

# 3. ALGORITHM SCALABILITY

## 3.1 Initial Scalability

The performance and scalability of the diffraction calculation was documented to provide benchmark data prior to the partnership with XSEDE Extended Collaborative User Services (ECSS). Benchmark diffraction computations were completed on the Texas Advance Computing Center (TACC) Stampede system, which is configured with compute nodes that host two 8-core Xeon E5-2680 processors and one Intel Xeon Phi SE10P coprocessor (MIC). The compute nodes are outfitted with 32 GB of memory (2GB/core) which is separate from the 8 GB of memory on the Xeon Phi coprocessor. The benchmark simulation contains 256,000 bulk Ni atoms and explores 9,006,316 reciprocal space nodes during a single x-ray diffraction computation of the static structure. The benchmark simulation is run within the Lammps atomistic simulation package. The output of the compute is an x-ray diffraction line profile for Ni with 2θ ranging from 10° to 90°. Timing data and memory usage are extracted directly from primary x-ray diffraction compute to avoid any overlaying functionalities within Lammps.

Initially, the code was parallelized via the native message passing interface (MPI) parallelization within the Lammps atomistic simulation package, which performs a spatial decomposition of the atoms in a simulation, illustrated in Fig. 3. The benchmark simulations were used to determine the speedup and efficiency of the diffraction calculation over 1,2,4,8, and 16 nodes. Results of the initial scalability tests are shown in Table 1. The speedup values are determined by comparing the absolute time for the computation using multiple nodes, compared to the computation time run using one 16-core node. Efficiency is computed by comparing the computation core-time to the reference core-time using one 16-core node. The results of the initial implementation of the code (prior to ECSS) show a 13.23 speedup with an efficiency reduction to 83% when scaled to 256 cores. The reduction in efficiency is primarily due to the finite number of atoms to parallelize over within the strong scaling study.

**Table 1. Initial scalability testing showing speedup, efficiency, and total memory usage.**

| Nodes | Cores | Speedup | Efficiency (%) | Memory (GB) |
|---|---|---|---|---|
| 1 | 16 | 1.00 | 100 | 9.67 |
| 2 | 32 | 1.91 | 96 | 19.34 |
| 4 | 64 | 3.65 | 91 | 38.67 |
| 8 | 128 | 6.70 | 84 | 77.35 |
| 16 | 256 | 13.23 | 83 | 154.70 |

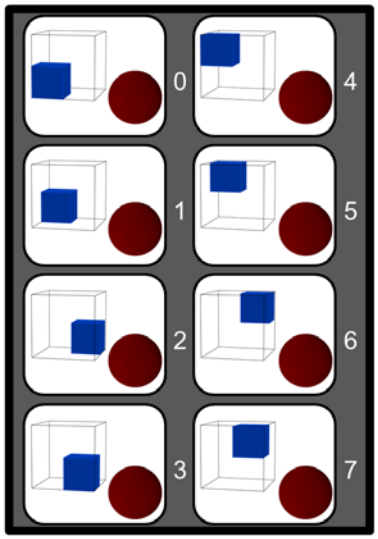


**Figure 3. Schematic of the MPI parallelization technique in the initial diffraction code. The blue cube represents the atomistic simulation while the red sphere represents the reciprocal space nodes sampled in the diffraction calculation.**

An identified weakness of the initial implementation of the virtual diffraction code is its memory footprint. The initial code requires that both the locations of the atoms and the locations of the reciprocal space nodes are kept in memory throughout the entire calculation. More importantly, a copy of all these locations must be accessible to each MPI process. Therefore, as the simulation size and resolution of reciprocal space increases, the memory required to complete the calculation dramatically increases. For example, the modest size benchmark simulation requires 9.7 GB of the available 32 GB of memory per node to compute a single x-ray diffraction line profile.

## 3.2 Scalability Improvements

Both speedup and efficiency of the diffraction calculation are improved through ECSS collaboration, resulting in a second generation of the code which has improved memory utilization and incorporated a second level of OpenMP parallelization shown schematically in Fig. 4.

