

**Atomistic Simulations of Chemomechanical Processes in  
Nanomaterials Under Extreme Environments**

By

Hansohl Cho

B.S., Seoul National University, 2004

Submitted to the Department of Mechanical Engineering  
In Partial Fulfillment of the Requirements for the Degree of  
Master of Science in Mechanical Engineering

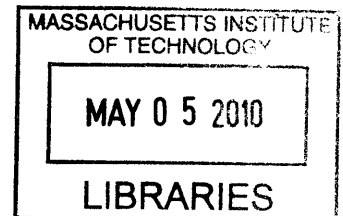
At the

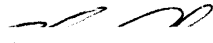
Massachusetts Institute of Technology

September 2009

© 2009 Massachusetts Institute of Technology  
All rights reserved

**ARCHIVES**

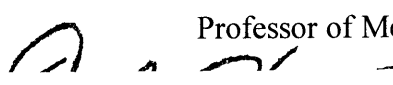


Signature of Author.....

Hansohl Cho  
Department of Mechanical Engineering  
Aug 19, 2009

Certified by.....

Thomas Lord Associate Professor of Material Science and Engineering  
Thesis Supervisor  
Krystyn J. Van Vliet

Certified by.....

David M. Parks  
Professor of Mechanical Engineering  
Thesis Reader

Accepted by.....

David E. Hardt  
Ralph E. and Eloise F. Cross Professor of Mechanical Engineering  
Chairman, Graduate Thesis Committee

# **Atomistic Simulations of Chemomechanical Processes in Nanomaterials Under Extreme Environments**

Submitted to the Department of Mechanical Engineering on August 19, 2009  
In Partial Fulfillment of the Requirements for the Degree of  
Master of Science in Mechanical Engineering

## **ABSTRACT**

The complex chemomechanical behavior of nanomaterials under extreme thermal and mechanical environments is of interest for a range of basic science and defense applications. By the limitation of experimental approaches for objects of nanometer, novel computational methods have been developed to investigate such phenomena in nanomaterials under extreme environments. In this thesis, novel continuum and atomistic mechanical modeling and simulations are implemented and constructed for the analysis of the chemomechanical behavior of the dissimilar nano-scale metals, Nickel and Aluminum under a variety of thermal and mechanical stimuli. These studies form the basis of preliminary research on the predictive design principles for reactive polymer nanocomposites.

Thesis Supervisor: Krystyn J. Van Vliet, Ph. D.

Title: Thomas Lord Associate Professor of Material Science and Engineering

Thesis Reader: David M. Parks, Ph. D.

Title: Professor of Mechanical Engineering

## ACKNOWLEDGEMENTS

As always, I would like to thank my Jesus for his own ways all through my life.

I would like to express my gratitude to my research advisor, Professor Krystyn J. Van Vliet for her support, guidance, and mentorship on my research and life, here. I also would like to thank my thesis reader, Professor David M. Parks.

Without great supports from my people listed here, I couldn't have finished my SM thesis and research.

Members in Laboratory for Material Chemomechanics: Mukul, Ilke, Meng, Karen, Emily, Adam, Irene, John, Ranjani, Sezen, Joan, and Sunyoung

Brothers in KGSAME

Brothers in KGSAME 2007: Yunseog, Seunghyuk, Yondae, Sungmin, Jongsub, Wonjoon, Yongjin, and Heonjoo

Members in Boston Mokyang Church: Pastor Choi and Mrs. Choi

My friends in Korea and US: 7-Up members (Kihyung, Jaewoong, Hyungjoon, Hyunwoo, Dongbok, and Sehoon), Jiwon, Minsir, and Junghoon

Dr. Aidan P. Thompson (Sandia National Laboratory), Dr. Steve J. Plimpton (Sandia National Laboratory)

Elder Kyoungghwa Lee in Albuquerque Church

My beloved family in Korea: Minjoe, Kyungsook, Sunhee, Younghwan, Aram, and Insook

Finally, I owe everything in my life to my girl, Hyunsook. I do love you.

Sincerely,  
HANSOHL

## TABLE of CONTENTS

<b>Chapter 1 Introduction.....</b>	<b>7</b>
1.1. Background and motivation	
1.2. Modeling and simulations	
1.3. Thesis scope and research objectives	
<b>Chapter 2 Preliminary study on self-sustained combustive reaction propagations in Nickel and Aluminum laminate foils: a continuum approach.....</b>	<b>11</b>
2.1. Background	
2.2. Mathematical modeling: the governing equations	
2.3. Numerical modeling: Finite difference equations	
2.4. Study for physical and chemical parameters, and issues on the parameters	
2.5. Numerical results with Dirichlet boundary conditions	
2.6. Numerical result with Neumann boundary conditions	
2.7. Effect of premixed thickness	
2.8. Effect of ambient temperature	
2.9. Comparison with experiments	
2.10. Conclusion	
<b>Chapter 3 Theoretical and computational background in molecular dynamics simulations .....</b>	<b>36</b>
3.1. Introduction to the molecular dynamics simulations	
3.2. Statistical mechanics fundamentals	
3.3. Calculations of physical properties in the molecular dynamics simulations	
<b>Chapter 4 Interatomic potentials: Embedded Atom Method (EAM) potentials for Ni and Al systems.....</b>	<b>48</b>

- 4.1. Embedded Atom Method potentials for Ni and Al systems
- 4.2. Potential functions: electron density, embedding energies, and pair interaction energies
- 4.3. Lattice, thermal and mechanical properties from EAM potentials

**Chapter 5 Equilibrium molecular dynamics simulations.....55**

- 5.1. Equilibration of Ni and Al nanoslabs
- 5.2. Equilibration of Ni and Al nanoparticles: the finite systems

**Chapter 6 Non-equilibrium molecular dynamics simulations of Ni and Al nanoslab impacts.....67**

- 6.1. Shock compression loading within periodic boundary conditions (PBC) and simulation condition setup
- 6.2. Evolutions of shock induced dynamic properties during the impact loading
- 6.3. Evolutions of kinetic and potential energies after the impact loading: long time scale phenomena
- 6.4. Conclusion

**Chapter 7 Non-equilibrium molecular dynamics simulations of Ni and Al nanosphere impacts.....87**

- 7.1. Shock loading within periodic boundary conditions (PBC): unexpected failure of the shock loading assumption
- 7.2. Shock loading within non periodic boundary conditions
- 7.3. Evolutions of shock induced dynamic properties during the impact loading
- 7.4 Dynamic properties during the long time scale simulation
- 7.5 Impact between nanoparticle having rough surfaces
- 7.6 Effect of nanoparticle sizes
- 7.7. Conclusion

<b>Chapter 8 Implementation of Reactive Force Field (ReaxFF) potential for future research.....</b>	<b>110</b>
8.1. Theoretical backgrounds in ReaxFF: fully reactive Carbon Hydrate systems	
8.2. Worked examples of RDX and TATB: comparisons with GRASP results	
8.3. Conclusion: Further research focus on the ReaxFF on Nickel, Aluminum, and Oxygen systems	
<b>Chapter 9 Summary and Conclusion.....</b>	<b>122</b>
9.1. Summary	
9.2 Future research plan	
<b>Appendix.....</b>	<b>125</b>
<b>References.....</b>	<b>142</b>

# Chapter 1

## Introduction

### 1.1. Background and motivation

Polymer nanocomposites, herein defined as a polymer matrix filled with inorganic fillers such as metal nanoparticles, are of interest for many basic science and defense applications. Nanocomposites exhibit chemical and physical properties as a function of the compositions and volume fractions of the constituents and imposed environmental conditions. Thus, many research groups have put effort to quantify and examine the different chemical and physical properties of nanocomposites under various demands. In addition, both enhanced thermomechanical performance has been predicted and analyzed in nanocomposites by experiments and computer simulations. Among potential polymer nanocomposites, here, we will focus on reactive polymer nanocomposites.

Specifically, potential nanoparticles for the reactive polymer nanocomposites can be energetic materials that can be thermally and mechanically mixed resulting in strong exothermic reactions. For example, Nickel and Aluminum nanoparticle (or Titanium and Aluminum) can be mixed by external stimuli such as thermal and mechanical input to form the lower energy state of the Nickel and Aluminum compounds. The ability to harness this potential within a polymer matrix could enable lightweight materials capable of localized heat generation; electronic energy storage and release-on-demand; and extensive energy dissipation upon mechanical insult. The extent and efficiency of such energy transfer is expected to depend in part on the physical length scales of the materials (e.g., macro-scale laminates vs. nanoscale particles) and time scales of deformation (e.g., quasi-static vs. shock loading).

The focus of the research on the reactive polymer nanocomposites is to define and investigate the potential capabilities of such composites through fundamental understanding of the chemomechanical reactions at the three chief material interfaces in such nanocomposites (metal, native oxide, and polymer) under extreme chemical and mechanical environments. These goals will be accomplished primarily through

novel computational modeling and simulation approaches, which will be ultimately verified and validated through collaboratively executed experiments. Many research groups are developing new structures and materials for mitigation of blast, ballistic, and localized low-velocity impact, as well as for energy storage and harvesting. Direct simulation and measurement of energy dissipation at the nanoscale affords new opportunities to rapidly optimize these new nano-architectures and materials at the development stage, and to understand the micro structural mechanisms and interface chemistry that afford optimized energy absorption and dissipation.

Finally, even though many properties of the nanocomposites have been investigated and verified under the equilibrium experimental conditions, those properties remain unknown in the nonequilibrium regime. Thus, it is of great practical interest to quantify the complex behaviors of nanomaterials under the non-equilibrium conditions such as extreme thermal and mechanical environments.

In this thesis, the potential energetic materials (Nickel and Aluminum) which would serve as fillers in the polymer matrix under extreme environments are investigated as a preliminary research on the reactive polymer nanocomposites for the various applications.

## **1.2. Modeling and simulations**

Macroscale thermal and mechanical behaviors of material systems have been well described with the classical continuum mechanics based computer simulations. However, by those continuum based approaches, detailed atomistic and molecular behaviors cannot be quantified in the simulations. In particular, in the regions having extreme gradients in the thermal and mechanical behaviors of the materials, continuum approaches cannot describe based on constitutive models at nanometer scales. Thus, the atomistic and molecular mechanical simulations have been adopted to investigate the physical and chemical behaviors of the materials at the nanoscale under extreme environments. However, it is not efficient and impossible to apply the atomistic and molecular level approaches to macroscale objects. For this limitation, coupling of the atomistic and molecular mechanics mechanical simulations coupled to continuum mechanics called multiscale mechanics has been investigated. For the purpose of realizing the multiscale mechanics, bridging the time and length scales is very important. Unfortunately, there exist many difficult problems related to the bridging methods which have not been solved. To construct appropriate



computational models for the polymer nanocomposites with the various time and length scales, it is essential to realize the multiscale models for the systems, especially at interfaces between polymers and metals. As preliminary research, in this thesis, we will focus on the atomistic details for the energetic metallic systems under extreme environments with atomistic computational modeling and simulation. This preliminary research on the atomistic approaches will support future research on the bridging method for polymer nanocomposites with the multiscale approaches, based on the continuum and atomistic mechanics.

### **1.3. Thesis scope and research objectives**

The objective of this research is to quantify the chemomechanical behavior of the potential nanofiller materials (Nickel and Aluminum) under extreme thermal and mechanical environments. The main research objectives and scope are summarized below:

A. Self-sustained intermixing reaction propagations initiated by direct thermal energy input via electric pulses in Nickel and Aluminum laminates have been modeled in the coupled partial differential equations for the reactive heat conduction and mass diffusion processes. By the semi-implicit finite difference method (FDM) based numerical computations for the PDE systems, the reaction front speed, self-sustainability, and reaction zone evolution and size have been predicted and compared to the current experimental data.

B. Interatomic potentials, based on the EAM (Embedded Atomic Method) potential for Nickel and Aluminum hybrid systems and the ReaxFF (Reactive Force Field) potential with the charge equilibration algorithm for C, H, O, N, and Si systems have been implemented and parallelized in LAMMPS. This public code has been used in all atomistic simulations in this research, and validated in terms of physical properties of the materials at equilibrium and computational issues such as parallelization and neighboring.

C. Equilibrium structures for semi-infinite nanoslabs and finite nanoparticles of Nickel and Aluminum have been constructed and predicted using various equilibrium molecular dynamics simulations, and validated in terms of physical properties under specific experimental conditions.

D. Non-equilibrium molecular dynamics (NEMD) simulations for the Nickel and Aluminum nanoslabs and nanoparticles subject to high speed impact (shock) loading have been performed to investigate the consequent intermixing reactions between the two materials initiated by the extreme mechanical stimuli.

## **Chapter 2**

# **Preliminary study on self-sustained combustive reaction propagations in Nickel and Aluminum laminate foils: a continuum approach**

### **2.1. Background**

It is known that self-sustained oscillatory combustion and the corresponding exothermic melting and alloying reactions occur in reactive composites. Such reactive energetic materials as Ni and Al could be mixed and reacted into a NiAl alloy, and consequently produce a large amount of formation energy by sufficient energy input.

One of the most preferred methods to initiate the reactions in Ni and Al systems is to directly input sufficient thermal energy via electric pulses over the effective adiabatic temperature for the Ni and Al mixing (1850K, [45]). By imposing thermal shock over the melting temperature of the two reactants, the reactions between Ni and Al systems are activated and propagate in a self-sustained manner.

In this chapter, evolutions of (1) atomic concentrations (composition) of the two reactants and (2) temperature of the composites with combustion are mathematically modeled. Two coupled partial differential equations are used to predict the mass and heat diffusion accompanied with the reaction with proper physical parameters such as atomic mass diffusivity, heat conductivity, material density, heat capacity and empirical formulae for the heat formation by the reactions. After constructing the proper governing equations, and initial conditions and boundary conditions, the finite difference methods for time and space are adopted to numerically solve the partial differential equations. Finally, the numerical results are compared with the experimental data in terms of reactivity, self-sustainability, reaction front propagation speed, and reaction zone sizes with respect to various conditions.

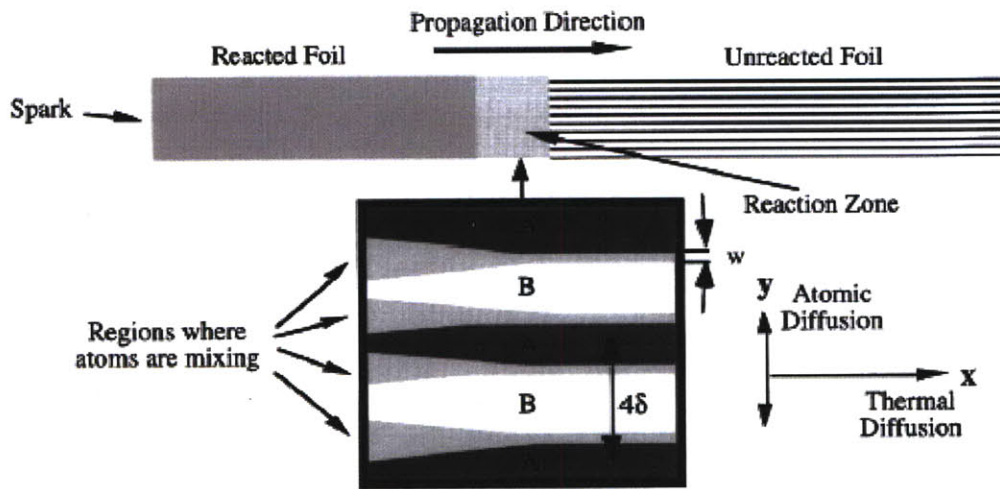


Figure 2-1 Schematic in thermally induced reactions in Ni and Al laminates [43].

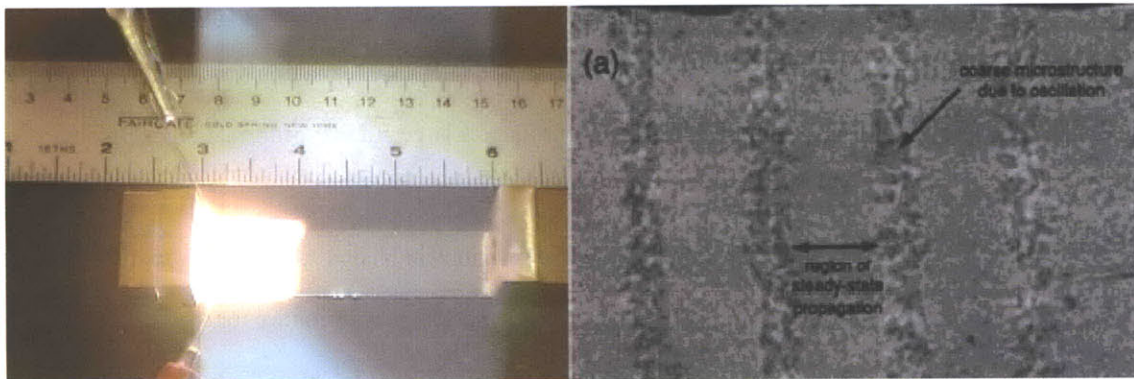


Figure 2-2 Experiment in thermally induced reactions in Ni and Al laminates (J. Trenkle et al., 2008, unpublished)

## 2.2. Mathematical modeling: the governing equations

In the laminated Ni and Al foils, the primitive phenomena during the reactions are heat diffusion with the additional heat source by the combustive reactions, and mass diffusion and intermixing between Ni and Al constituents. Those heat and mass diffusion could be modeled in the nonlinear, coupled parabolic type partial differential equations.

### 2.2.1. Atomic (mass) diffusion equations

The atomic mass diffusion with combustion could be described based on the mass conservation equation (2-1) as a balance equation and Fick's law (2-2) as a constitutive equation.

$$\frac{dC(x, y, t)}{dt} = \text{Div}(\vec{J}) \quad (2-1)$$

$$\vec{J} = D \cdot \nabla C \quad (2-2)$$

$C$  is the conserved Shvab-Zeldovich scalar variable for the mass concentration (composition) of Nickel and Aluminum. For the Ni and Al system,  $C$  is defined as +1 for pure Al, -1 for pure Ni, and 0 for NiAl, respectively. The atomic diffusivity,  $D$  is assumed to be independent of composition during the reactions and to follow an Arrhenius dependence on temperature given by

$$D = D_0 \exp\left(-\frac{E}{RT}\right) \quad (2-3)$$

Where  $D_0$  is the Arrhenius pre-exponent,  $E$  is the activation energy for Ni Al reaction, and  $R$  is the gas constant.  $D_0$  and  $E$  are determined using experimental data for Ni and Al system [43].

Finally, we obtain the first governing equation for the mass diffusion with the combustions as below.

$$\frac{dC(x, y, t)}{dt} = \frac{\partial C(x, y, t)}{\partial t} + \vec{V}(x, y, t) \circ \nabla C(x, y, t) = \nabla \circ [D(T(x, y, t)) \cdot \nabla C(x, y, t)]$$

$$\text{where, } \vec{V} = \frac{\vec{X}(x, y, t)}{dt} = 0$$

$$\frac{\partial C(x, y, t)}{\partial t} = \nabla \circ [D(T(x, y, t)) \cdot \nabla C(x, y, t)] \quad (2-4)$$

### 2.2.2. Heat diffusion equations

The energy transport via heat diffusion with the energy source by the reactions could be described based on the energy conservation equation (thermodynamic 1<sup>st</sup> law) as a balance equation (2-4) and Fourier's law (2-5) as a constitutive equation on heat diffusion.

$$\rho \cdot c_p \frac{dT(x, y, t)}{dt} = Div(\vec{q}) + \frac{\partial Q(C(x, y, t))}{\partial t} \quad (2-5)$$

$$\vec{q} = k \cdot \nabla T \quad (2-6)$$

In this equation,  $Q$  is the heat formation due to mixing of Ni and Al. It has an empirical form related to the mass concentration of Ni and Al. The empirical form of  $Q$  is quadratic to the mass concentration  $C$  based on experiment results.

$$Q = -\rho \cdot c_p (T_{f0} - T_0) \cdot C^2 \quad (2-7)$$

$$T_{f0} = T_0 + 1660K$$

Where,  $T_{f0}$  is the adiabatic temperature obtained from the heat of formation of Ni and Al and  $T_0$  is the ambient temperature [42].

Finally, we get the second governing equation for the heat conduction with the formation energy

$$\rho \cdot c_p \cdot \frac{\partial T(x, y, t)}{\partial t} = \nabla \circ (k \cdot \nabla T(x, y, t)) + \frac{\partial Q(C(x, y))}{\partial t} \quad (2-8)$$

The constructed governing equations (2-4) and (2-8) for mass and heat diffusion are two dimensional time dependent parabolic type partial differential equations, and temperature,  $T(x, y, t)$  and mass concentration,  $C(x, y, t)$  are strongly coupled via the atomic diffusivity  $D$  and formation heat  $Q$  in the two equations.

$$D = D(T(x, y, t) = D_0 \exp\left(-\frac{E}{RT(x, y, t)}\right) \quad (2-9)$$

$$Q = Q(C(x, y, t) = -\rho \cdot c_p (T_{f0} - T_0) \cdot C(x, y, t)^2$$

Also, in the heat equation (2-8), the thermal conductivity, material density, and heat capacity are different for Ni, Al and NiAl, and vary during the reaction because the spatial atomic composition of the system is varying with respect to the time. This issue would be discussed further in section 5 regarding the physical and chemical parameters of the equations.

### 2.2.3. Mathematical modeling: Geometric domain and initial/boundary condition

#### (1) Computational domain

Here, we define a geometric domain consisting of Ni and Al laminates in two dimensions.

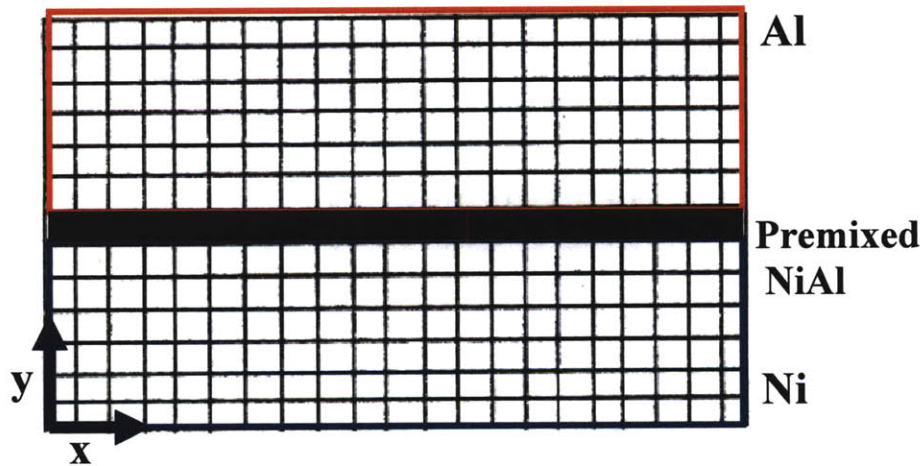


Figure 2-3

Fig. 4 Computational Domain for Ni and Al laminate

$$\rightarrow L_x = 1 \text{ mm} , L_y = 80 \text{ nm} , w = 2 \text{ nm}$$

In the domain,  $w$  is the premixed region of Ni and Al before the reactions. It is known that the premixed region thickness affects the reaction characteristic. The effect of the premixed region thickness would be discussed in this chapter.

## (2) Initial condition

Usually, analytical solutions for the steady state heat conduction equations in the Poisson type are expressed as impulse functions. Those impulse functions guarantee the existence and uniqueness of the given partial differential equations. So, it is important to adopt the initial temperature and atomic concentration profiles in terms of the impulse functions. Ideally, the Dirac delta function can be used for the initial conditions. Other types of impulse distributions such as triangular, rectangular, and Gaussian functions whose center is in peak value and has proper distribution width can be used. Among them, here, the Gaussian impulse distributions for the initial temperature and atomic compositions have been adopted for the C0, C1 and C2 continuity of the solutions for the governing equations.