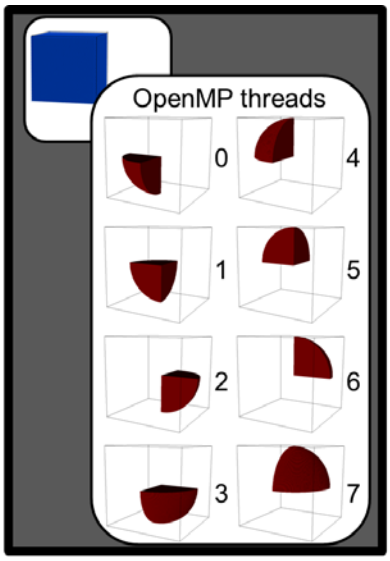


**Figure 4. Schematic of the OpenMP parallelization technique of the reciprocal space nodes sampled in the diffraction calculation.**

Further, a third generation of the diffraction code is developed via ECSS collaboration to take advantage of offloading tasks to the available MIC coprocessors on Stampede. Speedup values for each generation of the diffraction code are evaluated using the same benchmark simulation as in Section 3.1. The speedup value is computed from the absolute time of computation using the new code on multiple nodes as compared to the computation time run using the initial code on one 16-core node. Thus, speedup values are directly comparable across the different generations of the diffraction code. Differently, efficiency values are computed for each specific code generation and utilization of MPI/OpenMP. Results from the new scalability tests are shown in Table 2.

In the second generation of the virtual diffraction code, the MPI and OpenMP-based parallelization show speedup of 1.69 and 1.78, respectively, due to a reorganization of data structures and incorporation of more efficient calculations. MPI-based parallelization shows a similar efficiency drop (82% using 256 cores) as the initial code due to the strong scaling effects. To mitigate this effect, OpenMP threads are added to parallelize over the reciprocal lattice points. As shown in Table 2, the efficiency of the OpenMP parallelized second generation code is less affected by the finite number of atoms due to its second level of parallelism over the reciprocal lattice nodes (95% using 256 cores). The MIC enabled third generation code shows approximately 2x speedup from the second generation code by offloading a section of computation to the MIC. The reported values are taken when 90% of the reciprocal lattices nodes are offloaded to the MIC to be used in the solution of the structure factor equation. The remaining 10% of the reciprocal lattice nodes are utilized within concurrent computations on the CPU.

**Table 2. Scalability tests of the second and third generation diffraction code showing speedup compared to the original 16-core timing, efficiency relative to code generation and utilization of MPI/OpenMP, and total memory usage.**

| Nodes - MPI/OpenMP/MIC | Speedup | Efficiency (%) | CPU Memory (GB) |
|---|---|---|---|
| *Second Generation* | | | |
| 1 - 16/0/0 | 1.69 | 100% | 8.1 |
| 2 - 32/0/0 | 3.21 | 95% | 16.1 |
| 4 - 64/0/0 | 6.16 | 91% | 32.2 |
| 8 - 128/0/0 | 11.71 | 86% | 64.4 |
| 16 - 256/0/0 | 22.17 | 82% | 128.9 |
| 1 - 1/16/0 | 1.78 | 100% | 0.51 |
| 2 - 2/16/0 | 3.56 | 100% | 1.01 |
| 4 - 4/16/0 | 7.12 | 100% | 2.02 |
| 8 - 8/16/0 | 14.16 | 99% | 4.05 |
| 16 - 16/16/0 | 26.94 | 95% | 8.10 |
| *Third Generation (MIC – Enabled)* | | | |
| 1 - 1/16/240 | 4.61 | 100% | 0.51 |
| 2 - 2/16/240 | 9.19 | 100% | 1.01 |
| 4 - 4/16/240 | 17.43 | 95% | 2.02 |
| 8 - 8/16/240 | 34.65 | 94% | 4.05 |
| 16 - 16/16/240 | 60.93 | 83% | 8.10 |
| 1 - 16/0/240 | 4.37 | 100% | 8.1 |
| 2 - 32/0/240 | 8.38 | 96% | 16.1 |
| 4 - 64/0/240 | 15.94 | 91% | 32.2 |
| 8 - 128/0/240 | 29.51 | 84% | 64.4 |
| 16 - 256/0/240 | 49.32 | 70% | 128.9 |

# 4. WORKFLOW IMPLEMENTATION AND VISUALIZATION

## 4.1 SEAGrid Lammps_DS Workflow

The coupled execution of Lammps molecular dynamics followed by the simulation of X-ray diffraction or selected area electron diffraction calculations of the appropriately averaged system of interest and consequent visualization of the diffraction patterns required setting up of a workflow. The need for a workflow is particularly significant when multiple platforms for computing and visualization are to be used to address the large memory

VSMP implementation of the visualization software required in the final step. The workflow problem is tackled by using an existing XSEDE SEAGrid science gateway, that supported the Lammps application and the corresponding tools to achieve the remote job submission, but in this case using a remote workflow submission and managing the data such that a single jobID handle can provide all the data at the end of the multi-resource computations. Additionally, a script-based high-throughput submission of multiple jobs is implemented in SEAGrid client, as shown in Fig. 5. This uses an XML script with tags to specify job requirements and input files as needed and the system is equipped to execute each of the jobs specified as independent jobs. This is particularly useful for parameter sweep type runs and may benefit other communities that use the SEAGrid gateway as well.

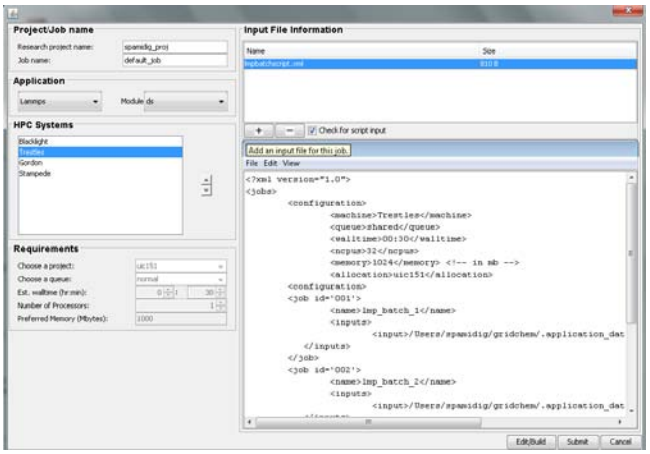


**Figure 5. Job set up panel in DESSERT client with multi-job submission script.**

The workflow implemented consists of coupled execution of three main tasks: Lammps_MD followed by Lammps_XRD, Lammps_SAED and a set of parallel VisIt executions as depicted in the schematic in the Figure 6. Initial implementations of the workflow used the same compute resource or a closely coupled resource at the same host site (such as Stampede and Ranch). A complete implementation involves execution of initial Lammps_MD at TACC's Stampede system and the latter two tasks in SDSC's Gordon system. The large memory available for the Gordon system was critical for the second stage of the workflow and it is useful in distributing the computing and visualization tasks. This also facilitates interactive access to the VisIt service deployed on Gordon in the future.

The workflow execution required additional inputs to drive the latter tasks and the original Lammps input was modified with a remark line for this purpose. The initial Lammps execution ignores this line in the input while the same is parsed and the queue instructions are set using this data for the subsequent steps. SEAGrid uses GSISSH based execution of a local script for the execution of a job task at a remote HPC site (for details of SEAGrid operation please refer to [26] and [27]). The Lammps specific portion of the script is modified to include the execution of the workflow transparently during this ECSS project. SEAGrid supported publicly released Lammps already, but a special module named DS (Doug Spearot's version) is implemented and currently restricted to the developer group to accommodate the special workflow version. The initial job is launched with the modified input on the Stampede system as a standard SEAGrid job. The Stampede job in turn prepares the inputs and provides the data required for the Lammps_XRD (and Lammps_SAED) and the VisIt components for the second stage. The script verifies the normal termination of the first stage, moves all the data and inputs for the subsequent jobs to the SDSC Gordon system and launches the jobs in Gordon. The path hierarchy is consistently managed and all the results are archived at the end of each stage into the mass storage device in SEAGrid organization. The entire workflow is tracked as a single job and all the corresponding result files can be retrieved based on the job handle through MyCCG job monitoring function in the client. The files retrieved can be further processed using the post processing tools available in the GridChem client. Though SEAGrid integrated the Apache Airavata based XBaya workflow execution system [28] which is described in detail in [27] for Paramchem project, this explicit implementation is chosen as the Gordon system did not support the Gram based orchestration of workflow available in XBaya. Currently, we are exploring the Unicore [29] based orchestration of tasks on Gordon system through Airavata/Xbaya system.

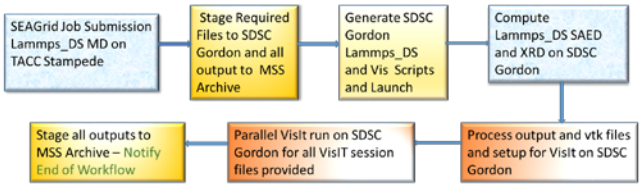


**Figure 6. Schematic workflow depicting the tasks orchestrated from GridChem client of SEAGrid Science Gateway.**

## 4.2 Visualization

Visualization uses the VisIt program and utilizes either a session-less protocol or a set of session files supplied along with the Lammps input by the user. The session files are staged to the appropriate computer resource and used in the final task. A rendering python script drives the VisIt computation in a parallel batch job for each of the vtk files generated during the processing of the diffraction patterns that are provided as Lammps dump files in the Lammps_XRD compute step. The VisIt runs result in the images outputted in .png format. The visualization of the images from the simulation can be launched from the GridChem client automatically and an example set for an alumina surface is shown