### → Atomic composition

$$C = 1 \text{ for } 0 \leq x \leq L_x \text{ and } 0 \leq y \leq L_y/2 - w/2$$

$$C = 0 \text{ for } 0 \leq x \leq L_x \text{ and } L_y/2 - w/2 \leq y \leq L_y/2 + w/2$$

$$C = -1 \text{ for } 0 \leq x \leq L_x \text{ and } L_y/2 + w/2 \leq y \leq L_y$$

$$\text{premixed region } w : L_y/2 - w/2 \leq y \leq L_y/2 + w/2$$

$$C = 0 \text{ for } x = 0$$

This complex initial atomic composition profile also decays rapidly to the corresponding atomic composition (1 or -1) through the slab, and is described with the equation (2-10).

$$C(x, y) = (1 - \exp(-100 \cdot 10^3 \cdot x))^2 \cdot \tanh(10 \cdot 10^9 \cdot (y - L_y/2)) \quad (2-10)$$

### → Temperature profile

Also, an initial temperature profile is imposed in which the left boundary of the domain is at an effective adiabatic temperature for the initiation of the reaction. Also, the initial temperature profile decays rapidly to room temperature from the left boundary with the Gaussian distribution with its height as the adiabatic temperature for the impulsive thermal activation, and is described with equation (11).

$$T(x, y) = (T_f - T_{ambient}) \cdot \exp(-500 \cdot 10^6 \cdot x^2) + T_{ambient} \quad (2-11)$$

## (3) Boundary conditions



All boundaries are assumed to be insulated in terms of temperature and mass. That is, the Neumann boundary conditions with no heat and mass fluxes are imposed to all four boundaries.

$$\frac{\partial C}{\partial y} = \frac{\partial T}{\partial y} = 0$$

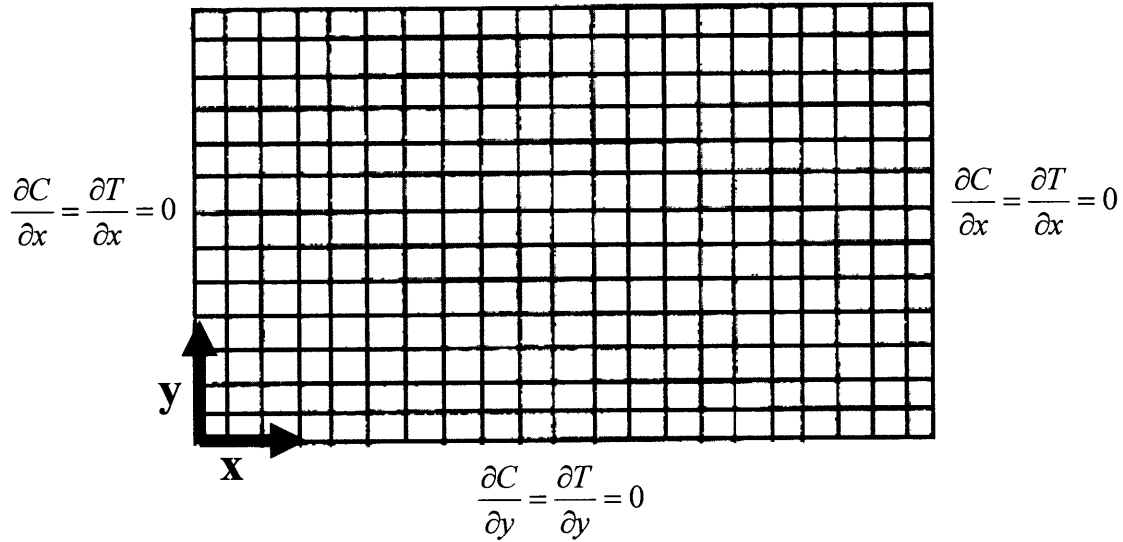


Figure 2-4 Neumann (insulated) Boundary Conditions for Four Boundaries

After this Neumann boundary condition at  $x = 0$ , the Dirichlet boundary condition would be imposed to the left boundary, and compared with the Neumann boundary's results.

### 2.3. Numerical modeling: Finite difference equations

To solve the governing equations with the proper physical and chemical parameters, the finite difference scheme is adopted. For the 1<sup>st</sup> and 2<sup>nd</sup> spatial derivative, 2<sup>nd</sup> order central finite difference methods are used.

However, the time integration scheme should be more carefully treated. In this problem, due to the broad spectrum of characteristic length scales associated with the problem, ranging from the bi-layer thickness (tens of nm) to the layer's longitudinal length (several mm), the problem is very stiff. In addition, the Arrhenius dependence of the atomic diffusion on temperature also compounds the spatial stiffness problem. Due to this stiffness, it is very dangerous to use explicit time integration scheme. For the explicit scheme, we should use an extremely small time step (under 0.1 nsec) for stability. So, in this problem, several implicit time integration schemes have been adopted and tested. Initially, the implicit trapezoidal

(Crank-Nicolson) scheme was adopted. However this scheme was proved to be not accurate for this problem. Also, the stability was not as good as expected. Thus, the fully implicit time integration scheme has been adopted and tested. By this scheme, the system has been proved to be very stable even in larger time steps. Also, the accuracy of the solutions was not sensitive to the time integration step size. So, in this problem, the time stepping size is adopted to be around 50 nsec. For this size, the solution was quite stable and proved to be reasonable.

(1) Finite difference equations for the heat equation

$$\begin{aligned}
& \rho(C_{i,j}^{n+1}) \cdot c_p(C_{i,j}^{n+1}) \cdot \frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} = \\
& \frac{\partial}{\partial x} \left( k(C^{n+1}) \frac{\partial T}{\partial x} \right) \Big|_{n+1} + \frac{\partial}{\partial y} \left( k(C^{n+1}) \frac{\partial T}{\partial y} \right) \Big|_{n+1} + \frac{\partial Q(C)}{\partial t} \\
& = \frac{1}{\Delta x} \left( k(C_{i+\frac{1}{2},j}^{n+1}) \frac{\partial T}{\partial x} \Big|_{i+\frac{1}{2},j}^{n+1} - k(C_{i-\frac{1}{2},j}^{n+1}) \frac{\partial T}{\partial x} \Big|_{i-\frac{1}{2},j}^{n+1} \right) \\
& + \frac{1}{\Delta x} \left( k(C_{i,j+\frac{1}{2}}^{n+1}) \frac{\partial T}{\partial y} \Big|_{i,j+\frac{1}{2}}^{n+1} - k(C_{i,j-\frac{1}{2}}^{n+1}) \frac{\partial T}{\partial y} \Big|_{i,j-\frac{1}{2}}^{n+1} \right) + \frac{\partial Q(C_{i,j}^{n+1})}{\partial t} \\
& = \frac{1}{\Delta x} \left( k \left( \frac{C_{i,j}^{n+1} + C_{i+1,j}^{n+1}}{2} \right) \frac{T_{i+1,j}^{n+1} - T_{i,j}^{n+1}}{\Delta x} - k \left( \frac{C_{i-1,j}^{n+1} + C_{i,j}^{n+1}}{2} \right) \frac{T_{i,j}^{n+1} - T_{i-1,j}^{n+1}}{\Delta x} \right) \\
& + \frac{1}{\Delta y} \left( k \left( \frac{C_{i,j+1}^{n+1} + C_{i,j}^{n+1}}{2} \right) \frac{T_{i,j+1}^{n+1} - T_{i,j}^{n+1}}{\Delta y} - k \left( \frac{C_{i,j-1}^{n+1} + C_{i,j}^{n+1}}{2} \right) \frac{T_{i,j}^{n+1} - T_{i,j-1}^{n+1}}{\Delta y} \right) \\
& + \frac{\partial Q(C_{i,j}^{n+1})}{\partial C} \cdot \frac{C_{i,j}^{n+1} - C_{i,j}^n}{\Delta t}
\end{aligned} \tag{2-12}$$

(2) Finite difference equations for the atomic diffusion equation

$$\begin{aligned}
\frac{C_{i,j}^{n+1} - C_{i,j}^n}{\Delta t} &= \frac{\partial}{\partial x} \left( D(T) \frac{\partial C}{\partial x} \right) + \frac{\partial}{\partial y} \left( D(T) \frac{\partial C}{\partial y} \right) \\
&= \frac{\partial}{\partial x} \left( D(T^n) \frac{\partial C}{\partial x} \right) \Bigg|_{n+1} + \frac{\partial}{\partial y} \left( D(T^n) \frac{\partial C}{\partial y} \right) \Bigg|_{n+1} \\
&= \frac{D(T^n_{i+\frac{1}{2},j}) \frac{\partial C}{\partial x} \Big|_{i+\frac{1}{2},j}^{n+1} - D(T^n_{i-\frac{1}{2},j}) \frac{\partial C}{\partial x} \Big|_{i-\frac{1}{2},j}^{n+1}}{\Delta x} \\
&\quad + \frac{D(T^n_{i,j+\frac{1}{2}}) \frac{\partial C}{\partial y} \Big|_{i,j+\frac{1}{2}}^{n+1} - D(T^n_{i,j-\frac{1}{2}}) \frac{\partial C}{\partial y} \Big|_{i,j-\frac{1}{2}}^{n+1}}{\Delta y} \\
&= \frac{1}{\Delta x} \left( D \left( \frac{T_{i+1,j}^n + T_{i,j}^n}{2} \right) \frac{C_{i+1,j}^{n+1} - C_{i,j}^{n+1}}{\Delta x} - D \left( \frac{T_{i-1,j}^n + T_{i,j}^n}{2} \right) \frac{C_{i,j}^{n+1} - C_{i-1,j}^{n+1}}{\Delta x} \right) \\
&\quad + \frac{1}{\Delta y} \left( D \left( \frac{T_{i,j+1}^n + T_{i,j}^n}{2} \right) \frac{C_{i,j+1}^{n+1} - C_{i,j}^{n+1}}{\Delta y} - D \left( \frac{T_{i,j-1}^n + T_{i,j}^n}{2} \right) \frac{C_{i,j}^{n+1} - C_{i,j-1}^{n+1}}{\Delta y} \right)
\end{aligned} \tag{2-13}$$

To solve the system of PDE for (15) and (16), at first, the atomic diffusion equation for (16) are solved for  $t = n+1$ . At this point, the diffusivity is calculated based on the current temperature ( $t = n$ ). After then, the atomic composition for each point is calculated implicitly. After calculating the atomic composition at  $t = n+1$ , the temperature is calculated fully implicitly with equation (16). By this semi-implicit time integration scheme, we could not guarantee unconditional stability. That is, we need to restrict the time integration step  $dt = C * \min(dx, dy)$ . However, the stability could be achieved with much bigger time step than the fully explicit time integration steps. That is, we could achieve the stable solutions for the system even with  $C = 100$ . However, for the accuracy of the solutions, the  $C$  was adopted in 50. Also, when we construct the derivative operators in (15) and (16), we could not construct one operator containing all the first derivatives in equation (15) and (16) because the atomic diffusivity, heat conductivity, material density, and heat capacity are all varying over space and time with the updated

solutions for the atomic compositions and temperatures. So, each derivative operator for the leftward, rightward, upward and downward first differentiation in space has been constructed separately with the updated  $C$  and  $T$  at every time step. That is, four first-derivative operators have been constructed with the variable coefficient with the updated temperature and atomic compositions using 2D tensor product at every time step. Most of the time in the simulations has been spent in constructing those operators at every time step.

#### 2.4. Study for physical and chemical parameters, and issues on the parameters

$$\frac{\partial C(x, y, t)}{\partial t} = \nabla \circ [D(T(x, y, t)) \cdot \nabla C(x, y, t)] \quad (2-4)$$

$$\rho \cdot c_p \cdot \frac{\partial T(x, y, t)}{\partial t} = \nabla \circ (k \cdot \nabla T(x, y, t)) + \frac{\partial Q(C(x, y))}{\partial t} \quad (2-8)$$

To solve the time-dependent and coupled parabolic system of partial differential equations (4) and (8), we should first study the proper physical and chemical parameters before numerically solving. In the governing equations, the atomic diffusion is represented with a single binary diffusivity  $D$  for Ni-Al system. This is also calculated based on experimental data.

$$D = D_0 \exp\left(-\frac{E}{RT}\right) \quad (2-12)$$

The values for  $E$  and  $D_0$  are 137 kJ/mol and  $2.18 \times 10^{-5} \text{ m}^2 / \text{s}$  obtained from experimental data [41-42].

Also, the formation heat in Ni-Al reactions is quadratic-fitted from experimental data [41-42].

$$\begin{aligned} Q &= -\rho \cdot c_p (T_{f0} - T_0) \cdot C^2 \\ T_{f0} &= T_0 + 1660K \end{aligned} \quad (2-13)$$

Where  $\rho$  and  $c_p$  are the weighted average density and heat capacity of those for the pure Ni and Al at room temperature, respectively.

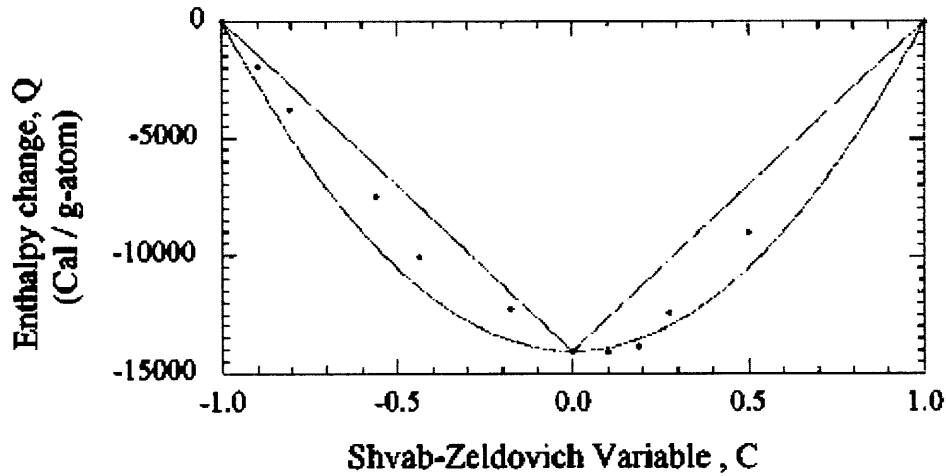


Figure 2-5 Experimental data for the formation heat during reactions [46].

In the heat equation (8), the heat conductivity, material density and heat capacity vary with respect to Ni, Al, and NiAl. However, usually, in this kind of alloying combustion problem, those physical and chemical parameters have been assumed to be constant. That is, the weighted average values from pure reactants are used for the reacted region, and kept constant during the time evolution. As the reaction front propagates through the laminate, those assumptions would be reasonable for the reacted region. Thus, the heat equation is consequently treated as a simple parabolic PDE having constant coefficient. Consequently,  $\nabla \circ (k \cdot \nabla T(x, y, t))$  is simply converted to  $k \nabla^2 T(x, y, t)$ . However, through this chapter, the temporary and spatially varying parameters for the heat conductivity, heat capacity and density during the time evolutions are considered in the heat equations. This presents an interesting challenge for this problem. That is, the heat equation would be considered as the following equation.

$$\rho(C(x, y, t)) \cdot c_p(C(x, y, t)) \cdot \frac{\partial T(x, y, t)}{\partial t} = \nabla \circ (k(C(x, y, t)) \cdot \nabla T(x, y, t)) + \frac{\partial Q(C(x, y))}{\partial t} \quad (2-14)$$

The varying parameters would be calculated by linear interpolations between those of Ni ( $C=-1$ ) and Al ( $C=1$ ) based on the mass concentration  $C(x, y, t)$  updated at every time step.

## 2.5. Numerical results with Dirichlet boundary conditions

For figure 2-5 to 2-12, the following conditions are assumed:

$T_0$  at  $x = 0$ : 1830K (fixed)

$C_0$  at  $x = 0$ : 0 (fixed)

$T_{ambient} = 300K$ , Premixed thickness  $w$ : 4 nm,  $dx$ : 2  $\mu m$ ,  $dy$ : 1 nm,  $dt$ : 50 nsec

(1) Initial T and C profiles

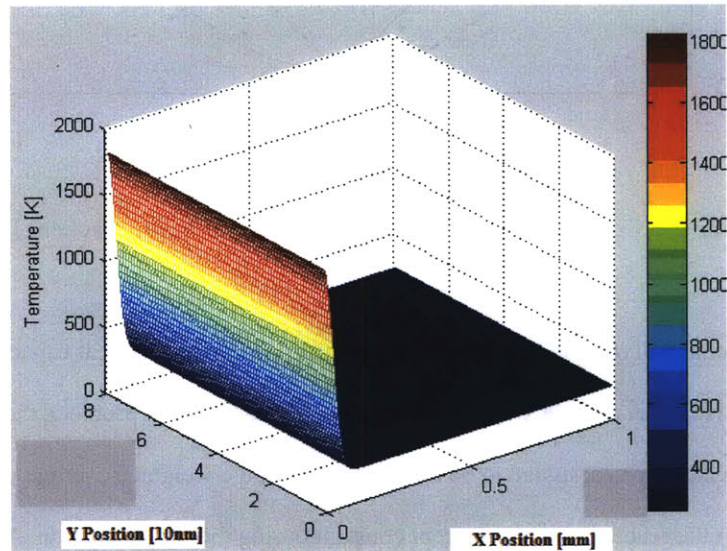


Figure 2-5 Initial temperature profile with Dirichlet boundary condition

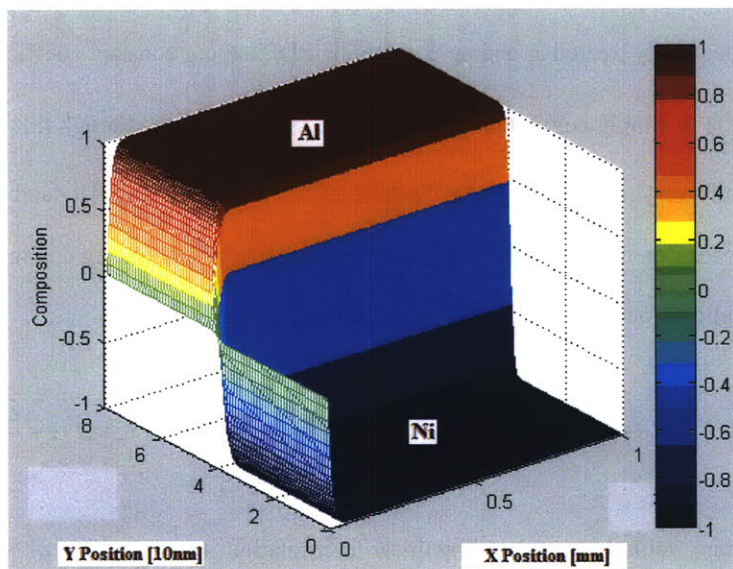


Figure 2-6 Initial composition profile with Dirichlet boundary condition (1 for Al, -1 for Ni, and 0 for NiAl)

(2) Time evolution of temperature and composition

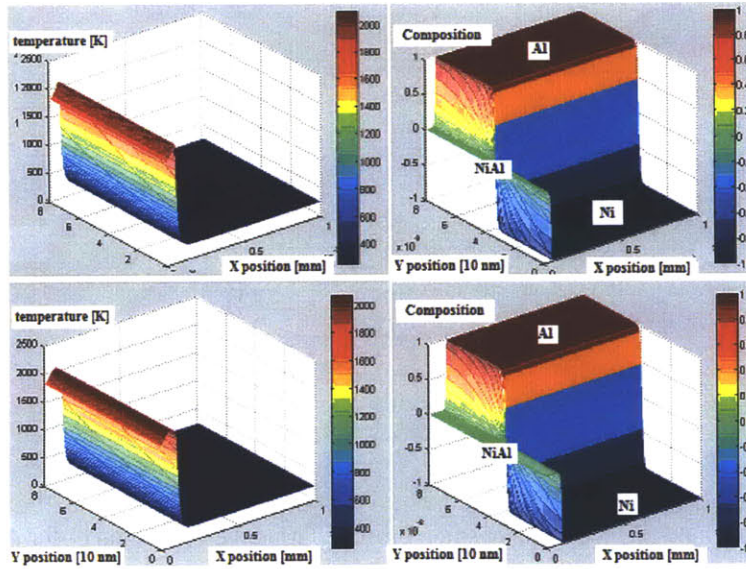


Figure 2-7 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

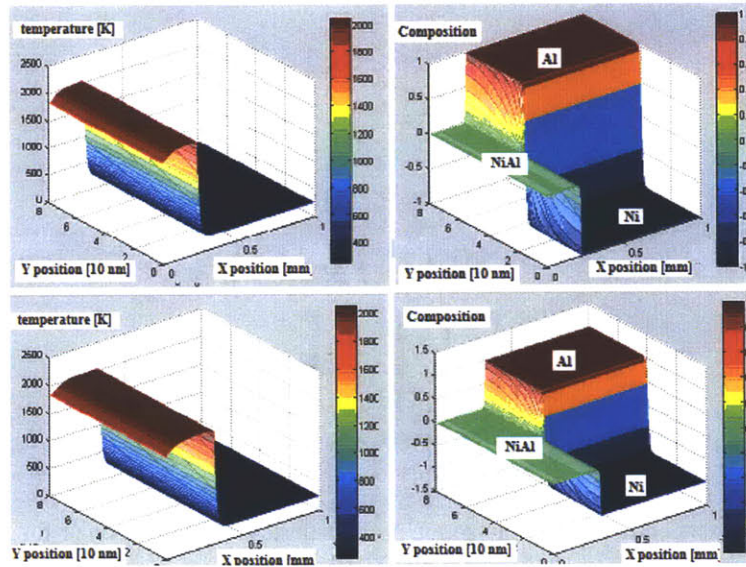


Figure 2-8 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

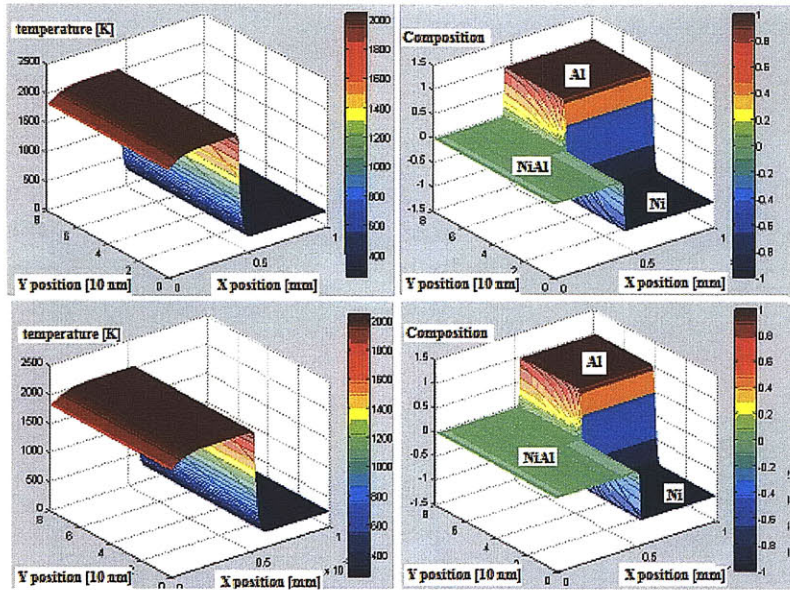


Figure 2-9 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

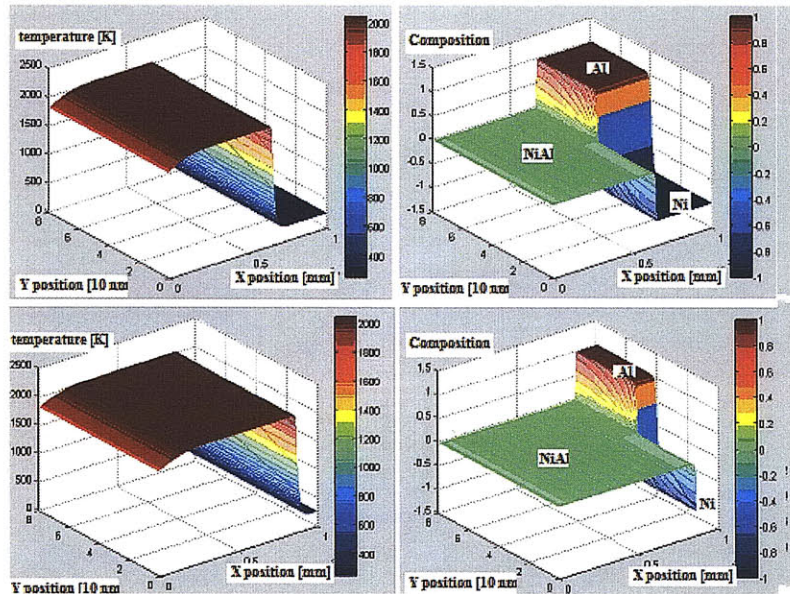


Figure 2-10 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

The temperature and composition distributions over the laminate have been captured every 200 time steps (10  $\mu$  sec) for the whole time (5000 time steps, 250  $\mu$  sec) for the simulation. As seen in the above figures, the heat propagates through the laminate, and its shape is also sustained after first overshoot.



Also, periodic superheated regions by small disturbances in the reactions are observed. However, over the simulation time, the steady state propagations are dominantly observed. That is, over time, we observe: (1) first overshoot in temperature in the early stage of the reactions (transient region); (2) steady state propagations with constant reaction front velocity; and (3) periodic small oscillations (about 20 – 40 K superheating) in temperature between steady state propagations.

Also, heat diffusion via laminate longitudinal direction is much more dominant than the heat diffusion in the thickness direction because the laminate thickness (80 nm) is much smaller than the laminate length (1mm). However, mass diffusion with the reactions via the thickness direction is much more dominant than the mass diffusion in the longitudinal direction because the reaction between Ni and Al occurs across Ni and Al slabs (thickness direction). Also, with the temperature evolution, atomic composition of the remaining backside regions of the reaction front clearly converges to 0 values (fully reacted).

(3) Time evolution of temperature, power and reaction zone

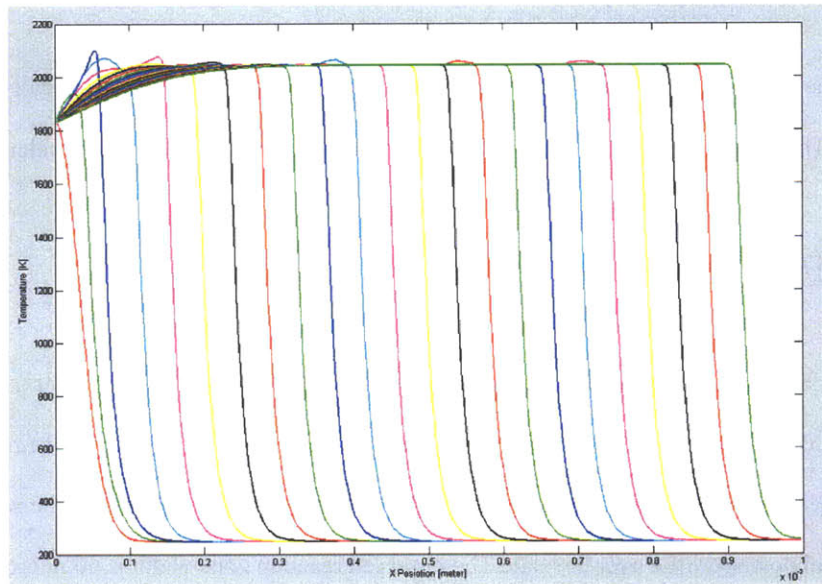


Figure 2-11 Selected temperature profiles in  $y = \frac{L_y}{2}$  (middle of the laminate) during the simulation

As seen in Fig. 2-11 for the temperature profile evolution in the middle of the laminate, we could clearly observe (1) the early stage of the first overshoot (first superheating), (2) steady state propagations, and (3) oscillatory superheating (20 – 40 K over the reaction temperature, 2000K) regions between steady state propagations. Also, the period of the oscillation is 200  $\mu\text{m}$ . We find that the reaction front velocity in

the steady state propagations is 4.5 m/s. These values would be compared with the experimental data in the next section.

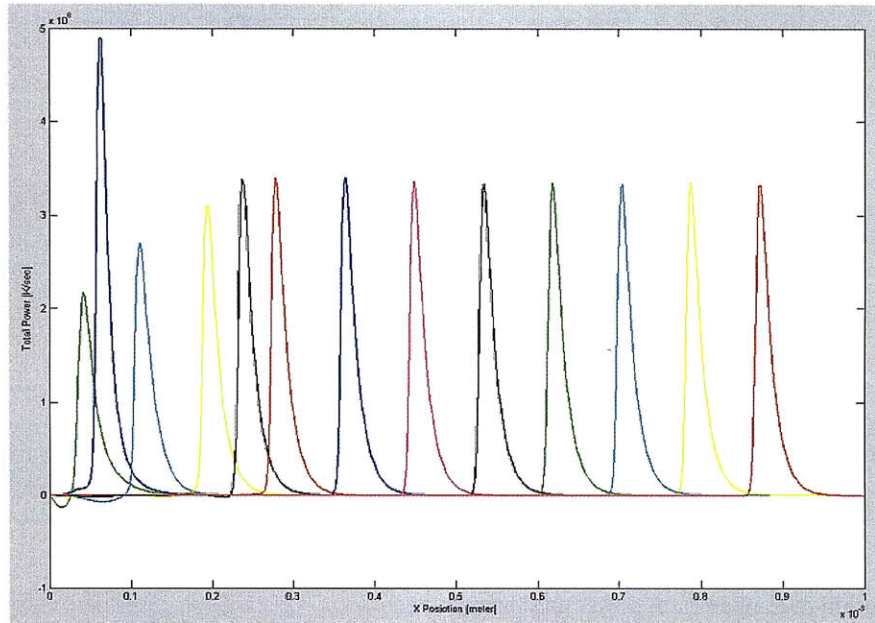


Figure 2-12 Selected total power profiles during the simulation (averaged along thickness (y) direction)

With the temperature evolutions, spatially averaged total power evolutions are calculated during the simulation. That is, every time step, the total power exerted on each spatial point is calculated as below.

$$total\ power\ [K/sec] = \rho(C(x,y,t)) \cdot c_p(C(x,y,t)) \cdot \frac{\partial T(x,y,t)}{\partial t} \quad (2-14)$$

These data are averaged over the thickness direction for every longitudinal point (x point) and captured every 200 time steps. By this total power evolution, we could clearly observe the reaction propagations over the laminate. Also, we could calculate the reaction zone width from the width of the peak region in the total power profile over the slab. The observed reaction zone width is 60 microns.

## 2.6. Numerical result with Neumann boundary conditions

For figure 2-13 to 2-20, the following conditions are assumed:

$$T_0\ at\ x = 0: 1830K\ (dT/dx = 0)$$

$$C_0\ at\ x = 0: 0\ (dC/dx = 0)$$

$$T_{ambient} = 300K, \text{ Premixed thickness } w: 4\ nm, \ dx: 2\ \mu m, \ dy: 1\ nm, \ dt: 50\ nsec$$

In this section, the left boundary conditions for temperature and composition are selected as Neumann boundary conditions. That is, during the simulation, insulated boundary condition ( $dT/dx = dC/dx = 0$ ) is imposed at the left boundary. Other simulation conditions are the same as the previous simulation.

(1) Initial T and C profiles

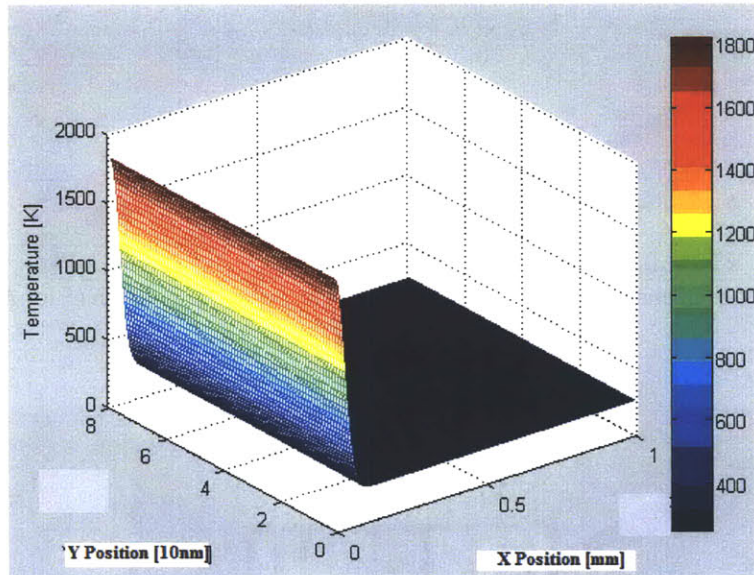


Figure 2-13 Initial temperature profile with Neumann boundary condition

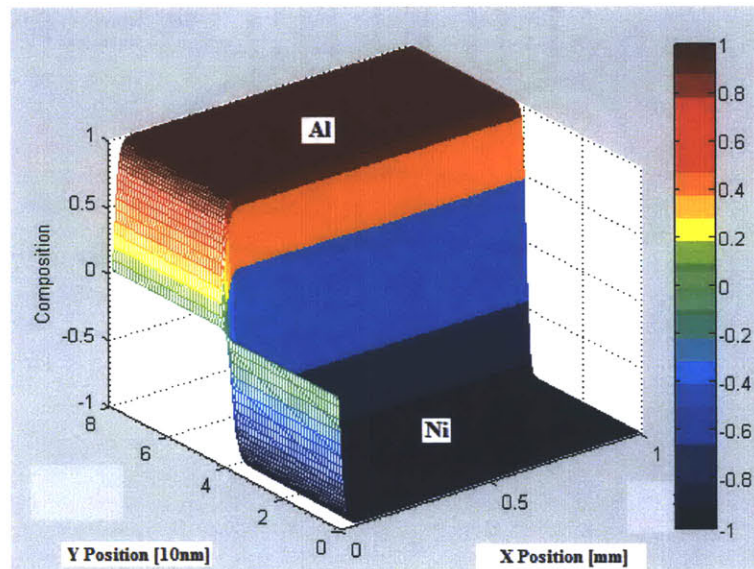


Figure 2-14 Initial composition profile with Neumann boundary condition (1 for Al, -1 for Ni, and 0 for NiAl)

(2) Time evolution of temperature and composition

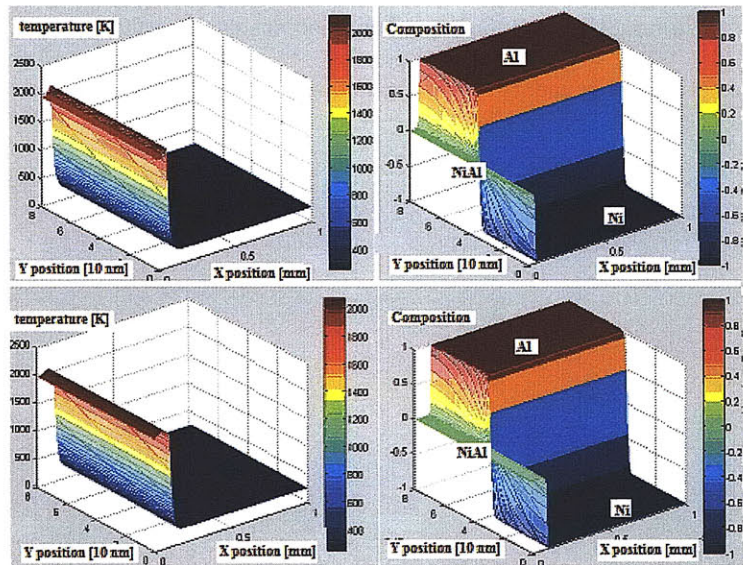


Figure 2-15 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

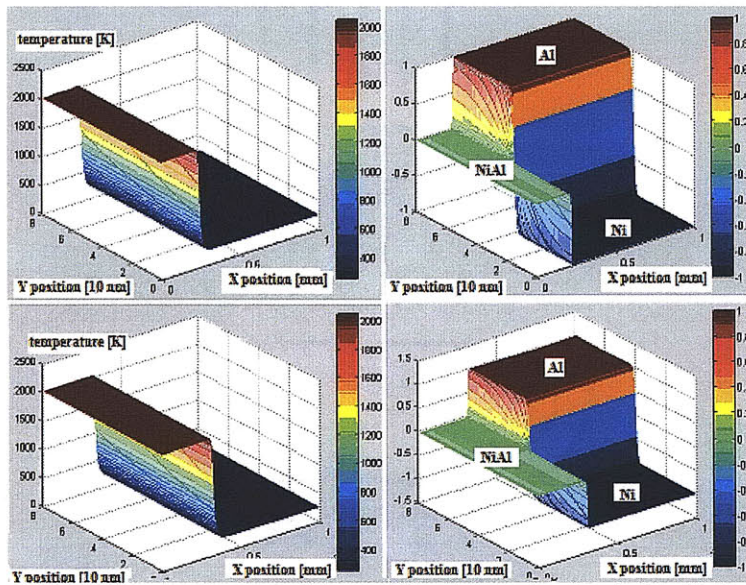


Figure 2-16 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

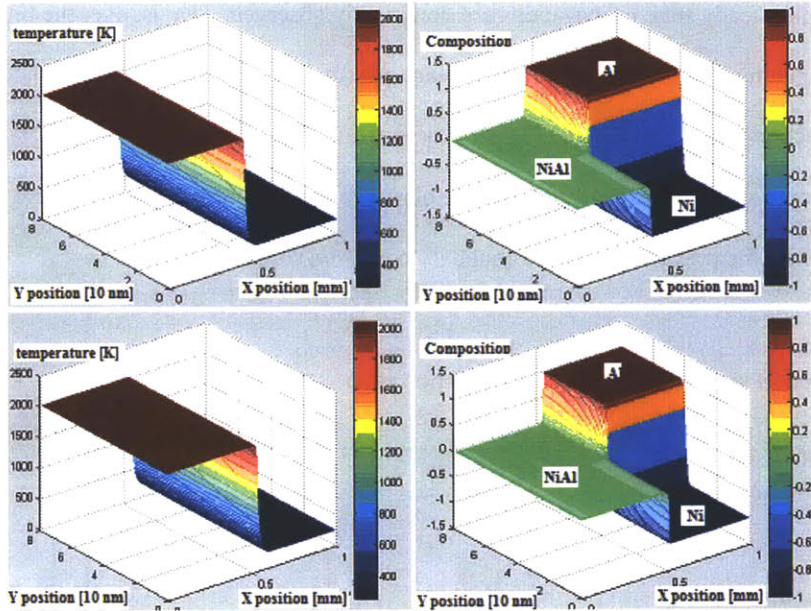


Figure 2-17 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

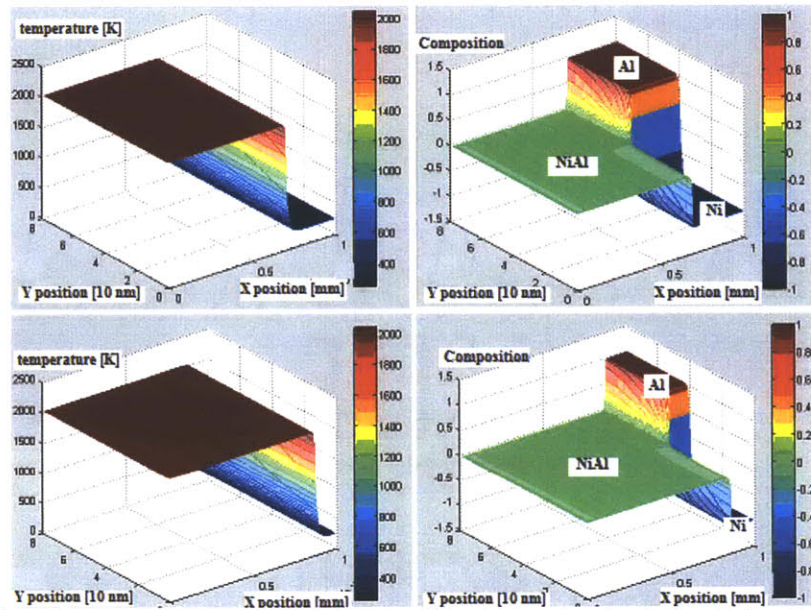


Figure 2-18 Selected temperature and composition profiles during the simulation (1 for Al, -1 for Ni, and 0 for NiAl)

The temperature and composition distributions over the laminate have been captured every 200 time steps ( $10 \mu\text{SEC}$ ) for the whole time (5000 time steps,  $250 \mu\text{SEC}$ ) for the simulation. As seen in the above figures, the heat propagates through the laminate and its shape is also sustained after first overshoot. Also, periodic superheated regions by small disturbances in the reactions are observed. However, over the

simulation time, the steady state propagations are dominantly observed. That is, over the time, (1) first overshoot in temperature in the early stage of the reactions (transient region) (2) steady state propagations with constant reaction front velocity, and (3) periodic small oscillations (about 20 – 40 K superheating) in temperature between steady state propagations, are observed through the simulation. These observations are very similar with the previous Dirichlet boundary condition case.

(3) Time evolution of temperature, power and reaction zone

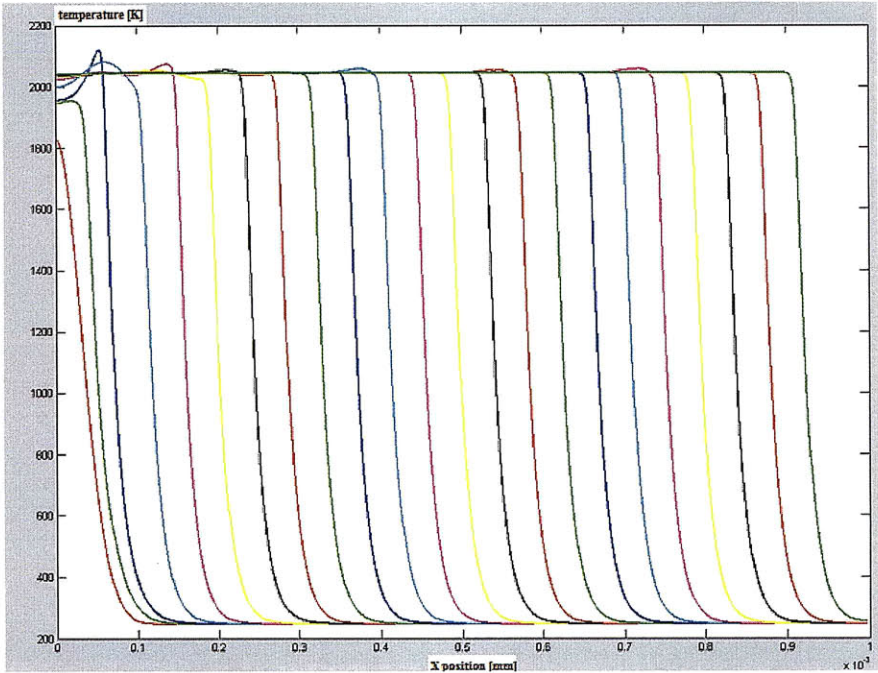


Figure 2-19 Selected temperature profiles in  $y = \frac{L_y}{2}$  (middle of the laminate) during the simulation

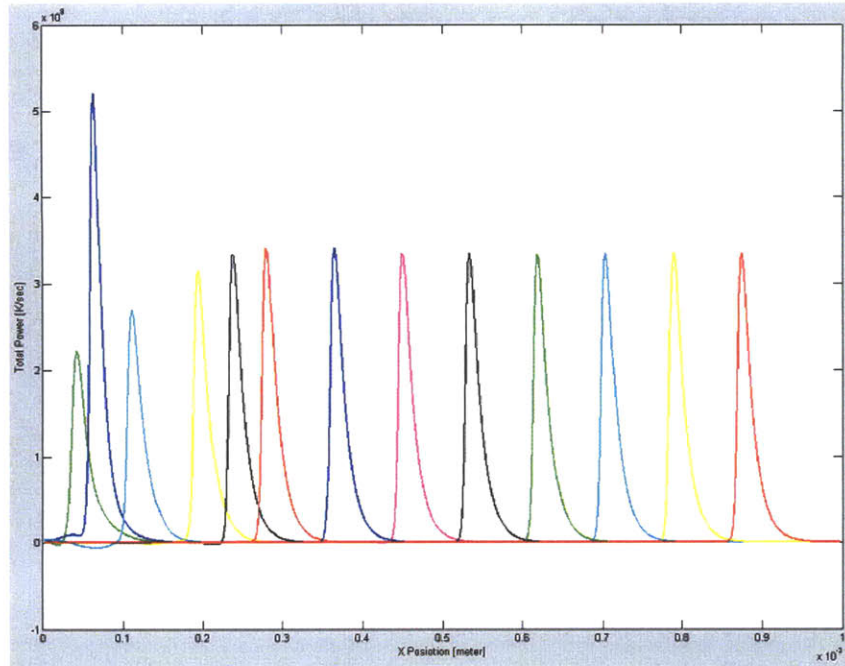


Figure 2-20 Selected total power profiles during the simulation (averaged along thickness (y) direction)

The reaction front velocity (4.5m/s), oscillation period (200  $\mu\text{m}$ ) and strength (20 – 40K), and reaction zone width (50  $\mu\text{m}$ ) are very close to those of Dirichlet boundary condition case. This means that simulation results are not very sensitive to the boundary conditions for the  $T$  and  $C$  at the left boundary because the overall simulation time is very short, and consequently, the Dirichlet boundary condition for  $T = T_0$  at the left boundary is also considered as an impulsive heat input for the activation. That is, the Dirichlet boundary condition for  $T = T_0$  at the left acts as a heat bath at the left during the simulation. However, it does not affect the total simulation result much. Also, the additional heat input during the time could be negligible because the heat formation generated by the combustive reactions is much bigger than the additional heat input from the left heat bath.

## 2.7. Effect of Premixed Thickness

$T_0$  at  $x = 0$ : 1830K ( $dT/dx = 0$ )

$C_0$  at  $x = 0$ : 0 ( $dC/dx = 0$ )

$T_{\text{ambient}} = 300\text{K}$ , Premixed thickness  $w$ : 2 nm,  $dx$ : 2  $\mu\text{m}$ ,  $dy$ : 1 nm,  $dt$ : 50 nsec

It is known that the characteristics of the heat propagations in the laminate are sensitive to the premixed region thickness by experimentations. In this chapter, the premixed Ni-Al region between Ni and Al slabs is adjusted to 2 nm (previous cases: 4 nm).

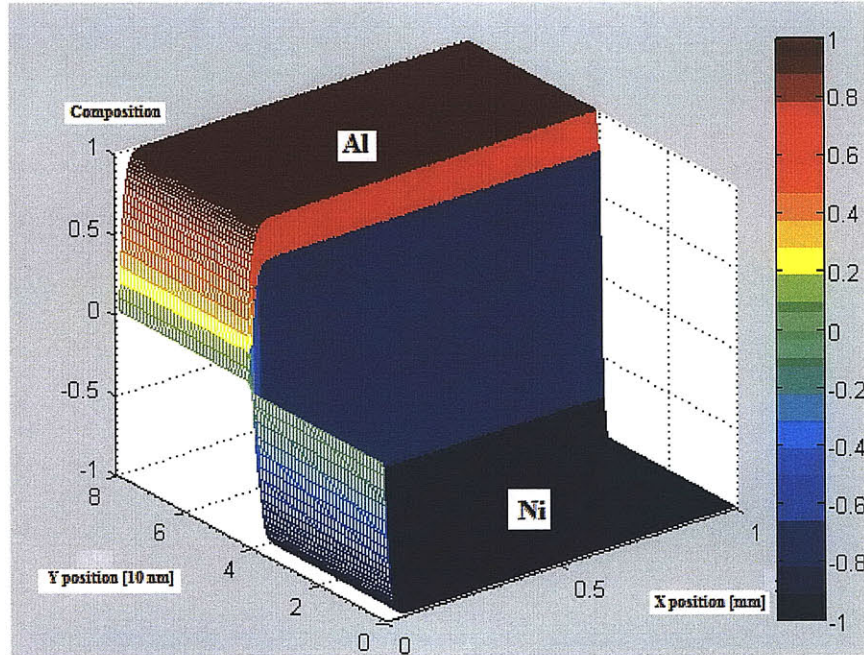


Figure 2-21 Initial composition profile with Neumann boundary condition (1 for Al, -1 for Ni, and 0 for NiAl)

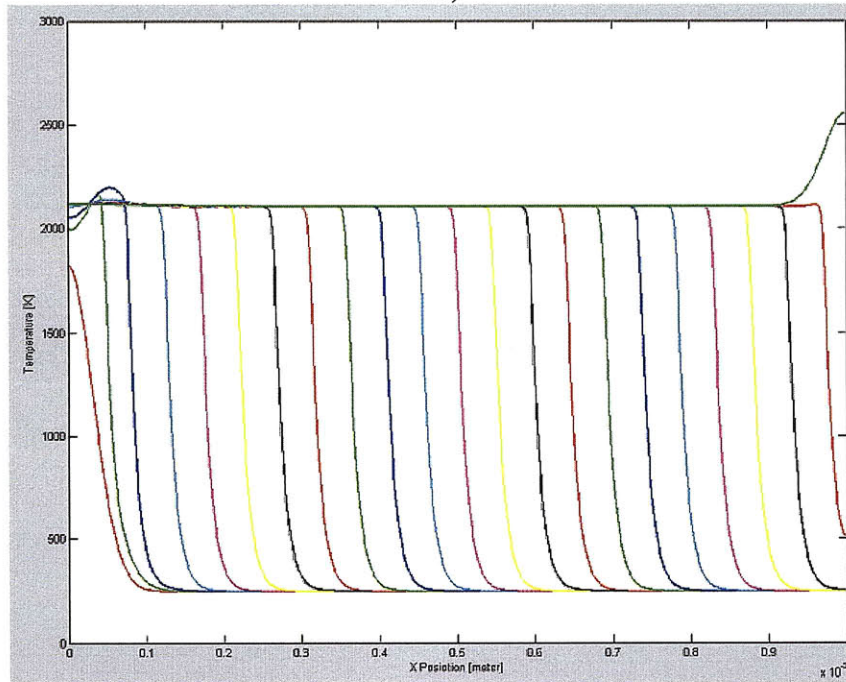


Figure 2-22 Selected temperature profiles in  $y = \frac{L_y}{2}$  (middle of the laminate) during the simulation

With the small premixed thickness, we could observe that the first overshoot (superheating) and, consequent oscillatory superheating between steady state propagations are clearly alleviated. The heat



propagations become steady states more rapidly. Also, the reaction front velocity is 4.8 m/s which is faster than the previous cases with 4 nm thickness (4.5 m/s).