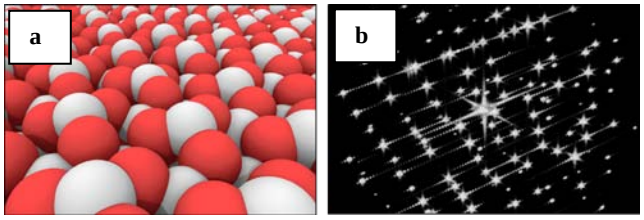


**Figure 7. Visualization of the (a) Van der Waals model of an α-alumina surface using Ovito and (b) the 3D reciprocal space map of the calculated electron diffraction pattern of an α-alumina surface. Aluminum atoms are shaded white and oxygen atoms are shaded red in (a).**

in Fig. 7. Alternatively if the visualization programs are locally installed, such as Ovito used to generate Fig. 7(a), appropriate files can be exported to these applications from the GridChem post processing tool.

## 5. SUMMARY

In this work, a computational algorithm is presented to produce electron and x-ray diffraction patterns directly from atomistic simulation data. First, through XSEDE ECSS support, the scalability and performance of the virtual diffraction algorithm has been considerably improved. The diffraction algorithm parallelization has been enhanced to support offloading to the MIC coprocessor on Stampede. Second, through XSEDE ECSS support, the virtual diffraction algorithm has been integrated into a workflow platform that allows for job submission across heterogeneous computing hardware at different sites within the XSEDE network. Specifically, atomistic simulations using LAMMPS, diffraction calculations using LAMMPS and visualization using VisIt are coupled in the workflow using the SEAGrid Science gateway. Ultimately, the diffraction algorithm provides a novel bridge between experiments and computation and will be of significant use to both communities to study the atomic-level structure of materials.

## 6. ACKNOWLEDGMENTS

The authors acknowledge support of the National Science Foundation under grant #0954505. Additional support is provided by the 21st Century Professorship in Mechanical Engineering at the University of Arkansas. This work utilized the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant #OCI-1053575. Consultant support was made possible through the XSEDE Extended Collaborative Support Services (ECSS) program. Beneficial discussions concerning FFTW implementation with XSEDE Campus Champion Fellow Luis Cueva-Parra of Auburn University Montgomery are appreciated.

## 7. REFERENCES

[1] Willams, D.B., Carter, C.B. 2009 *Transmission Electron Microscopy: A Textbook for Materials Science.* Springer, New York, NY.

[2] Ungar, T. and Borbely, A. 1996. The effect of dislocation contrast on x-ray line broadening: A new approach to line profile analysis. *Appl. Phys. Lett.* 69, 3173-3175.

[3] Meyer, K.E., Felcher, G.P., Sinha, S.K., Schuller, I.K. 1981. Models of diffraction from layered ultrathin coherent structures. *J. Appl. Phys.* 52, 6608-6610.

[4] Bristowe, P.D., Sass, S.L. 1980. The atomic-structure of a large-angle [001] twist boundary in gold determined by a join computer modeling and x-ray-diffraction study. *Acta Metall.* 28, 575-588.

[5] Budai, J., Bristowe, P.D., and Sass, S.L. 1983. The projected atomic-structure of a large-angle [001] sigma=5 twist boundary in gold: Diffraction analysis and theoretical predictions. *Acta Metall.* 31, 699-712.

[6] Bristowe, P.D. and Balluffi, R.W. 1984. Effect of secondary relaxations on diffraction from high-sigma [001] twist boundaries. *Surf. Sci.* 144, 14-27.

[7] Oh, Y. and Vitek, V. 1986. Structural multiplicity of sigma=5(001) twist boundaries and interpretation of x-ray-diffraction from these boundaries. *Acta Metall.* 34, 1941-1953.

[8] Fitzsimmons, M.R. and Sass, S.L. 1988. Quantitative x-ray-diffraction study of the atomic-structure of the sigma=5 [001] twist boundary in gold. *Acta Metall.* 36, 3103-3122.

[9] Brandstetter, S., Derlet, P.M., Van Petegem, S., and Van Swygenhoven, H. 2008. Williamson-Hall anisotropy in nanocrystalline metals: X-ray diffraction experiments and atomistic simulations. *Acta Mater.* 56, 165-176.