### 2.8. Effect of Ambient Temperature

$T_0$  at  $x = 0$ : 1830K ( $dT/dx = 0$ ),  $C_0$  at  $x = 0$ : 0 ( $dC/dx = 0$ )  
 $T_{\text{ambient}} = 330\text{K}$ , Premixed thickness  $w$ : 2 nm,  $dx$ : 2  $\mu\text{m}$ ,  $dy$ : 1 nm,  $dt$ : 50 nsec

It is known that the characteristics of the heat propagations in the laminate are also sensitive to the ambient temperature by experimentations. In this section, the ambient temperature is adjusted to 340K (previous cases: room temperature).

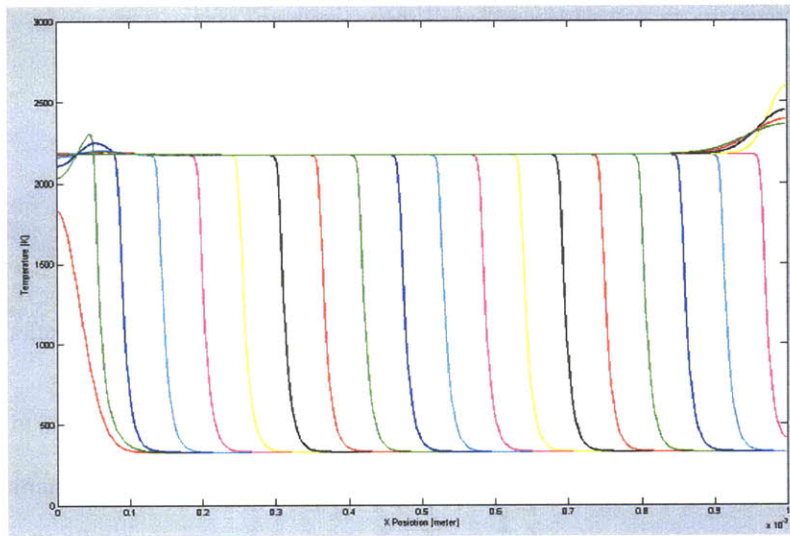


Figure 2-23 Selected temperature profiles in  $y = \frac{L_y}{2}$  (middle of the laminate) during the simulation

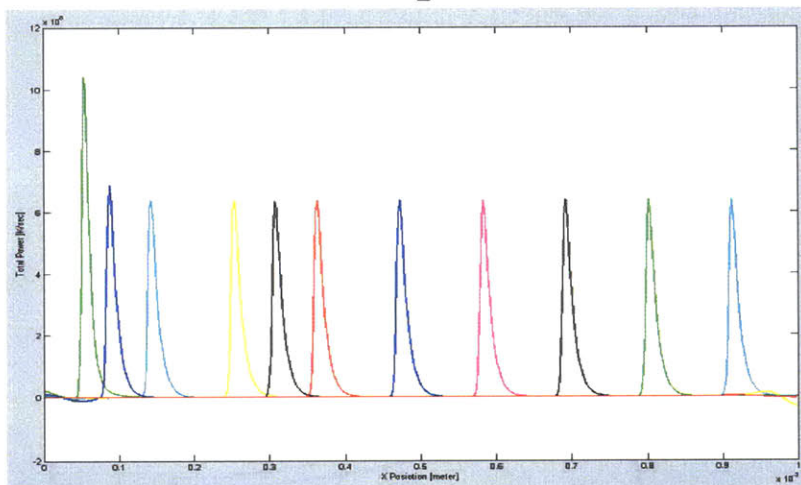


Figure 2-24 Selected total power profiles during the simulation (averaged along thickness ( $y$ ) direction) With the higher ambient temperature, here, 30K than the room temperature, we could observe that

the first overshoot (superheating) and, consequent oscillatory superheating between steady state

propagations are clearly alleviated. The heat propagations become steady states more rapidly. Also, the reaction front velocity is 5.2 m/s, which is faster than the previous cases with the room temperature and smaller premixed region thickness.

### 2.9. Comparison with Experiments

In this chapter, several features from the numerical results would be discussed by comparison with the possible experimental results.

#### → Self-sustainability and overall behavior of the heat propagation

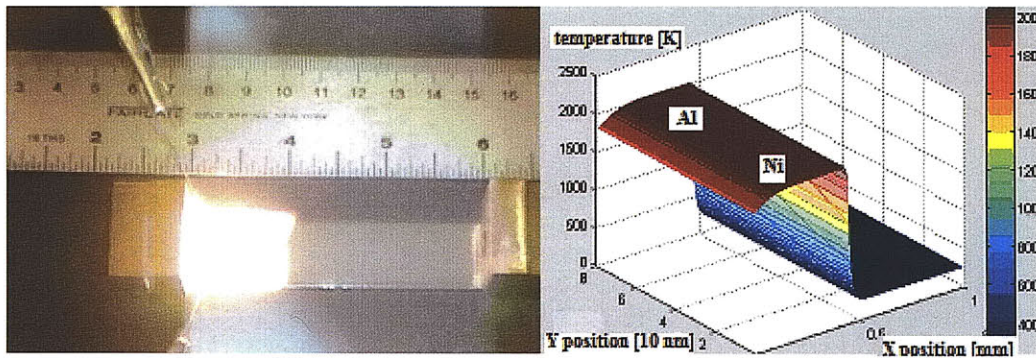


Figure 2-25 Overall behavior of the heat propagation in experiment and numerical result (J. Trenkle et al., 2008, unpublished)

As seen in the figure, the overall behavior of the heat propagation is very close to the experimental result. That is, after initial activation of the reactions, the heat propagates through the laminate longitudinal direction with a self-sustained manner.

#### → Reaction front velocity

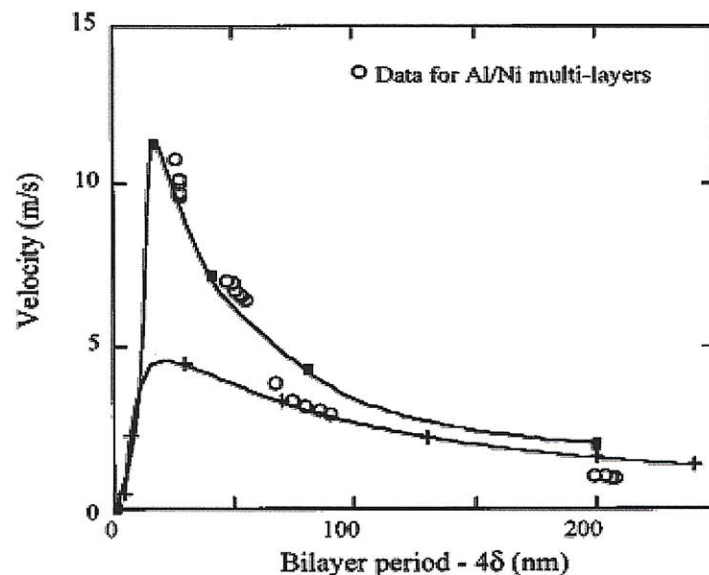


Figure 2-26 Experimental data for the reaction front velocities [42].

Experimentally, the reaction front (propagation) velocity in the steady state propagations is observed in 4 – 5 m/s with the bilayer period (the Ni-Al laminate's overall thickness). In our simulation results with 80 nm bilayer thickness, the reaction front velocity was calculated as 4.5 – 4.7 m/s.

Also, it is known that the reaction front velocity is accelerated by decreasing the premixed thickness. In our simulation, the reaction front velocity for a 2 nm of premixed width  $w$  was faster than the velocity of 4 nm.

### 2.10. Conclusion

In this chapter, the thermally induced reactions in Ni and Al laminates have been modeled and numerically solved in the coupled partial differential equations. By the numerical simulation, we could observe the self-sustainability of the reaction propagations with the proper boundary and initial conditions for the thermal shock. We compared the numerical results on the reaction front velocity, oscillatory phenomena, reaction width, and the affect of ambient temperature and premixed regions in the laminates to the current experimental data, and they are well matched to the experiments. However, in the numerical simulations, we assumed only insulated thermal boundary conditions and do not consider any phase transformation during the reactions. However, in real systems, the heat should be lost by convection and radiation, and the phase of the system should be changed to liquid state during the reaction propagations. In the further study on this topic, these fundamental assumptions should be updated.

## Chapter 3

# Theoretical and computational background in molecular dynamics simulations

### 3.1. Introduction to the classical molecular dynamics simulations

#### 3.1.1. Classical mechanics formulation for equations of motions

Molecular dynamics simulations have been developed to describe molecular systems composed of  $N$  interacting particles in terms of the classical mechanics on the interacting particles. From Newton's 2<sup>nd</sup> law, we describe the equations of motions of  $N$  interacting particles.

$$\begin{aligned} m_i \ddot{\vec{r}}_i &= \vec{F}_i = -\nabla U \\ i &= 1, 2, \dots, N \end{aligned} \quad (3-1)$$

Equivalently, in the classical mechanics, we describe Newton's equations of motions of  $N$  interacting particles as Hamilton's principles of least action.

Using the generalized coordinates,  $\vec{q}_i$  and velocities,  $\dot{\vec{q}}_i$ , we define the action,  $S$  as the equation (3-2)

$$\begin{aligned} L(\vec{q}_i, \dot{\vec{q}}_i, t) &= K - U \\ S &= \int_{t_1}^{t_2} L(\vec{q}_i, \dot{\vec{q}}_i, t) dt \end{aligned} \quad (3-2)$$

In classical mechanics, we need to find  $\vec{q}_i$  and  $\dot{\vec{q}}_i$  such that the action  $S$  is an extremum.

Applying the arbitrary small variations  $\delta\vec{\eta}_i(t)$  and  $\delta\dot{\vec{\eta}}_i(t)$  to  $\vec{q}_i$  and  $\dot{\vec{q}}_i$  which satisfy the boundary conditions, we can find the equations of motion for the generalized coordinates and velocities. From equation (3-3) with the variations, we obtain the following Euler Lagrange equations of motions for the system.

$$\delta S = 0 \quad (3-3)$$

$$\frac{\partial L}{\partial \vec{q}_i} - \frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\vec{q}}_i} \right) = 0, \quad t_1 < t < t_2 \quad (3-4)$$

Usually,  $L$  is called the Lagrangian function of the system defined as (Kinetic energy) – (Potential energy).

Finally, by introducing the generalized momenta from the Lagrangian function instead of the generalized velocities, we can describe the system in another version of Hamiltonian mechanics.

Hamiltonian function of the system is defined as a Legendre transformation on the generalized velocities of the Lagrangian function.

$$H(\vec{q}_i, \vec{p}_i, t) = \sum_i \vec{p}_i \cdot \dot{\vec{q}}_i - L(\vec{q}_i, \dot{\vec{q}}_i, t) = K + U \quad (3-5)$$

$$\vec{p}_i = \frac{\partial L}{\partial \dot{\vec{q}}_i}$$

By introducing the generalized momenta from the Lagrangian function, we can make the system from 2<sup>nd</sup> order system of  $N$  degrees of freedom to 1<sup>st</sup> order system of  $2N$  degrees of freedom.

Finally, from the Euler Lagrange equation (3-4) and the definition of the generalized momenta (3-5), we obtain the canonical equations of motions in Hamiltonian systems by taking the total derivative of the Hamiltonian function in the equation (3-5).

$$\dot{\vec{q}}_i = \frac{\partial H}{\partial \vec{p}_i}$$

$$\dot{\vec{p}}_i = - \frac{\partial H}{\partial \vec{q}_i} \quad (3-6)$$

In the molecular dynamics simulations, we solve the equations of motions, (3-1), (3-4), or (3-6) of the systems with the interacting forces acting on each atom, and the system of the equations of motions is a

set of 1<sup>st</sup> or 2<sup>nd</sup> order ordinary differential equations which can be numerically solved with various methods.

Also, we can calculate the interatomic forces acting on atoms by taking the gradient of the potential energy calculated from the current atomic configuration. In the molecular dynamics simulations, the interatomic potential called the force field in the system is the most important basis by which to describe the atomic system. A variety of interatomic potentials have been developed for many material systems based on both quantum mechanical calculations and experimental data. For the interatomic potentials, we would discuss more in Chapter 5 and 8 on the embedded atom method potential for the metallic systems and the reactive force field potential for the fully reactive systems of various chemical species, particularly Carbon, Hydrogen, Oxygen, and Nitrogen.

### **3.1.2. Computational techniques in the molecular dynamics simulations**

In the molecular dynamics simulations, we obtained the current atomic position and velocity by solving the systems of the differential equations. In the computer simulations of molecular dynamics, most computational resources are spent in calculating the interatomic potential energy given at the atomic configurations, and the interatomic potential energy vanishes rapidly as the interatomic distance increases. Therefore, it is impossible and inefficient to calculate all the interactions in the system for each time. That is, we need to calculate the interatomic potential energy or force of the system based on the nearest neighbor atoms which contribute the energy or force significantly. To realize this, we need to construct the neighbor lists of atoms in the system during the simulations. The most popular and simplest neighboring scheme in the molecular dynamics is to track and store neighbor atoms on each atom during the simulations. In this method, the neighbor list of the atoms is constructed based on the atomic distance within the proper cutoff radius. That is, for a specific atom, all the atoms within the cutoff radius from the central atom are stored in the neighbor list for the atom. However, it is not efficient to construct the neighbor lists for all atoms in the system every single time step. So, in the practical simulations, the neighbor lists are constructed by tracking whether some atoms move into the cutoff distance or escape from it through the skin distance out of the cutoff distance. In molecular dynamics simulations, choosing the neighboring scheme and frequency is very important in terms of computational efficiency. However, all the different neighboring schemes should give the same results for the same conditions.

One of the most important computational issues on the molecular dynamics simulations is the periodic boundary condition. In the infinite systems of the materials such as bulk systems, the periodic conditions can be applied to the boundaries in specific directions assumed to be in the infinite bulks.

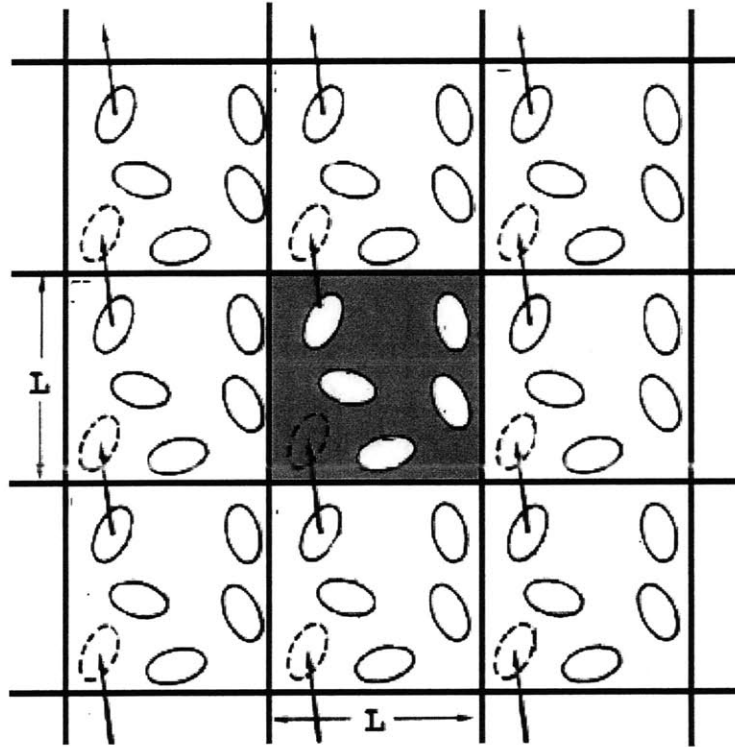


Figure 3-1 Periodic boundary conditions in MD [47].

In the periodic boundary conditions, if some atom moves out of the current simulation box, another atom having the same velocity moves into the simulation box. This is very useful to describe the bulk systems of materials. However, if the system is finite and has free surfaces, the periodic boundary conditions should not be applied to the direction.

The required computational resources increase rapidly as increasing the number of atoms in the systems. Though it is more realistic to increase the number of atoms to be as many as possible in terms of physical properties calculated from the simulations, it is not possible to increase the number of atoms as required for the macroscale continuum level, in the order of mm. Currently, the maximum number of atoms simulated is around tens or hundreds of billion. By this computational limitation in the molecular dynamics simulations, the parallelized computations have been used for the simulations. There are many methods to

parallelize the calculations on the computers. Through this research, the spatial decomposition method which decomposes the simulation box into processors used in the parallel computing and distribute the atomic information through the processors has been implemented and used. This spatial decomposition based parallelization is very similar to the implementation of the periodic boundary conditions.

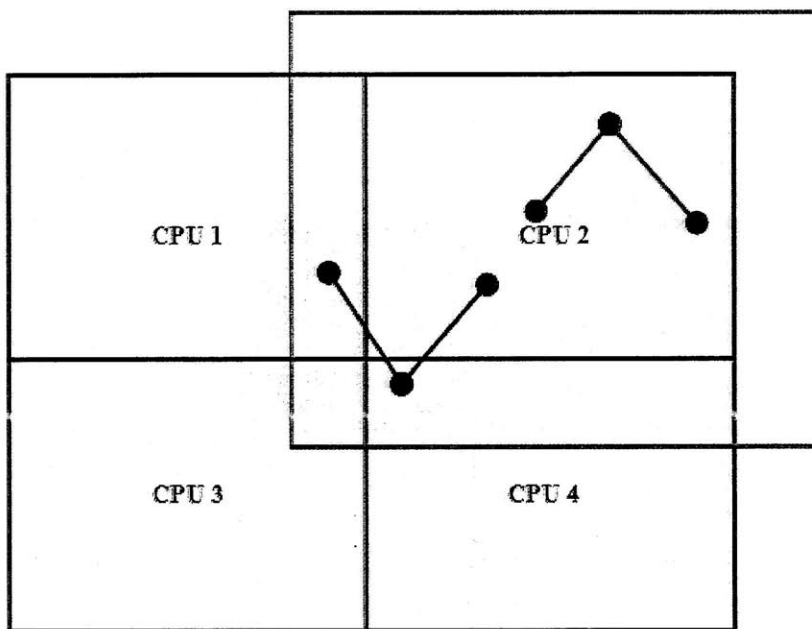


Figure 3-2 Spatial decomposition method in parallelized computation in MD

As described in figure 3-2, the simulation box is divided into the number of processors in the simulation, and the atomic information including the velocities and positions corresponding to each spatial bin in the simulation box is allocated into each corresponding processor. Then, each processor calculates the dynamics for the atomic in the bin. Around the boundaries of each processor, there should be interactions between atoms in different processors. To track this, every single processor traces and stores neighbor lists called the ghost atoms around the boundaries. For sharing and communicating the information of the ghost atoms in the lists, message passing interfaces (MPI) is used to broadcast it through the processors. By this MPI, overlapping calculations by the different processors on the same atoms could be prevented. In this research, all the MD simulations have been performed using the parallel computations with 8 – 128 processors, and the interatomic potentials including the embedded method potentials for Nickel and Aluminum hybrid systems, and the reactive force field (ReaxFF) potential with the charge



equilibration method with the parallelized conjugate gradient algorithms have been implemented in parallelized computation environments.

### 3.2. Statistical mechanics fundamentals: Ensembles and Ergodic hypothesis

#### 3.2.1. Statistical assumption: microstates and ensembles with the Ergodic hypothesis

In the molecular dynamics simulations, the classical statistical mechanics of the particles is a fundamental theoretical background. From the statistical treatments on the interacting particles, we can describe the physical properties of the systems consisting of the interacting particles. In the statistical mechanics, each atomic configuration called a microstate has the same macroscopic physical properties such as temperature and pressure with a small fluctuation for the given equilibrium state. The fundamental postulate of the statistical mechanics is the Ergodic hypothesis such that all possible microstates where the macroscopic physical properties take the same appear with an equal probability during a long time. By the Ergodic hypothesis, we can calculate the macroscopic physical properties from the ensemble average of the microstates which is assumed to be the same as the long time average.

$$\langle A \rangle_{NVE \text{ or } NVT \text{ or } NPT} = \langle A \rangle_{\tau} \quad (3-7)$$

Also, there are several fundamental equilibrium ensembles such as the micro canonical ( $NVE$ ), canonical ( $NVT$ ), isothermal-isobaric ( $NPT$ ), and grand canonical ensemble ( $\mu VT$ ) for the given experimental conditions. In the next sections, each equilibrium ensemble and the corresponding thermodynamic potential would be discussed.

#### 3.2.2. Micro canonical ensemble: standard isolated system

For the isolated systems, the number, volume and total energy (kinetic energy and potential energy) of the system are fixed for the equilibrium state. In the NVE ensemble, the system wants to maximize the total entropy for the equilibration. For example, the isolated system consisting of two subspaces having different temperature is equilibrated by the heat transport between the subspaces by maximizing the total entropy of the system, and, therefore, the temperature of the subspaces is equilibrated into the same value. In the NVE ensemble, the temperature and entropy of the system are defined in the

next equations.

$$S(E) = k_B \ln \Gamma(E)$$

$$1/T = \frac{\partial S}{\partial E} \quad (3-8)$$

In the equation (3-8),  $\Gamma(E)$  is the total number of microstates corresponding to the total energy, E.

### 3.2.3. Canonical ensemble: constant temperature system

For the canonical ensemble, the number, volume, and temperature of the system are fixed for the equilibrium state. The system in the equilibrium with the constant temperature is assumed to contact to the external heat bath to supply the energy to conserve the temperature. For the constant temperature system, we can calculate the partition function of all the possible states from the Boltzmann distribution for the given energy state.

$$Q_{NVT} = \sum_{\text{all microstates}} e^{-E_N/k_B T} = \int dE \cdot \Gamma(E) e^{-E_N/k_B T}$$

$$= \int dE \cdot e^{-\frac{1}{k_B T}(E-T \cdot S(E))} = \int dE \cdot e^{-\frac{1}{k_B T}A(T,E)} \quad (3-9)$$

In the equation (3-9), we define the thermodynamic potential of the constant temperature system, called Helmholtz free energy.

$$A = E - TS \quad (3-10)$$

In the equation (3-9), we assume most microstates in the equilibrium are around the peak value of the distribution for the canonical ensemble.

$$Q_{NVT} = \int dE \cdot e^{-\frac{1}{k_B T}A(T,E)} = \max_E \left( e^{-\frac{1}{k_B T}A(T,E)} \right)$$

$$\Leftrightarrow \min A(T, E) \quad (3-11)$$

By this assumption, we conclude the thermodynamic potential, Helmholtz free energy,  $A = E-TS$  is minimized at the equilibrium.

### 3.2.4. Isothermal-Isobaric ensemble: constant temperature and pressure system

For the isothermal-isobaric ensemble, the number, pressure, and temperature of the system are fixed for the equilibrium state. The system in the equilibrium with the constant temperature and pressure is assumed to contact to the external heat bath and pressure bath to conserve the temperature and pressure. For the constant temperature and pressure system, we can calculate the partition function of all the possible states from the Boltzmann distribution for the given energy state.

$$\begin{aligned}
 Q_{NPT} &= \sum_{\text{all microstates}} e^{-(E_n + PV_n)/k_B T} = \sum_{\text{all volumes, } V} \sum_{\text{all microstates for a } V} e^{-(E_n + PV)/k_B T} \\
 &= \int dV \cdot e^{-(A(T, V) + PV)/k_B T} \\
 &= \max_V e^{-(A(T, V) + PV)/k_B T}
 \end{aligned} \tag{3-12}$$

In the equation (3-12), we define the thermodynamic potential of the constant temperature and pressure ensemble, called Gibbs free energy.

$$G = A + PV = E - TS + PV \tag{3-13}$$

Finally, in the equation (3-12), we conclude the thermodynamic potential, Gibbs free energy,

$G = A + PV$  is minimized at the equilibrium.

### 3.2.5. Classical mechanics counterparts and implementations of NVE, NVT and NPT ensembles: the extended Hamiltonian methods

In molecular dynamics simulations, we need to realize the equilibrium ensembles discussed in the previous sections in classical mechanics formulations which can be implemented in the simulations. In this section, the extended Hamiltonian (Lagrangian) methods would be introduced briefly.

To realize the system contact to the external heat bath to conserve temperature, in the canonical, NVT ensembles, the nonphysical artificial and dynamical variables called thermostat variables are introduced to the systems in the classical mechanics formulations. The thermostat variable has also the coordinates and corresponding momentum with the artificial “effective” mass. By adding the dynamic thermostat variables, though the real Hamiltonian of the system is not conserved, the extended Hamiltonian

including the kinetic energy of the thermostat variables is conserved during the simulations. There are many algorithms to realize the constant temperature molecular dynamics simulations such as Nose-Hoover thermostat and chain methods [33]. Through this research we have used the Nose-Hoover thermostat algorithm for the constant temperature molecular dynamics simulations.

In the constant pressure (stress) molecular dynamics simulations, the ensembles are generated by allowing the volume of the system to fluctuate. Thus, the volume must be introduced as a dynamical variable in the equations of motions. By introducing another artificial and dynamical variables having the effective mass, coupled to the volume of the system which is also dynamical, called barostat variable, the internal pressure (stress) is conserved, fluctuating about the external, applied pressure. Also, the barostat variable is coupled to the thermostat variables during the dynamics. By adding the dynamic barostat variable, though the real Hamiltonian of the systems is not conserved, the extended Hamiltonian including the kinetic energy of the thermostat and barostat variables, and PV energy is conserved during the simulations. Through this research, we have used the Melchionna-Ciccotti-Holian algorithms [32, 35], another version of the Hoover algorithm for the constant pressure molecular dynamics simulations.

In chapter 6 on the molecular dynamics simulations of the Nickel and Aluminum nanoslab impacts, the anisotropic NPT simulations have been used to relax the global axial stresses of the system with the constant temperature in the equilibration processes before applying the impacts.

In chapter 7 on the molecular dynamics simulations of the Nickel and Aluminum nanoparticle impacts, the NVT (constant temperature and shape ensemble) simulations have been used to relax the system with constant temperature and shape in the equilibration processes before applying the impacts. For the finite system such as the nanoparticles, we can never control the external pressure by the barostat variables. For the finite systems under desired external pressure, we need to add inert gases around the system in the simulation box to control the external pressure applied to the system. However, for the metallic systems, the external pressure around 1 ATM (0.1 MPA) cannot affect the physical properties of the system significantly. That is, for the finite system under vacuum, we just allow the system to expand free to the vacuum with fluctuations with the desired temperature. Finally, during the non-equilibrium processes, we can never apply the thermostat and barostat variables during the simulations. That is, for the non-equilibrium processes, by introducing the temperature and pressure controls, the dynamic properties

we obtain from the simulations are non-physical. Through this research, we will use the thermostat and barostat methods based molecular dynamics simulations for the equilibration of the systems, but we will use the non-equilibrium molecular dynamics simulations to realize the impact processes of the Nickel and Aluminum systems accompanied with complex chemomechanical behaviors after the equilibration processes.

### **3.3. Calculations of physical properties in the molecular dynamics simulations**

We obtain the physical properties of the system consisting of the interacting atoms from the results of the molecular dynamics simulations. In this section, important physical properties calculated from the molecular dynamics simulations would be addressed.

#### **(1) Potential energy**

In the molecular dynamics simulations, the interatomic potential energy of the system is calculated from the interatomic force fields for the system based on the current atomic configurations. In terms of the interacting forces acting on atoms, the interatomic potential function is the most important in the molecular dynamics simulations. Currently, there are many potential functions for materials. In the early stage of the molecular dynamics simulations, one of the simplest, a pairwise interaction function was developed to describe the interacting monatomic gases and solids. As pairwise interactions, various Lennard Jones (9-3 or 12-6) potentials have been developed for many material systems. As the Lennard Jones potentials are relatively simple to implement, they are still used in many material systems. Also, the pairwise potential is a fundamental to many complex potential functions which have been developed. For the metallic systems which we are interested in the current research, the embedded atomic method (EAM) potential is very powerful and widely used. In the EAM, the potential is composed of pair interaction parts and embedding energy parts, which are functions of the electron energy of the atom. By the embedding energy part contributed by the nonlinear electron energy functions, we can describe the many body interaction effects in the solid metals. We would discuss the EAM more in Chapter 4.

#### **(2) Temperature**

From the equi-partition theorems in the classical statistical mechanics, we describe the

instantaneous temperature of the system in the equation (3-14).

$$T = \frac{2}{3NK_B} \sum_i \frac{1}{2} m_i (\vec{V}_i - \vec{u}) \quad (3-14)$$

In the classical molecular dynamics, we describe the instantaneous temperature from translational kinetic energy contributions in three dimensions. The rotational and vibration kinetic energy contributions are not taken into account because we treat every single atom as a point mass particle. In the equation (3-14), we need to extract local fluctuations of the atomic velocities with respect to the average flow velocity of the system. That is, we need to consider the localized material flow velocities for the temperature. If some atoms in a spatial bin have localized flow velocities, we have to remove the effect of the flow velocities in the temperature calculations.

### (3) Stress and pressure

From the virial theorem in the classical mechanics, we obtain the instantaneous local stress tensors called virial stress tensors of the local volume in the equation 3-15 [31].

$$\tilde{\Omega} = -\frac{1}{V_C} \left( \sum_{\alpha \in V_C} m_\alpha (\vec{V}_\alpha - \vec{u}) \otimes (\vec{V}_\alpha - \vec{u}) + \frac{1}{2} \sum_{\alpha \in V_C} \sum_{\beta \neq \alpha} \vec{x}_{\alpha\beta} \otimes \vec{F}_{\alpha\beta} \right) \quad (3-15)$$

The first term is the kinetic contribution to the stress tensor, and the second term is the virial contribution to the stress tensor. Also, the interatomic force in the virial part is calculated from the gradient of the interatomic potential with respect to the relative position vector between atoms. In the pairwise potential and EAM potential, the virial contribution part in the equation is held correctly. However, for other potentials including bond, angle, and dihedral interactions, the virial contribution part should be extended to include those interaction terms to the stress tensors. Also, there is another important stress measure in the molecular dynamics simulations, Hardy stress tensor [49]. In the Hardy stress tensor, we don't need to consider the atomic volume held in the equation (3-15). Instead, we define the local atomic distribution function for the kinetic part and bond function for the virial part in the stress tensor for the Hardy stress tensor. In many papers, there are still conflicts between the classical virial tensor and Hardy

stress tensor. Also, it has been proven that the Hardy stress tensor is more accurate in some cases [30]. However, there are no critical differences in the two stress measures, especially in the shock stress evolutions which would be discussed in the next chapters. Thus, in this research, the classical virial stress measures have been used to describe the instantaneous stress states of the system during the simulations. Also, the macroscopic pressure of the system is obtained from the trace (normal stress components) of the virial stress tensor divided by the current volume of the system.

## Chapter 4

### Interatomic potentials: Embedded Atom Method (EAM) potentials for Ni and Al systems

#### 4.1. Embedded Atom Method potentials for Ni and Al systems

In classical atomistic and molecular dynamics simulations of material systems, systems of linear differential equations for motions of particles are explicitly integrated every time step. In the equations of motions, external forcing term could be obtained by calculating gradients of potential energy between particles based on the given atomic configurations. The interatomic potential energy is constructed by empirical force fields. So, in the atomistic and molecular dynamics simulations, it is most important to choose and construct empirical force fields which could describe physical and chemical properties of the system accurately. Such empirical force fields are constructed and fitted based on quantum mechanical first principle calculations and experimental data.

For metallic bond systems, the first principle calculations based the Embedded Atomic Method (EAM) potential has been established. It has been proved that the EAM potential accurately describes the physical and chemical properties of many transition metals including Ni and Al at various phases [40].

The EAM potential energy is described in the equation (4-1).

$$E_i = F_i^\alpha \left( \sum_{j \neq i} \rho_j^{\alpha(j)}(r_{ij}) \right) + \frac{1}{2} \sum_{i,j} \psi_{ij}(r_{ij}) \quad (4-1)$$

Unlike a common linear pair interaction potential such as Lennard Jones model, the EAM potential consists of nonlinear functional terms for multibody interactions described by the embedding energy functions based on electronic densities around the central atom as well as linear pair interaction terms. By this multibody contribution to the embedding energy, the EAM potential can accurately describe surface and interface properties of the systems as well as bulk properties.



The general EAM potential does not differentiate atom species of the surrounding host atoms when constructing the density function around the central atom. The atom species of the central atoms is only considered in the density function regardless of atom species of the host atoms. So, for the alloy systems consisting of several different atom species, the general EAM potential is not appropriate. There are several advanced types of the EAM potentials which could correct this limitation. Finnis-Sinclair (F-S) EAM potential is one of the possible choices. Unlike the general EAM, the atom species of the host atoms and central atom are taken into account simultaneously in the F-S EAM potential when constructing the density function. The F-S EAM potential energy is described in the equation (2) [50].

$$E_i = F_i^\alpha \left( \sum_{j \neq i} \rho_j^{\alpha(i), \beta(j)}(r_{ij}) \right) + \frac{1}{2} \sum_{i,j} \psi_{ij}(r_{ij}) \quad (4-2)$$

Using the F-S formalism, the EAM potentials for Ni, Al, and their alloy systems have been constructed recently by several research groups. For the atomistic and molecular dynamics simulations in this project, the F-S type EAM potential (Mishin NiAl potential) for Ni and Al systems has been adopted and the potential parameters have been implemented in tabulated forms [41]. In the Mishin's EAM potential for Ni and Al system, the atom species of the surrounding host atoms is taken into account when constructing the electron density function in the embedding energy part. In the F-S EAM potentials, it has been known that it is much more important to consider the atom species of the surrounding host atoms rather than the atom species of the central atoms for the binary systems. So, the Mishin's EAM potential for Ni and Al systems is expressed as the equation (4-3)

$$E_i = F_i^\alpha \left( \sum_{j \neq i} \rho_j^{\beta(i)}(r_{ij}) \right) + \frac{1}{2} \sum_{i,j} \psi_{ij}(r_{ij}) \quad (4-3)$$

Also, though other F-S EAM potentials for Ni and Al systems are generally constructed as a linear combination of two different atoms' EAM potentials which are constructed separately, the Mishin's EAM potential for Ni and Al systems is best fitted based on the first principle calculations and experimental data for Ni, Al and NiAl binary systems in various phases simultaneously. Thus, the Mishin's EAM potential has been proven to be one of the most appropriate for Ni and Al systems.

#### 4.2. Potential functions: electron density, embedding energies, and pair interaction energies

In this section, based on the implemented Mishin's Ni and Al systems potential, potential parameters fitted for the multibody embedding energy with the electron density corresponding to the surrounding host atom species, and pair interaction terms would be addressed.