[10] Stukowski, A., Markmann, J., Weissmüller, J. and Albe, K. 2009. Atomistic origin of microstrain broadening in diffraction data of nanocrystalline solids. *Acta Mater.* 57, 1648-1654.

[11] Markmann, J., Yamakov, V. and Weissmüller, J. 2008. Validating grain size analysis from X-ray line broadening: A virtual experiment. *Scr. Mater.* 59, 15-18.

[12] Markmann, J., Bachurin, D., Shao, L., Gumbsch, P. and Weissmüller, J. 2010. Microstrain in nanocrystalline solids under load by virtual diffraction. *Europhysics Lett.* 89, 66002.

[13] Derlet, P.M., Van Petegem, S. and Van Swygenhoven, H. 2005. Calculation of x-ray spectra for nanocrystalline materials. Phys. Rev. B 71, 024114.

[14] Van Swygenhoven, H., Budrovie, Z., Derlet, P.M., Froseth, A.G., Van Petegem, S. 2005. In situ diffraction profile analysis during tensile deformation motivated by molecular dynamics. *Mater. Sci. Eng. A* 400, 329-333.

[15] LAMMPS 2014. http://lammps.sandia.gov.

[16] Coleman, S.P., Spearot, D.E., Capolungo, L. 2013. Virtual diffraction analysis of Ni [010] symmetric tilt grain boundaries. *Model. and Simul. in Mater. Sci. and Engin.* 21, 055020.

[17] Coleman, S.P., Sichani, M.M., Spearot, D.E. 2014. A computational algorithm to produce virtual x-ray and electron diffraction patterns from atomistic simulations. JOM 66, 408-416.

[18] B. E. Warren 1990. *X-Ray Diffraction,* Dover Publications, New York, NY.

[19] Colliex, C., Cowley, J.M., Dudarev, S.L., Fink, M., Gjønnes, K., Hilderbrandt, R., Howie, A., Lynch, D.F., Peng, L.-M., Ren, G., Ross, A.W., Smith Jr., V.H., Spence, J.C.H., Steeds, J., Wang, J., Whelan, M.J. and Zvyagin, B.B. 2004. Scattering factors for the diffraction of electrons by crystalline solids. *International Tables for Crystallography, Volume C: Mathematical, Physical and Chemical Tables, 3rd ed.*, edited by: Prince, E., Kluwer Academic Publishers, Norwell, MA, 259–429.

[20] Peng, L.-M., Ren, G., Dudarev, S.L. and Whelan, M.J. 1996. Robust parameterization of elastic and absorptive electron atomic scattering factors. *Acta Crystallogr. Sect. A* 52, 257-276.

[21] Brown, P.J., Fox, A.G., Maslen, E.N., O'Keefe, M.A. and Willis, B.T.M. 2004. X-ray scattering. *International Tables for Crystallography, Volume C: Mathematical, Physical and*

*Chemical Tables, 3rd ed.,* edited by: Prince, E., Kluwer Academic Publishers, Norwell, MA, 554–595.

[22] Fox, A.G., O'Keefe, M.A. and Tabbernor, M.A. 1989. Relativistic Hartree-Fock x-ray and electron atomic scattering factors at high angles. *Acta Crystallogr. Sect. A* 45, 786-793.

[23] Ishizawa, N., Miyata, T., Minato, I., Marumo, F., Iwai, S. 1980. A structural investigation of alpha-$Al_2O_3$ at 2170 K. *Acta Crystallographica B* 36, 228-230.

[24] VisIt 2014. https://wci.llnl.gov/codes/visit/.

[25] Levin, I. and Brandon, D. 1998. Metastable alumina polymorphs: Crystal structures and transition sequences. *J. Am. Ceram. Soc.* 81, 1995–2012.

[26] Dooley, R., Milfeld, K., Guiang, C., Pamidighantam, S. and Allen, G. 2006. From proposal to production: Lessons learned developing the computational chemistry grid cyberinfrastructure. *J. Grid Comp.* 4, 195-208.

[27] Shen, N., Fan, Y. and Pamidighantam, S. 2014. E-science infrastructures for molecular modeling and parameterization. *J. Comp. Sci.* in press.

[28] Marru, S., Herath, C., Tangchaisin, P., Pierce, M., Mattmann, C., Singh, R., Gunarathne, T., Chinthaka, E., Gardler, R., Slominski, A., Douma, A., Perera, S., Gunathilake, L. and Weerawarana, S. 2011. Apache airavata: a framework for distributed applications and computational workflows. *Proceedings of the 2011 ACM Workshop on Gateway Computing Environments*, 21-28.

[29] http://www.unicore.eu.