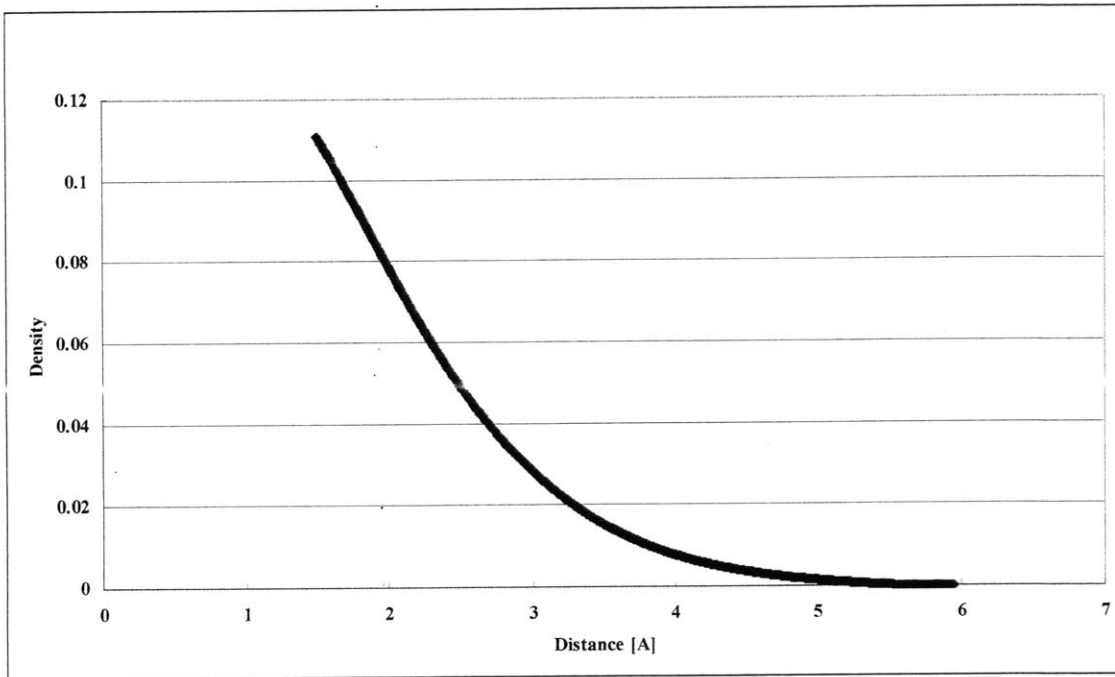


Figure 4-1 Ni density function for EAM

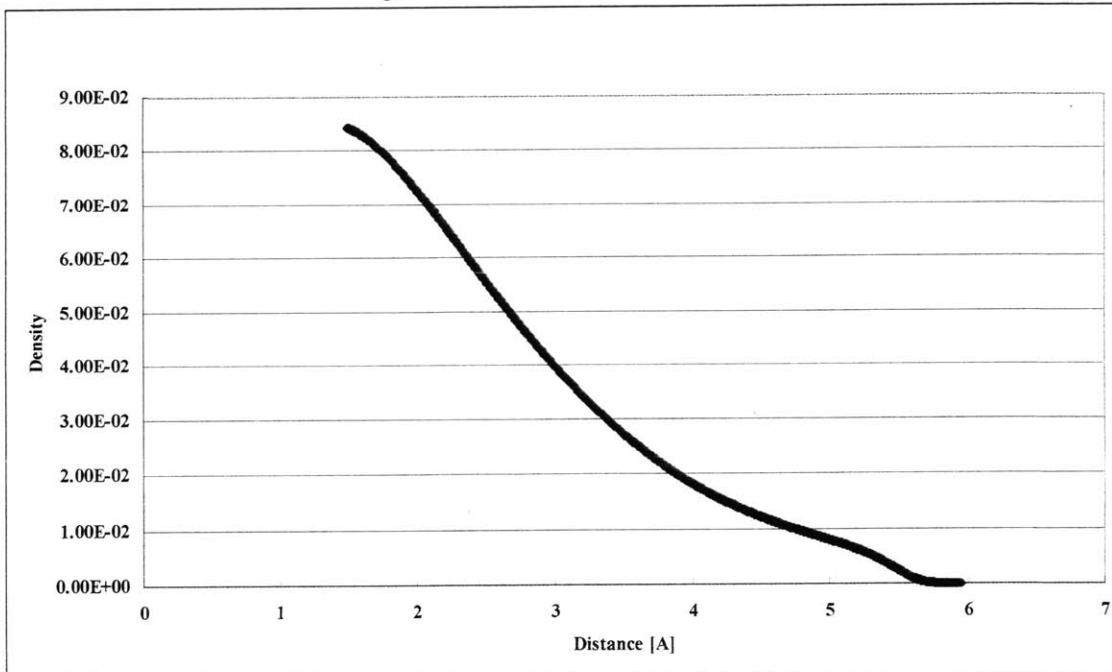


Figure 4-2 Al density function for EAM

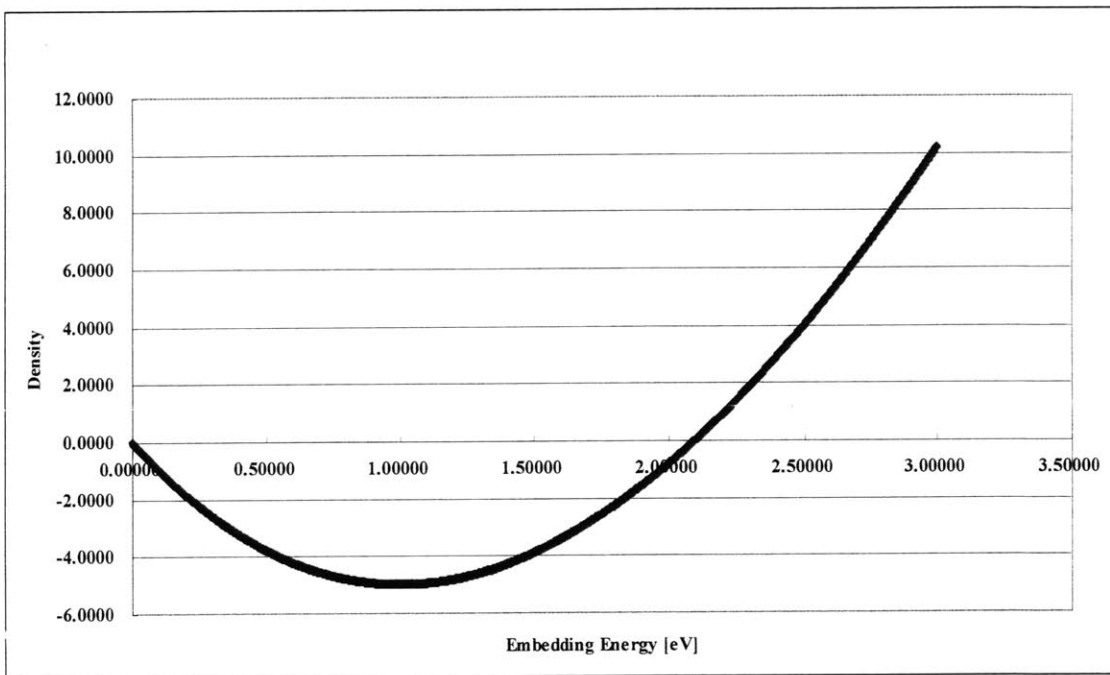


Figure 4-3 Embedding energy function for Ni in EAM

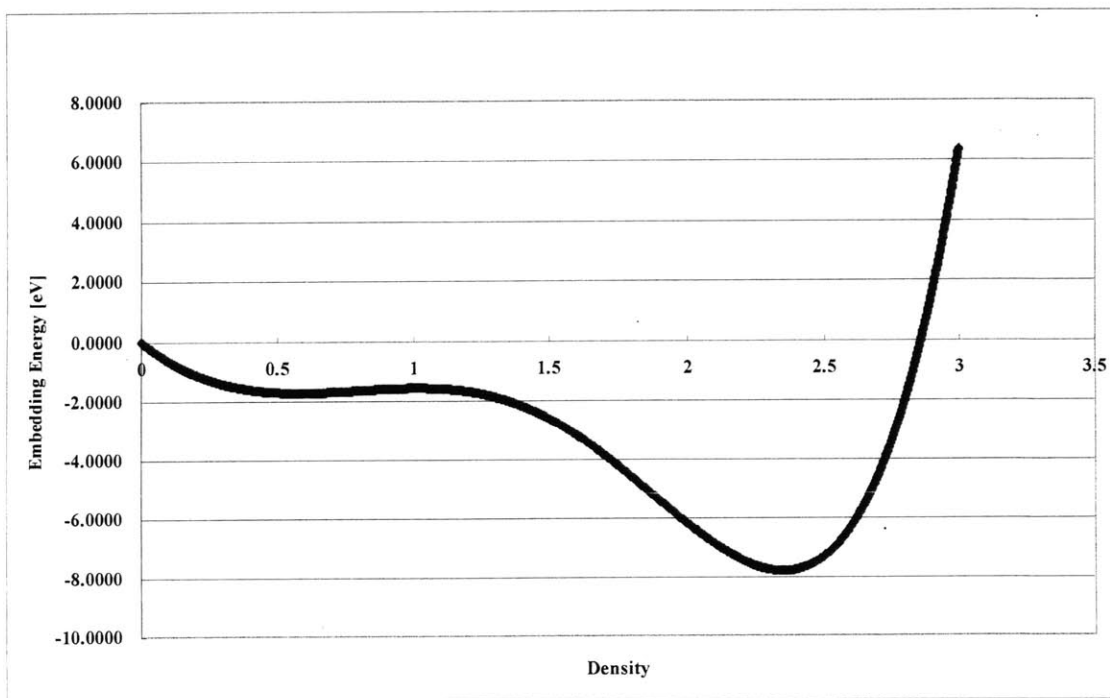


Figure 4-4 Embedding energy function for Al in EAM

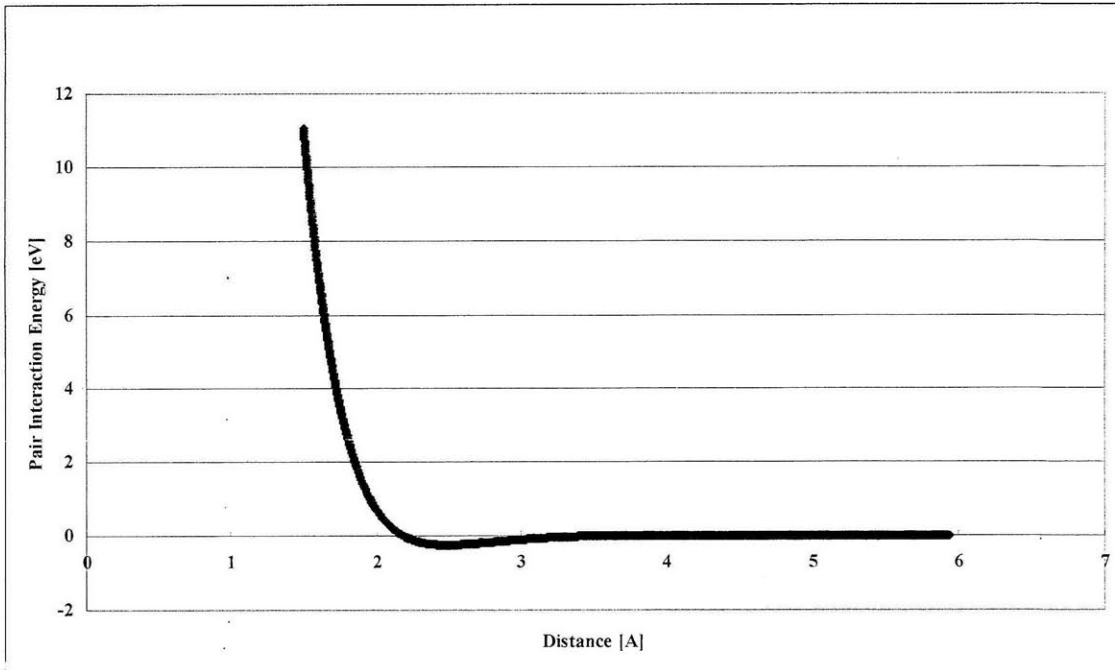


Figure 4-5 Pairwise interactions for Ni and Al in EAM

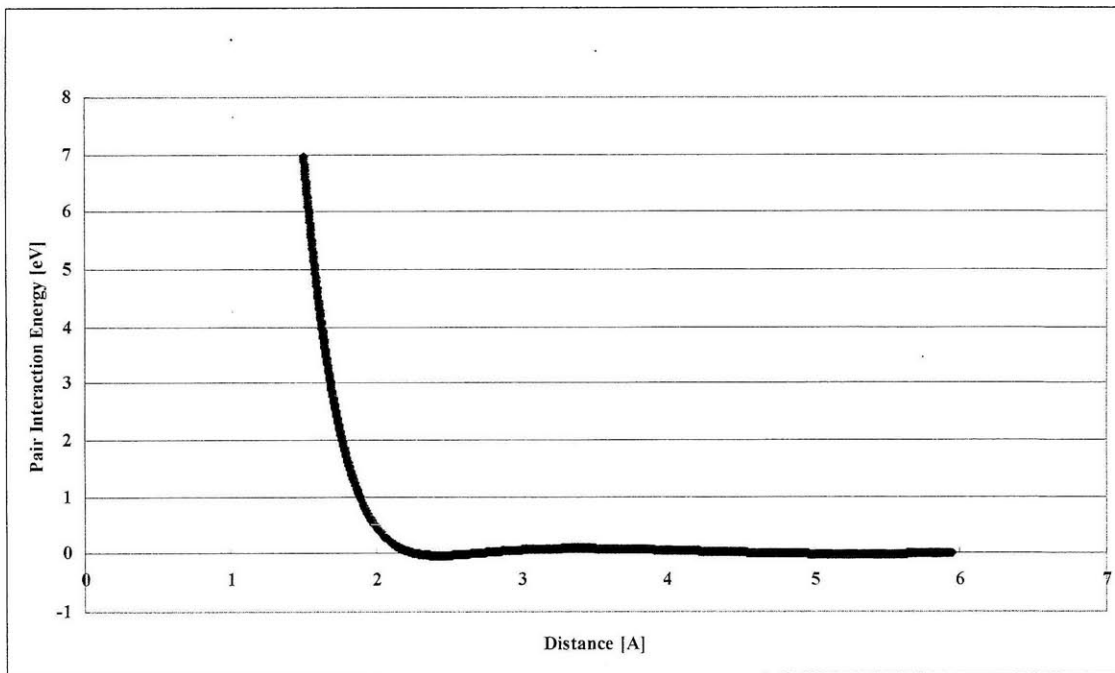


Figure 4-6 Pairwise interactions for Ni and Ni in EAM

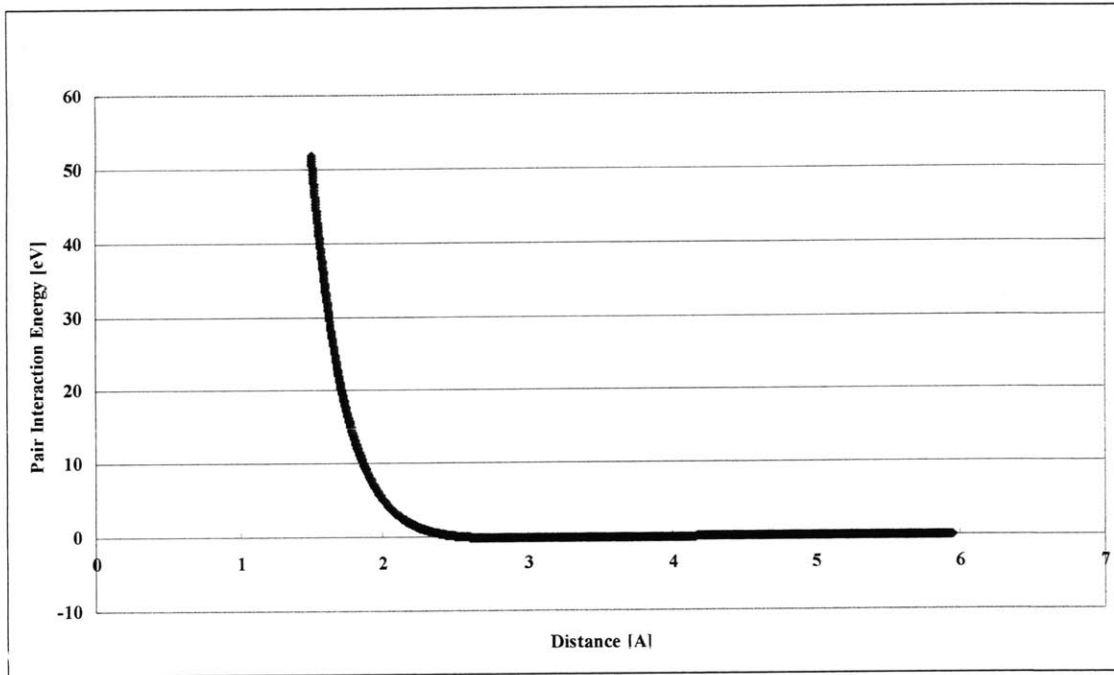


Figure 4-7 Pairwise interactions for Al and Al in EAM

#### 4.3. Lattice, thermal and mechanical properties from EAM potentials

In this section, important physical and chemical properties calculated using the constructed Mishin's EAM potentials for Ni and Al binary systems would be presented to verify the reliability of the chosen potentials for the atomistic and molecular dynamics simulations. All the properties addressed here are values at the equilibrium state with 300K and zero-pressure.

	Ni		Al		NiAl B2	
	EAM	Experiment	EAM	Experiment	EAM	Experiment
<b>C11 [Gpa]</b>	245	248	103	106	200	199
<b>C12 [Gpa]</b>	150	149	59	60	140	137
<b>C44 [Gpa]</b>	119	120	31	29	120	116
<b>Bulk Modulus</b>	177	181	79	76	160	158
<b>Lattice Constant [Å]</b>	3.453	3.52	4.052	4.05	2.859	2.88
<b>Cohesive Energy [eV]</b>	-4.501	-4.45	-3.362	-3.36	-4.465	-4.5

Figure 4-9 Mechanical properties for Ni, Al and NiAl for the current EAM [41, 51]

Phase	Structure	Experiment [41], [51]		EAM	
		Lattice Constant [Å]	Cohesive Energy [eV]	Lattice Constant [Å]	Cohesive Energy [eV]
Ni	A1	3.52	-4.45	3.453	-4.501
Ni3Al	L12	3.57	-4.57	3.525	-4.6
	D0 <sub>22</sub>			3.523	-4.591
	D0 <sub>3</sub>			2.812	-4.541
NiAl	B2	3.88	-4.5	2.859	-4.465
	B1			3.501	-3.988
	L10			3.662	-4.293
	L11			3.727	-4.082
	B32			2.939	-4.095
Ni3Al4		11.408		11.375	-4.283
Ni2Al3	D5 <sub>13</sub>	4.03	-4.38	4.016, 4.924	-4.189
NiAl3	D0 <sub>11</sub>	4.8	-4.02	4.950, 6.415, 7.484	-3.914
	L12			3.829	-3.805
	D0 <sub>2</sub>			3.047	-3.707
Al	A1	4.05	-3.36	4.052	-3.362

Figure 4-10 Lattice and energy properties for various phases of Ni and Al system for the current EAM

In the figure, the current EAM has a broad range for the various phases of Ni and Al system in terms of mechanical properties, and energetic properties. There are many EAM potentials for Ni and Al hybrid systems. Among them, the currently chosen EAM potential has the broadest range for Ni and Al systems. As a result, we chose this potential for the main potential to be used through the atomistic simulations in this research.

Also, for the intermixing of Ni and Al systems, the melting of Ni and Al play a key role in high temperature. Usually, most classical potentials are not good for the melting properties of the systems. In the current potential, predicted melting point at the equilibrium state is 990K and 1820K for Al and Ni respectively. In experiment, the melting point for the Al and Ni is 930K and 1750K at the bulk phases. Though those values don't agree to the experiments, the difference is under 50 – 100K and reasonable. Also, especially, for NiAl B2 phases, the melting temperature predicted with the current EAM is very well matched to the experiment. And, the current potential had been originally developed for high temperature applications for Ni and Al hybrid systems including phonon transport phenomena in the systems. This is one of the reasons we have chosen this potential for this project.

## Chapter 5

### Equilibrium molecular dynamics simulations

In atomistic and molecular dynamics simulations, initially constructed configurations with atomic positions and velocities are not on phase space in equilibrium for the given conditions such as temperature, pressure and volume. The initial atomic positions and velocities can be chosen in any random structure and distribution. However, for the computational efficiency, it is necessary to choose the initial configurations around the estimated equilibrium structures. For example, it is useful to choose the face-centered-cubic (FCC) lattice structures for the FCC metallic systems. However, even if the FCC structure is chosen as an initial configuration, the lattice distance should not be in the equilibrium value for the given conditions. In addition, for the finite temperature systems, the temperature is initially distributed using Gaussian distribution for the given finite temperature. However, without further equilibration processes, it does not guarantee the conserved temperature and lattice parameters. Therefore, we need to equilibrate the system in terms of the given conditions before applying additional loading conditions. In this chapter, the initially constructed nanoslab and nanosphere structures taken from FCC bulk systems are equilibrated for the given temperature and pressure (stress) using the Nose-Hoover canonical ensemble (NVT) and isothermal-isobaric ensemble (NPT) simulations for a very long time simulation, using the extended Hamiltonian methods discussed in Chapter 3. After the equilibration processes, the equilibrated structures are validated and tested by applying the microcanonical ensemble (NVE) simulations. If the systems are equilibrated well, the macroscopic thermodynamic properties such as temperature (kinetic energy), pressure tensors and potential energy fluctuate about the target values with small oscillations without any further thermostating and barostatting.

## 5.1. Equilibration of Ni and Al nanoslabs

In the next chapter, we analyze the impact simulations between the dissimilar Ni and Al nanoslabs. Before then, we need to find the equilibrium structures of Ni and Al nanoslabs for the given temperature (300K) and pressure (0 Pa). In addition, it is challenging to equilibrate the dissimilar phases of materials within one single simulation box. Thus, we need to equilibrate each single Ni and Al nanoslab in a different simulation box. Finally, we need to have two free surfaces in each Ni and Al nanoslab for the impact simulations, which requires consideration of unique boundary conditions.

### → Equilibration steps

- (1) Separate equilibration of Ni (or Al) nanoslab in NPT ( $T = 300\text{K}$ ,  $P = 0\text{ Pa}$ ) is performed with periodic boundary conditions in bulk
- (2) Equilibration of Ni and Al composite systems taken from (1) is performed in NVT ( $T = 300\text{K}$ ) with periodic boundary conditions in Y and Z directions only; Here, the lateral direction size of the composite system is determined as an average of the lateral sizes of Ni and Al nanoslabs in (1). This issue requires consideration of lattice mismatch between the two metals, as discussed in the following sections.

### 5.1.1. Equilibration of Ni bulk

#### (1) Initial structure

Lattice distance:  $3.45 \text{ \AA}$  (chosen to be close to the lattice distance for the current EAM potential)

X size:  $69 \text{ \AA}$  (20 unit cells)

Y size:  $93.15 \text{ \AA}$  (27 unit cells)

Z size:  $93.15 \text{ \AA}$  (27 unit cells)

Number of atoms: 58320 atoms

Integration time step: 1 fsec



(2) Temperature and pressure evolution during equilibration

For the NPT simulations of the initial structure, we elevate the temperature from 0K to 300K by adding 30K every 100000 steps with the fixed external pressure, 0 Par.

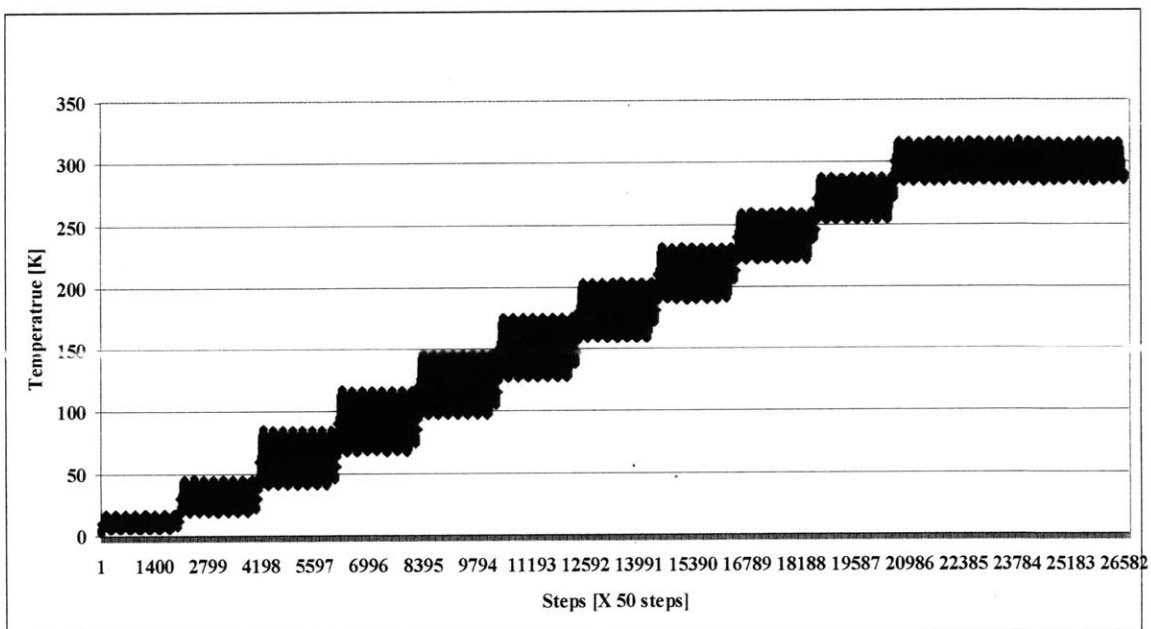


Figure 5-1 Temperature evolution of Ni nanoslab from 0K to 300K

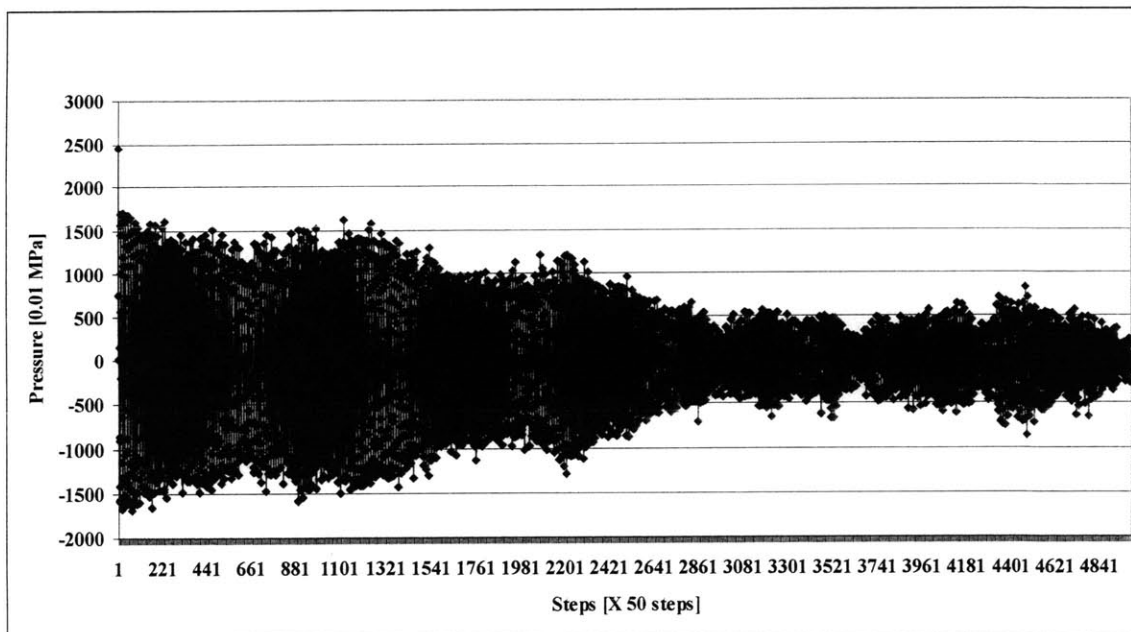


Figure 5-2 Global pressure evolution of Ni nanoslab at 300K

By thermostating during the simulations, we clearly observe temperature fluctuations around the desired temperature. Also, for the final target temperature (300K), the system is equilibrated during 300000 time steps (300 psec). In addition, the global pressure fluctuates around the target temperature (0 Bar). However, the width of fluctuations becomes smaller as the equilibration time goes by. Finally, we obtain the stable pressure fluctuation within +/- 10MPa.

(3) Final equilibrium values for 300K and 0 MPa for Ni bulk

Lattice parameter: 3.461 Å

Potential energy per atom: 4.45 eV

For the final structures, we observe the global axial stresses and shear stresses are relaxed around +/- 5 MPa.

### 5.1.2. Equilibration of Al bulk

(1) Initial structure

Lattice distance: 4.05 Å (chosen to be close to the lattice distance for the current EAM potential)

X size: 101.25 Å (25 unit cells)

Y size: 93.15 Å (23 unit cells)

Z size: 93.15 Å (23 unit cells)

Number of atoms: 52900 atoms

Integration time step: 1 fsec

(2) Temperature and pressure evolutions during equilibration

For the NPT simulations of the initial structure, we elevate the temperature from 0K to 300K by adding 30K every 100000 steps with the fixed external pressure, 0 Pa.

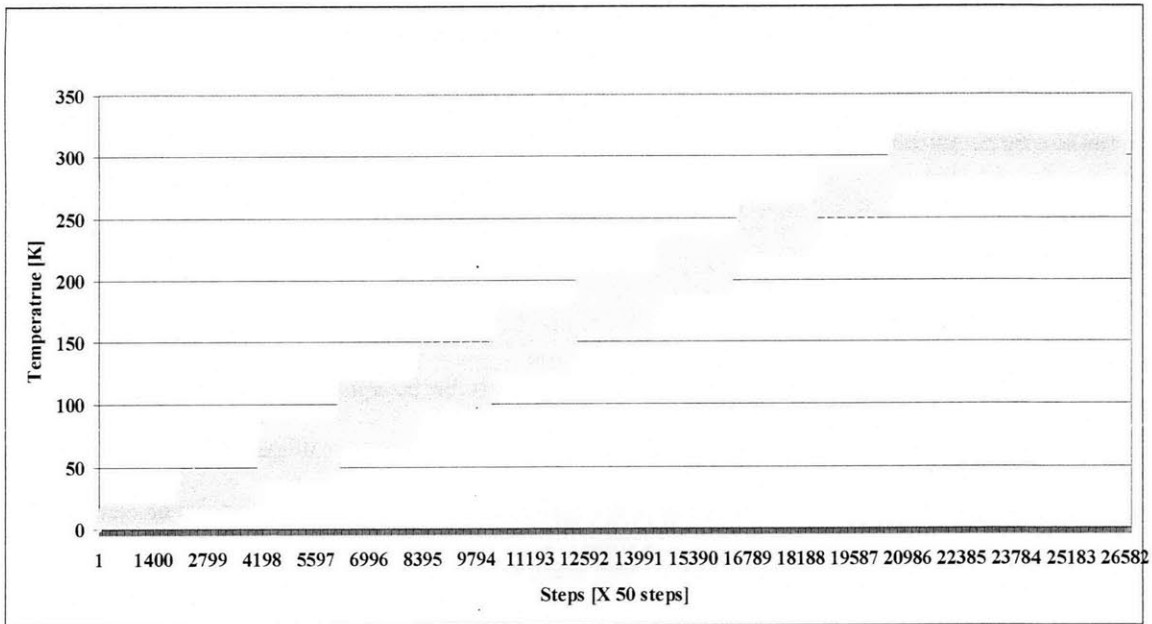


Figure 5-3 Temperature evolution of the Al nanoslab from 0K to 300K

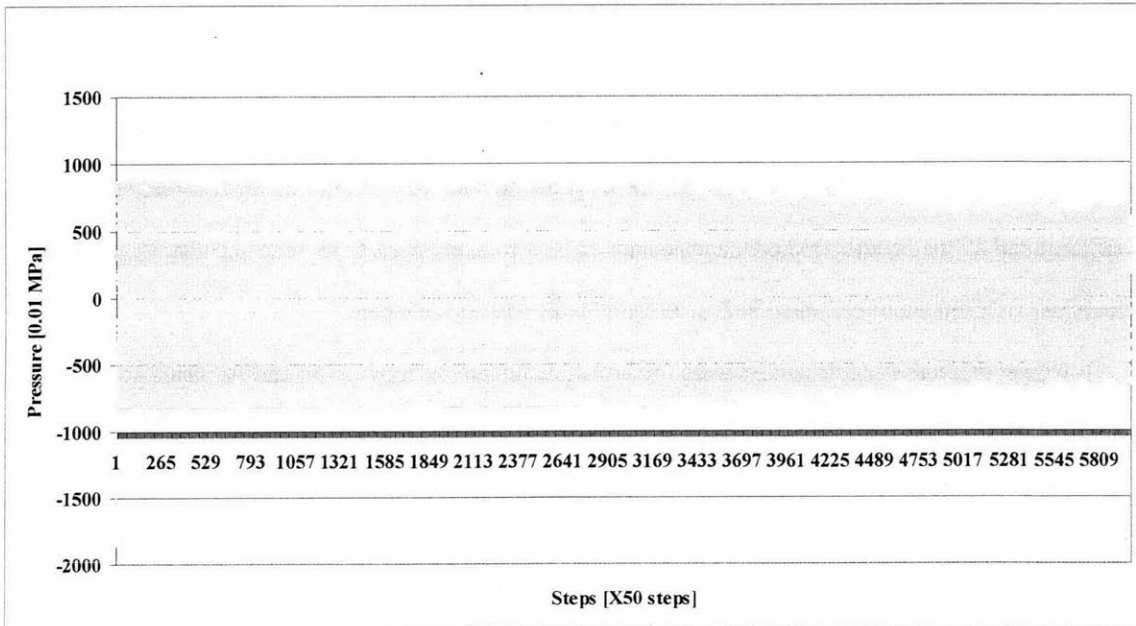


Figure 5-4 Global pressure evolution of the Al nanoslab at 300K

In Fig. 5-4, the global pressure fluctuates around the target temperature (0Pa). However, the width of fluctuation becomes smaller as the equilibration time goes by. Finally, we obtain the pressure fluctuation within +/- 5 MPa.

(3) Final equilibrium values for 300K and 0 MPa for Al bulk

Lattice distance:  $4.07 \text{ \AA}$

Potential energy per atom: 3.43 eV

For the final structures, we also observe the global axial stresses and shear stresses are relaxed around  $\pm 5$  MPa.

Finally, we performed the long-time NVE simulations for the equilibrated Ni and Al nanoslabs with the periodic boundary conditions for validating equilibration processes. Without any further barostatting and thermostatting, we clearly observe the desired temperature and pressure to be conserved during the NVE simulations. In addition, there are still small fluctuations around the desired values.

### 5.1.3. Equilibration of Ni and Al nanoslabs with free surfaces

For the impact simulations between two nanoslabs, we need to have Ni and Al nanoslabs with two free surfaces. To create the surfaces, we turn off the periodic boundary conditions for X direction (impact direction) and allow the equilibrated the nanoslabs in bulk to expand freely to vacuum. That is, for starting structures, we used the equilibrated bulk structures in the previous chapter.

However, there is an important issue on the lattice mismatch between Ni and Al nanoslabs. That is, we will put the nanoslabs into one single simulation box for the impact simulations. Thus, the size of Y and Z (perpendicular to impact direction) must be different for Ni and Al nanoslabs. However, within a single rectangular simulation box, this cannot be solved with the box size and shape. Usually, for a composite systems of dissimilar material phases, the composite system is equilibrated for global stress/strain minimizations. That is, the system is relaxed with the desired global stress (pressure) and temperature in the simulation box. By this equilibration process, one single phase is in a slight compression state and other phase is in a slight tension state. Thus, in this research, we took the average lateral sizes of the bulk Ni and Al nanoslabs equilibrated separately. That is, in the initial stage of the equilibration in bulk, we chose Y and Z sizes of Ni and Al nanoslabs in  $93.15 \text{ \AA}$ . (27 unit cells for Ni and 23 unit cells for Al). Finally, after the separate equilibration processes, we obtained Y and Z sizes for Ni and Al nanoslabs in  $93.50 \text{ \AA}$  and

93.67 Å respectively. These values are also time-averaged over the long-timescale NPT equilibrations for the bulk nanoslabs. Finally, we chose Y and Z sizes of the single simulation box for the final equilibration in 93.57 Å. By taking an average value, Ni nanoslab is under a slight tension and the Al nanoslab is under a slight compression state in lateral Y and Z directions. In terms of global strain, Ni slab is under  $\varepsilon = 0.0007$  (tension) and Al slab is under  $\varepsilon = -0.0007$  (compression) in Y and Z directions.

Finally, after equilibration of Ni and Al composite system in the fixed Y and Z size at 300K temperature, the Ni slab is under 5 MPa tension, and the Al slab is under 8 MPa compression, globally.

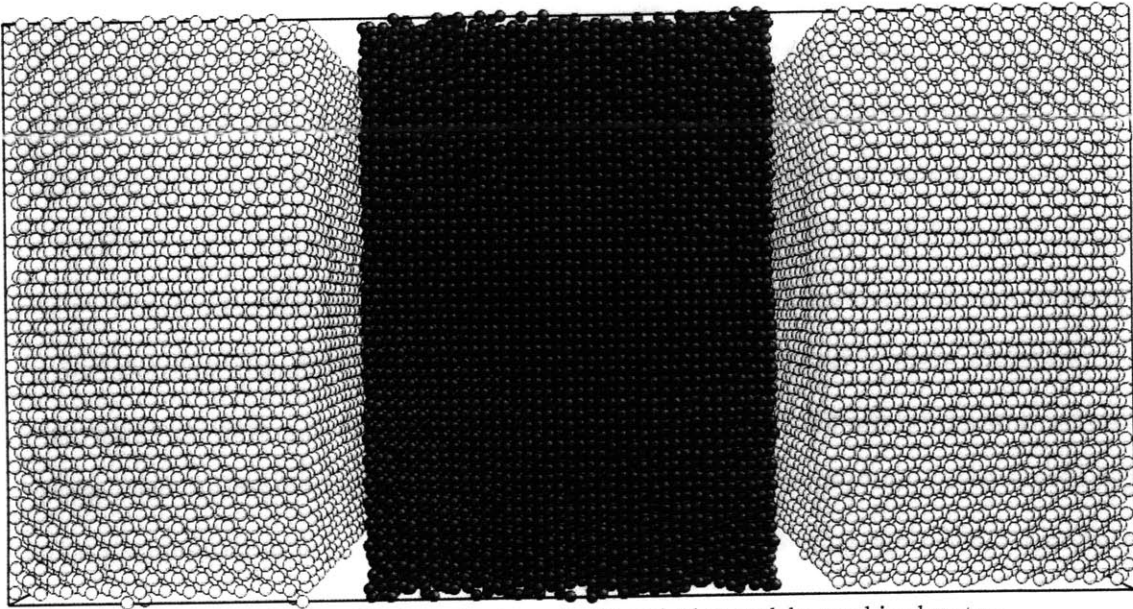


Figure 5-5 Final equilibrium structures for Ni and Al nanoslabs combined system

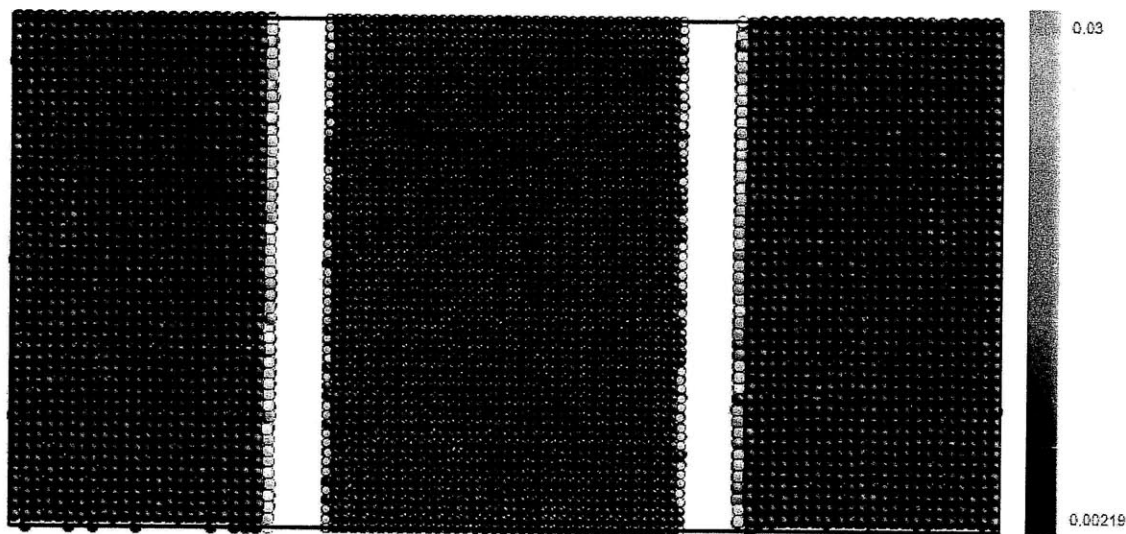


Figure 5-6 Von Mises strain invariants for the final structures

Also, for the final equilibrated nanoslabs, we checked the Von Mises shear strain invariants. As seen in the figure, shear strain over the slabs is around 0.002. However, we observe shear strain at surfaces. This is not the realistic Von Mises strain invariant at surface because the Von Mises strain invariant is calculated based on the current atomic configuration and coordination number without any relative atomic configuration for strain measurement. Therefore, the Von Mises strain invariant at surface appears high because the atomic coordination number at surface is smaller than one at bulk. In addition, we can relax normal stresses at the surfaces with the equilibration processes by surface reconstruction, but we cannot eliminate the lateral components of the stress tensors with the fixed lateral size. In experiments for metal surfaces, large surface stresses have been observed [52].

## 5.2. Equilibration of Ni and Al nanoparticles: the finite systems

In this section, we address the equilibration processes for finite systems of Ni and Al nanoparticles. Unlike the bulk systems, we cannot apply barostatting for the pressure equilibration in the finite systems. That is, to realize the finite pressure in the finite systems, we need to fill the simulation box with inert gases surrounding the sample. In this research, we investigate the behavior of Ni and Al nanoslabs and nanoparticles within the zero pressure (vacuum) state. For metallic systems, the atmospheric pressure (0.1 MPa) do not significantly affect the dynamic properties of the system. In the vacuum assumption, the equilibration for the finite systems is not quite complicated. That is, for a given temperature, the same

thermostatting methods can be applied during the equilibration. For pressure relaxation to vacuum, we make the finite system to expand freely to the vacuum. However, during the equilibration processes, the shape of the finite systems should be conserved. Thus, for the equilibration of Ni and Al nanoparticles with large surfaces, we adopted the NhT ensembles (constant temperature, number of atoms, and shape). Also, we chose the number of atoms in Ni and Al nanoparticles to be almost same in the previous nanoslab cases.

Finally, the Ni and Al nanoparticles are equilibrated in separate simulation boxes, and then combined in one single simulation box for the impact simulations, that is discussed in the following chapters.

### **5.2.1. Unexpected finite rotation during the equilibration**

By thermostatting during the equilibration processes, the system's dynamics are periodically perturbed disturbed by thermostatting disturbance. By the disturbances, the system can move rigidly within the simulation box. To remove the unexpected motions during the equilibration, we need to confine the system in translation and rotation. That is, the center of mass of the system must be fixed during the equilibration. By this process, there is no change in the global dynamics. However, for the finite rotation during the equilibration, it is not easy to eliminate angular velocities of the systems. Also, removing angular velocities artificially is not realistic in terms of the system's dynamics, even if the magnitude is extremely small. Thus, to prevent finite rotations of the system, we applied one constraint to the system during the equilibration simulations.

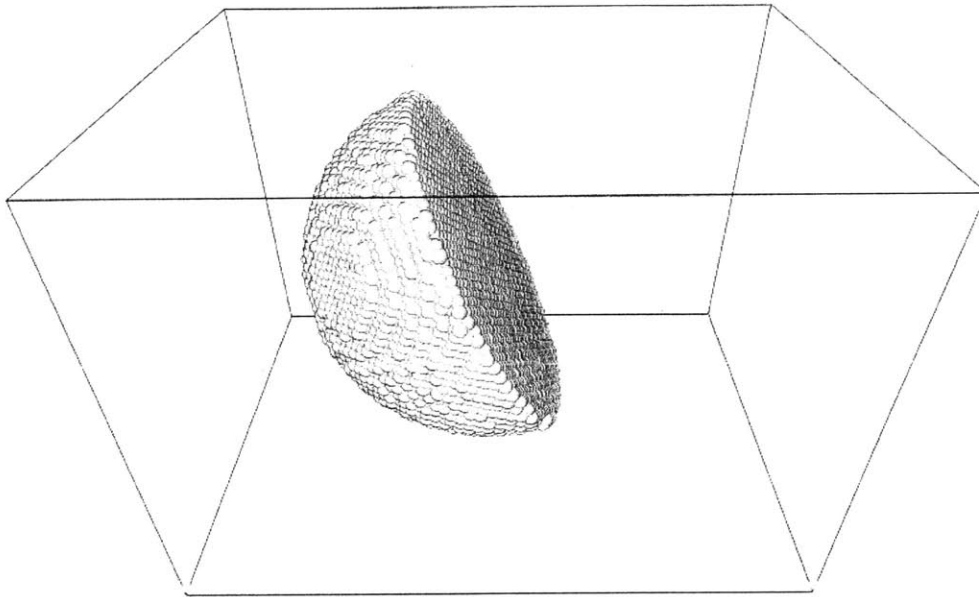


Figure 5-7 Finite rotation during NhT ensemble equilibration

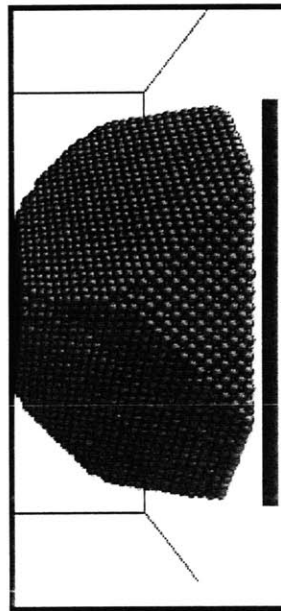


Figure 5-8 NhT ensemble equilibration with the reflective boundaries

We impose one free surface to the reflective momentum mirroring boundary. By imposing the reflective boundary, we eliminate finite rotations during the equilibration processes. This approach is more realistic than directly removing angular velocities during the equilibration process. However during the



constant temperature equilibration from 0 to 300K, there is thermal expansion. This means that, when determining distance between the atoms in the free surface and the reflective boundary, it should be located far enough away so as not to restrict the thermal expansion. For the current systems, we relocated the position of the reflective boundary to the right for every constant temperature case by carefully observing the thermal expansion.

### **5.2.2. Surface roughness in the nanoparticles**

Through processing of the nanoparticles, surface roughness can be controlled. That is, typically, the representative shape of the nanoparticles of face centered cubic (FCC) metals is one having large surface facets in room temperature. Also, in terms of minimum energy at the surfaces,  $\{1\ 1\ 0\}$  facet usually does not exist in the nanoparticle because it is of higher energy than the  $\{1\ 1\ 1\}$  and  $\{1\ 0\ 0\}$  surfaces. So, we took only  $\{1\ 0\ 0\}$  and  $\{1\ 1\ 1\}$  surfaces for the facet structures from the equilibrated structure. That is, for the faceted structure, we took the nanoparticle having octahedral radii from its center, from the equilibrated FCC bulk systems. This system was then equilibrated again in the NVT ensemble. At the same time, we consider a spherical nanoparticle of one radius, where the atomic islands (rough surfaces) on each facet are allowed. Usually, in studies of melting in finite nanoparticles of 10 nm diameter, nanoparticles with such rough surfaces have been considered the representative. We constructed two different shapes of nanoparticles (faceted and non-faceted with islands) having the same number of atoms, in order to investigate the role of the surface structures in Ni and Al nanoparticle impact processes in the following chapters.

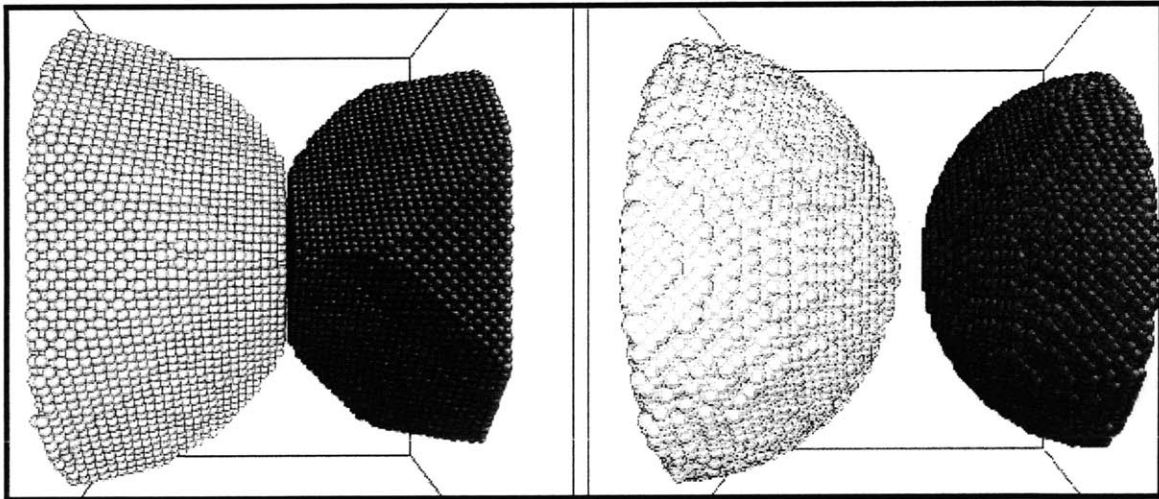


Figure 5-9 Figure 5-5 Final equilibrium structures for Ni (red) and Al (yellow) nanoparticles combined system (a) Faceted structure, (b) Non-faceted structure

Finally, with the equilibrated structures of Ni and Al nanoparticles, we observed fully relaxed states for global axial stresses and desired temperatures. Also, with the NVE simulations, we observed small fluctuations around the desired temperature (300K) and global pressure (0 MPa).

## Chapter 6

### **Non-equilibrium molecular dynamics simulations of Ni and Al nanoslab impacts**

Impact induced shock compression is defined as high speed longitudinal compressive waves much faster than the sound wave speed in the given media [20]. During shock compression loading, strain rate is extremely fast and usually greater than  $10^8$  [/sec]. Under extremely high strain rate loading, unexpected phenomena such as dislocation based plastic deformation, grain boundary sliding, and phase transformation have been observed in many experiments and atomistic simulations for the solid state systems [18-20]. Also, in manufacturing processes for intermetallic compounds for the Ni and Al systems, the shock induced reaction initiations have been observed in extremely high rigid flyer speeds over 1000 m/sec for the shock loading and high pressure over tens of GPa [28-29, 45]. However, those critical conditions for the shock induced reaction initiation depend strongly on initial composition and microstructure of the Ni and Al mixtures, and important details of physics and chemistry related to the shock induced reaction initiations still remain unknown. In this chapter, based on the equilibrated structures in the previous chapter on the equilibrium molecular dynamics simulations, the impact induced phenomena in Ni and Al nanoslabs is investigated. First, the impact loading process is described with the nonequilibrium molecular dynamics simulations via shrinking boundary techniques, and the dynamic properties such as shock temperature, normal shock stress, instantaneous Von Mises shear strain invariants, and shock front velocities during the impact loading are addressed. After the shock loading, the long time scale microcanonical ensemble simulations are applied to the system to investigate the resultant impact induced dynamic phenomena.

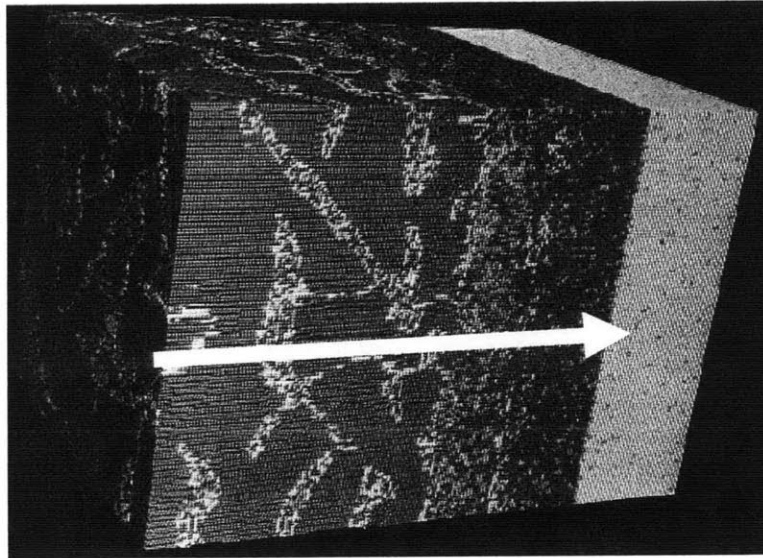


Figure 6-1 Shock induced phase transformation and plastic deformation process in Copper [20]

### **6.1. Shock compression loading within periodic boundary conditions (PBC) and simulation condition setup**

In atomistic and molecular dynamics simulations of general deformation processes in the solid states, quasi-static loading processes are assumed in the sample so as to be deformed homogeneously. That is, during the deformation processes of atomic structures, atomic coordinates are remapped homogeneously in an artificial manner. In addition, for the homogeneous displacement field, a linear velocity field is assigned to atomic structures in equilibrium ensembles such as NVE and NVT, and, consequently, the material is continuously strained. However, to describe highly dynamic process such as impact or shock loading, more advanced molecular dynamics simulation techniques should be implemented. Also, during such dynamic processes, it is unreasonable to estimate the dynamic properties of the system within an equilibrium ensemble approach.

There are several methods to create shock loading within the solid state systems. Even though the shock loading process is usually far from the equilibrium, many papers have proposed the shock loading procedures via the equilibrium analysis. That is, by taking very small time steps during the shock loading, the system could be assumed to be in equilibrium during each short time segment. However, in this kind of shock loading process, there are many physical flaws in the simulations. For this reason, in this research,

the nonequilibrium molecular dynamics simulations have been adopted to induce the shock loadings in the systems.

One of the most popular approaches, for this type of simulation, is to impose shock loading by the momentum mirroring method [1, 19-20]. In this method, a rigid piston hits one of the free surfaces of the target sample and, consequently, the sample is shocked from the free surface. During the impact between the sample and rigid piston, the perfect elastic collision is assumed. That is, if an atom in the sample collides with rigid piston, the reverse linear momentum, of opposite direction (called “mirroring”) to the incident direction and of same magnitude, is assigned to the atom again.

Another method is to create the shock from impact surfaces using shrinking boundary conditions within the periodic boundaries [53]. In this research, the shrinking boundary method was adopted to describe the impact loading processes because this was the final application of these simulation results is for the nanoslab (or nanoparticles) to be filled into a polymer nanocomposite and confined geometrically within the composite. After the previously detailed equilibration processes for Ni and Al combined system, the atoms are assigned the following initial velocities: atoms in the left Al slab 1 are assigned  $+u$  in the impact direction, superposed on their thermal velocities; the atoms in the right Al slab 2 are assigned an additional velocity equal to  $-u$ ; the remaining atoms in the central Ni slab are at rest. Also, in order to avoid unexpected shock release at the boundaries by a velocity discontinuity, the periodic boundaries in the impact direction are also assigned constant velocities which evolve with the impact velocities.

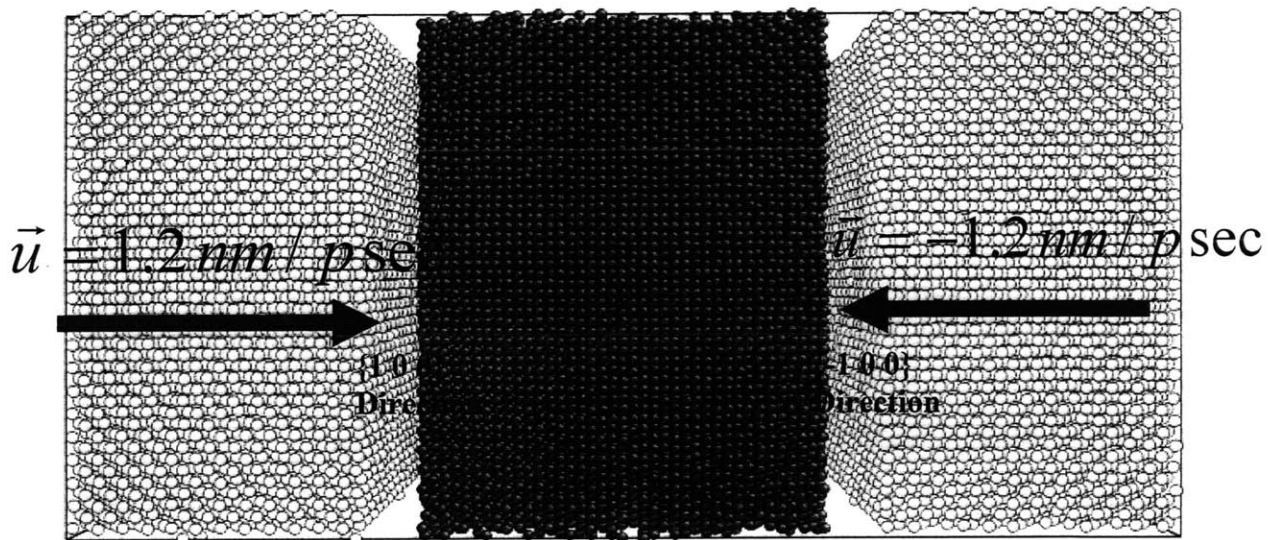


Figure 6-2 Initial simulation setup for Ni and Al nanoslab impact with 1200 m/s projectile speed (Red: Ni, Yellow: Al)

Upon collision of the two slabs, impact waves that are generated at the interfaces between Ni and Al slabs propagate in opposite directions, leaving behind compressed materia. Also, the impact waves propagate toward the center of Ni slab. In this simulation, geometric confinement is imposed on the system in order to capture the impact waves from Al slab to the Ni slab because we will ultimately intend to quantify the impact characteristics of Ni and Al slabs (or particles) as fillers in the polymer nanocomposites. As a result, we stop the shrinking boundary conditions and begin a microcanonical (NVE) equilibrium molecular dynamics simulation for a long timescale to observe the system dynamics after the two impact waves for the outward directions meet at the periodic boundaries. Also, there is a potential issue on the lateral relaxation during the impact loading. That is, in usual quasi-static loading, the lateral directions of the system should respond dynamically to the compressive or tensile loading with respect to the Poisson's ration in the elastic regions. However, in an extreme, high strain rate process such as shock loading, the lateral directions are assumed to have insufficient time to respond to the high-speed loading. Thus, for the simulations in this chapter, all the lateral directions have been fixed within the simulation box during the impact loading.

The simulation setup is listed as below:

### **Simulation setup and sizes**

Impact velocity: 100 - 1200 m/sec (0.02 - 0.234 eV/atom)

Initial temperature: 300K

Atoms in Ni slab: 50234

Atoms in Al Slab: 50278

In the following sections, the analysis for projectile velocity of 1200 m/s is discussed.

## **6.2. Evolutions of shock induced dynamic properties during the impact loading**

To calculate the local dynamics properties in the system, the system has been divided to 45 pieces of local bins over the longitudinal direction. All the local properties have been averaged within each local bin. Also, the size of the local bin has been also changed during the compression.

(1) Velocity (displacement) field

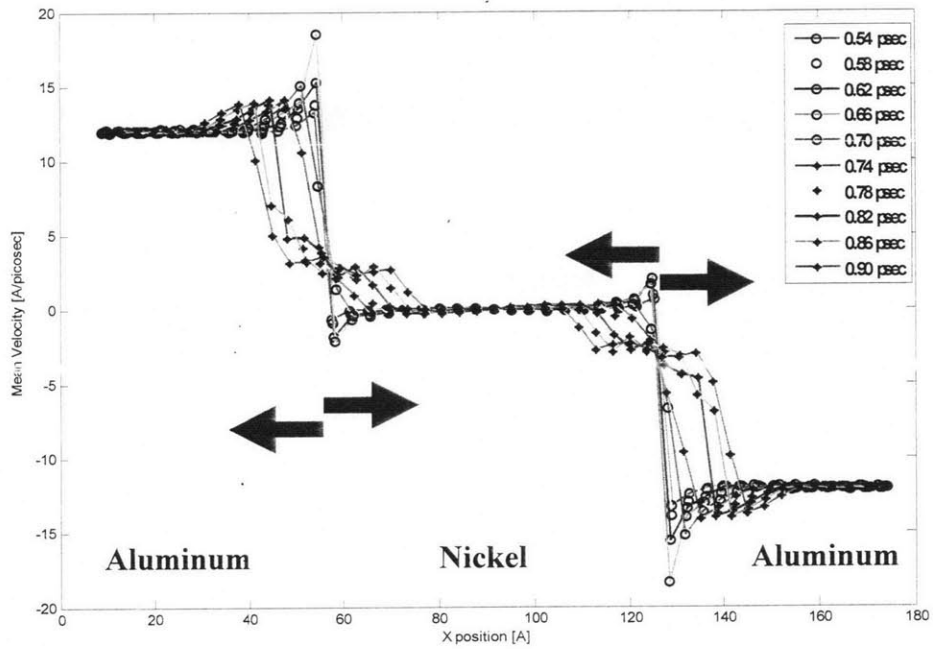


Figure 6-3 Velocity field evolution during impact loading from 0.54 psec to 0.90 psec

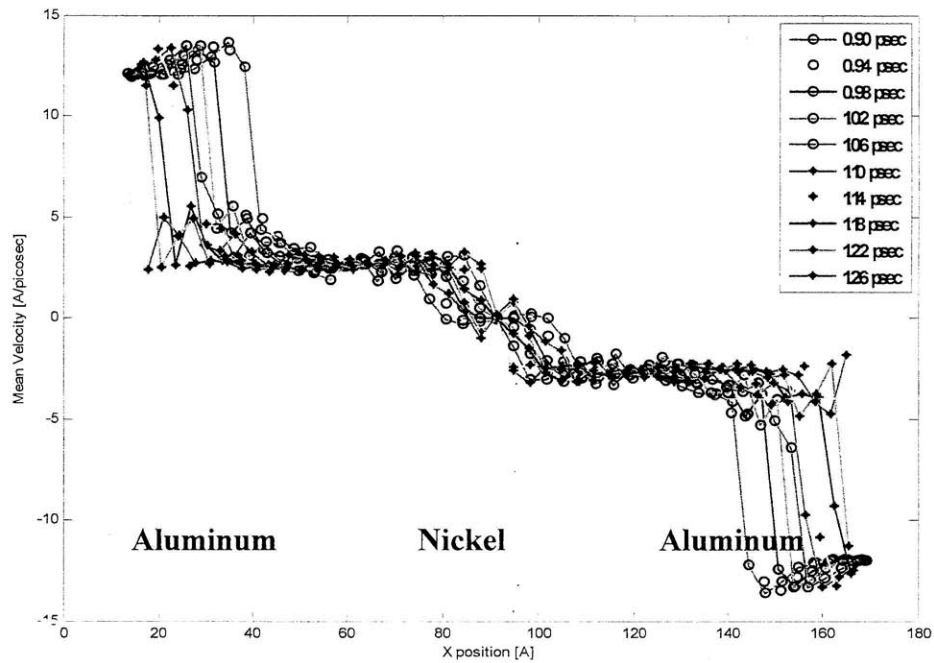


Figure 6-4 Velocity field evolution during impact from 0.90 psec to 1.26 psec

Initially, rigid velocity  $1.2 \text{ nm/psec}$  and  $-1.2 \text{ nm/psec}$  have been assigned to the atoms in the left and right aluminum slabs respectively upon the thermal velocity for 300K. After collisions at the interfaces between (1) Al slab-Ni slab and (2) Ni slab-Al slab, it is clear that shock compressive behavior has been created at the impact interfaces by the relative motions between atoms. The atoms behind the shock front are compressed by the different flow velocities in the systems, creating the compressive displacement fields from the impact interfaces. After 1.26 psec of the impact loading, the two shock waves meet at the periodic boundaries of the two aluminum slabs and, consequently, the shrinking boundaries are turned off.

## (2) Shock temperature

By the instantaneous shock compressions in the system, the local kinetic temperature called shock temperature increases from the compressed regions. In the classical molecular dynamics simulations, temperature is obtained by calculating kinetic energy of atoms in 3 degrees of freedom by the equipartition theory in the classical statistical mechanics. Also, in the impact loading process, the samples rigidly move by assigning initial velocities to Al slabs for the impact. Thus, it is required to remove the assigned rigid



kinetic energy part by subtracting local mean drift velocities from the total kinetic energy term in order to extract the thermal kinetic energy over the slab. In this simulation, we calculated the average linear momentum of the bins which, would be used as the local mean drift velocities of the local atoms in each bin.

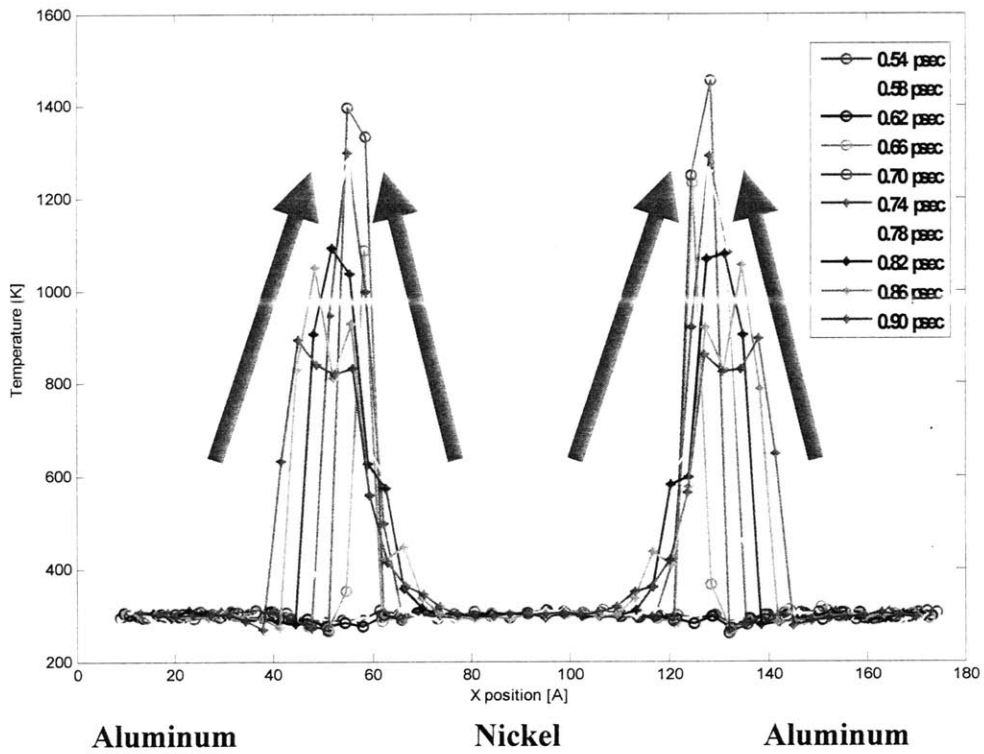


Figure 6-5 Instantaneous shock temperature evolution during impact from 0.54 psec to 0.90 psec

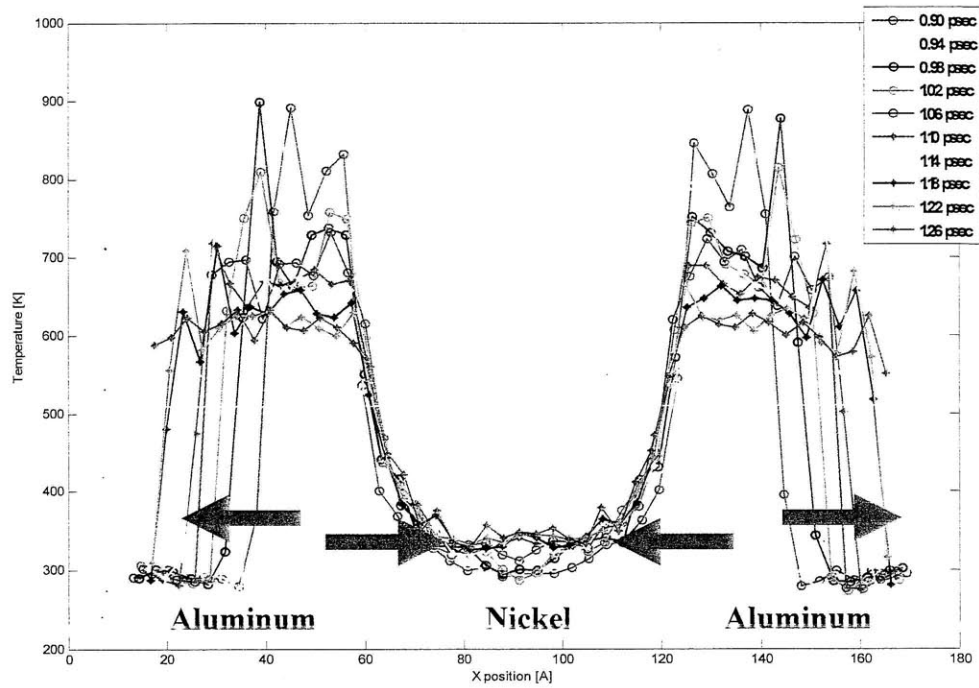


figure 6-6 Instantaneous shock temperature evolution during impact from 0.90 psec to 1.26 psec

Figure 6-6 Instantaneous shock temperature evolution during impact from 0.90 psec to 1.26 psec loading 2. As seen in Fig. 13 and 14, temperature becomes extremely high instantaneously when the Al slab heats the Ni slab by the impact. The shock heating is propagating away from the impact interfaces towards opposite directions over the slabs, along with the shock compressive wave propagations.

### (3) Normal shock stress

In classical molecular dynamics simulations, atomistic stress tensors could be obtained using the virial theorem in the classical mechanics. Here, we compared another local shock stress form proposed by Hardy. It has been proven that there is no significant difference between the virial stress and Hardy's shock stress in the shock loading processes.

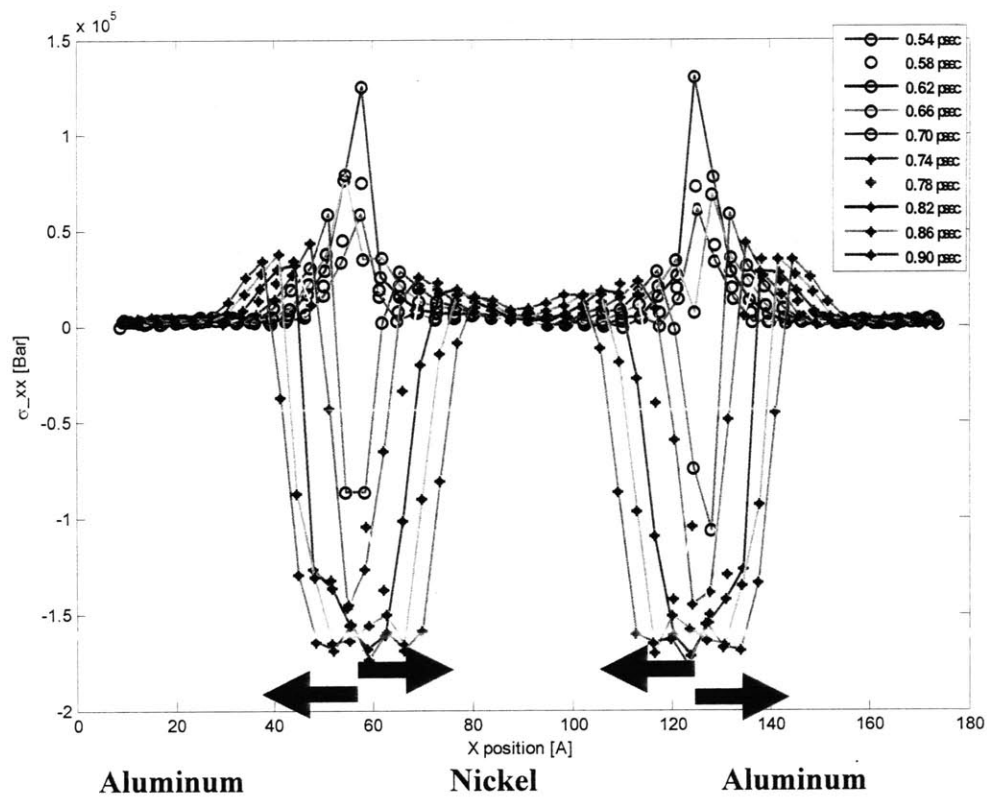


Figure 6-7 Instantaneous normal shock stress evolution during impact from 0.54 psec to 0.90 psec

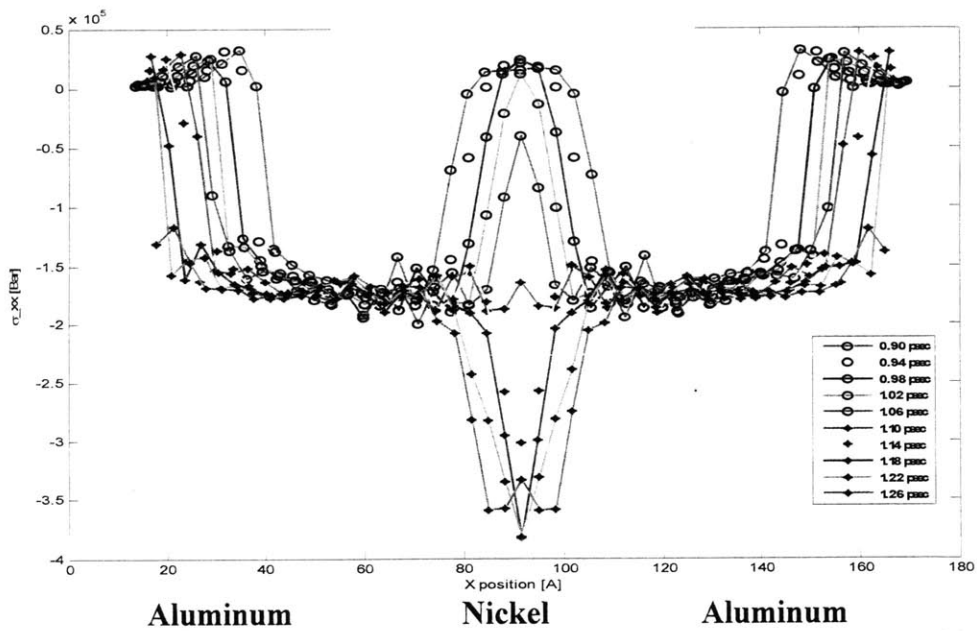


Figure 6-8 Instantaneous normal shock stress evolution during impact from 0.90 psec to 1.26 psec

As seen in Fig. 6-8, when the two compressive waves meet at the periodic boundary of the Al slab, the whole system is compressed completely.

(4) Potential energy per atom

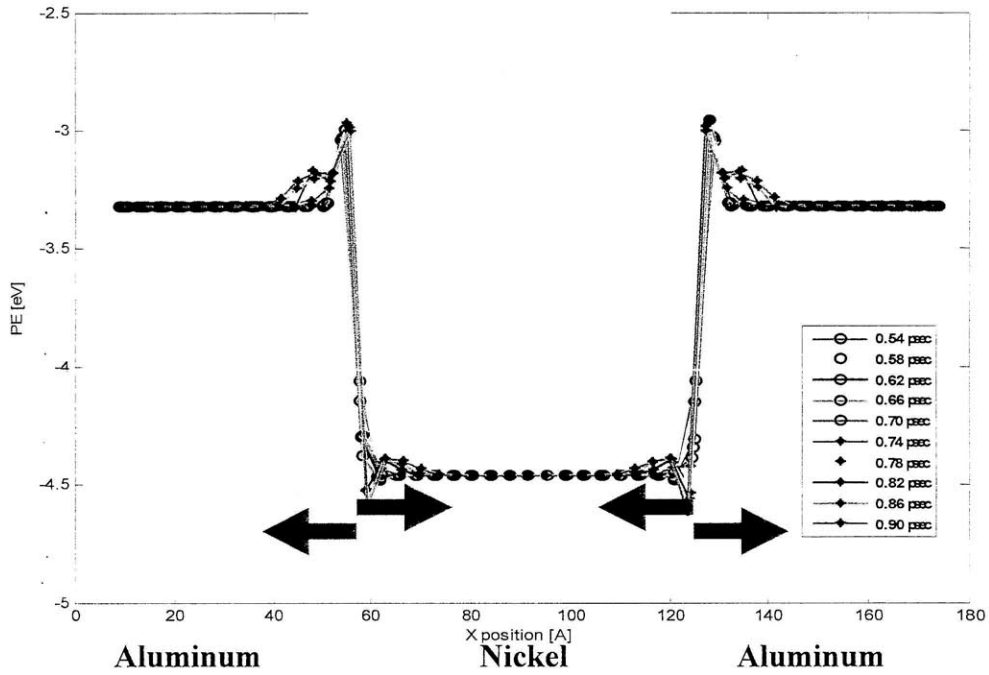


Figure 6-9 Average potential energy evolution during impact from 0.54 psec to 0.90 psec

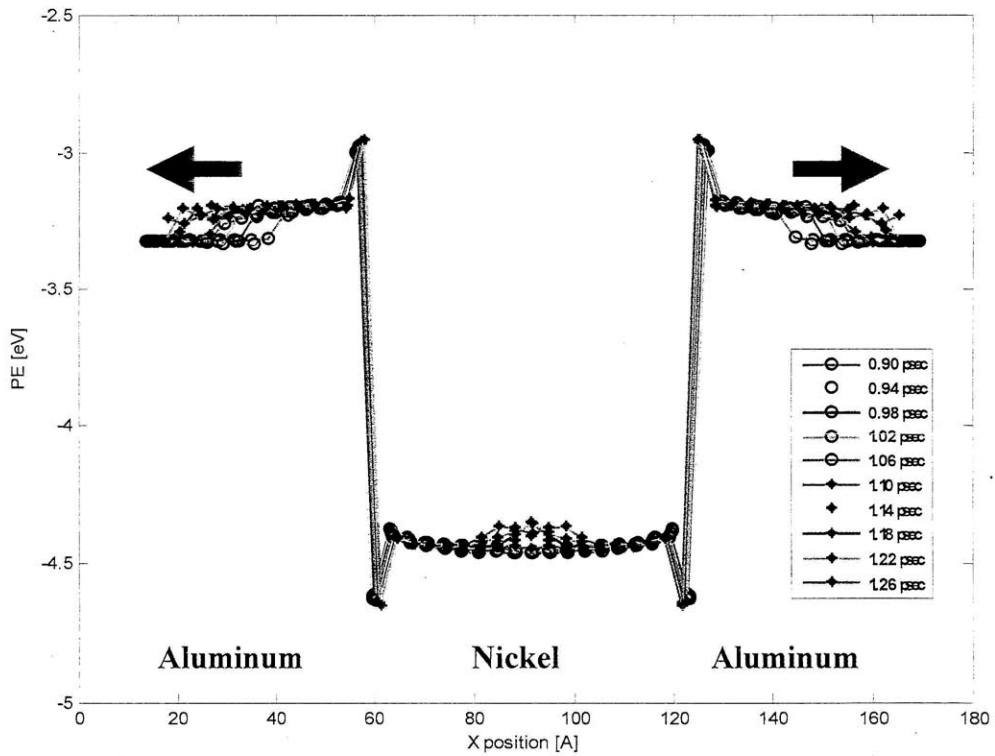


Figure 6-10 Average potential energy evolution during impact from 0.90 psec to 1.26 psec

(5) Total kinetic and potential evolutions

To validate the shrinking boundary technique for impact loading, we need to check the total kinetic energy (thermal and rigid), and potential energy evolutions during the impact process. It is clear that the total kinetic energy including the applied rigid velocities must be minimized when the system is compressed completely by the impact. Also, at the same time, the total potential energy of the system must be locally maximized because the potential energy of each atom increases under compression of the slab.

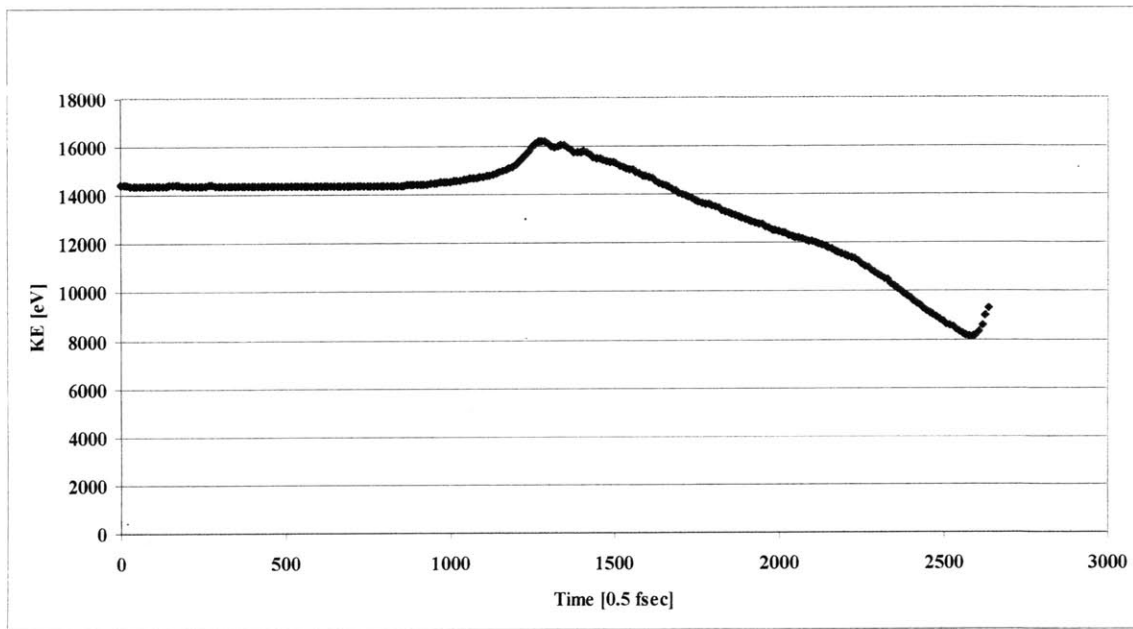


Figure 6-11 Total kinetic energy evolution during impact loading

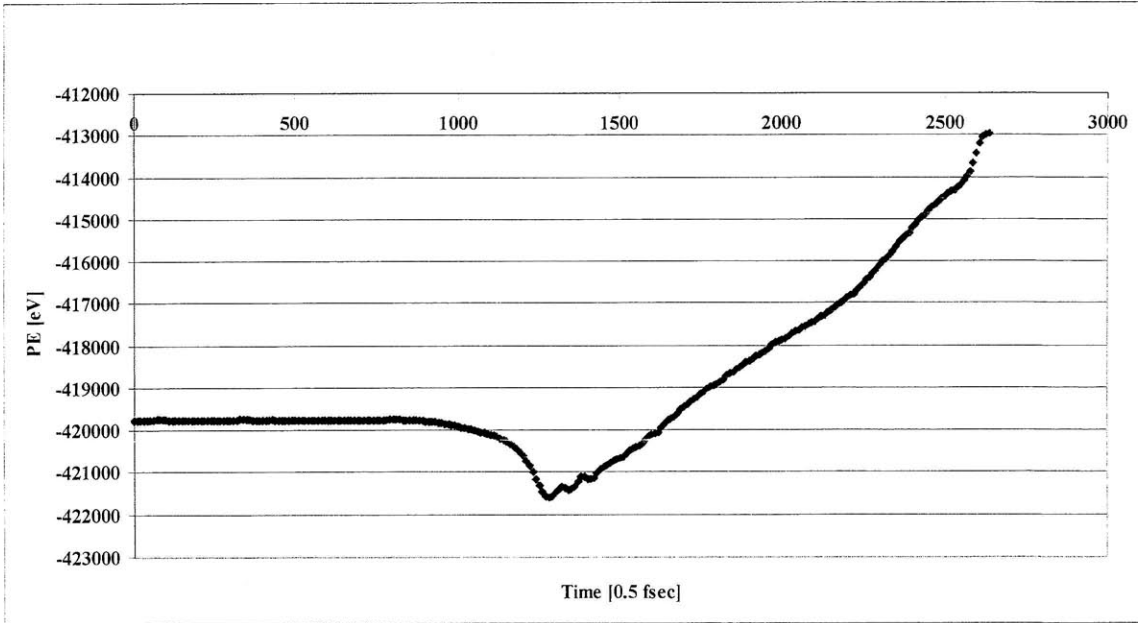


Figure 6-12 Total potential kinetic energy evolution during impact loading

As seen in Fig. 6-11, 12, the total kinetic energy is at a minimum, and the total potential energy is at a maximum at the completion of the compression (around 1.26 psec)

(6) Von Mises shear strain invariants

Unlike the instantaneous shock stresses calculated from the virial theorem, instantaneous properties related to the atomic positions are physically meaningful even in non-equilibrium processes. Here, the localized Von Mises shear strain invariants are shown during the impact loading (Fig. 6-13).

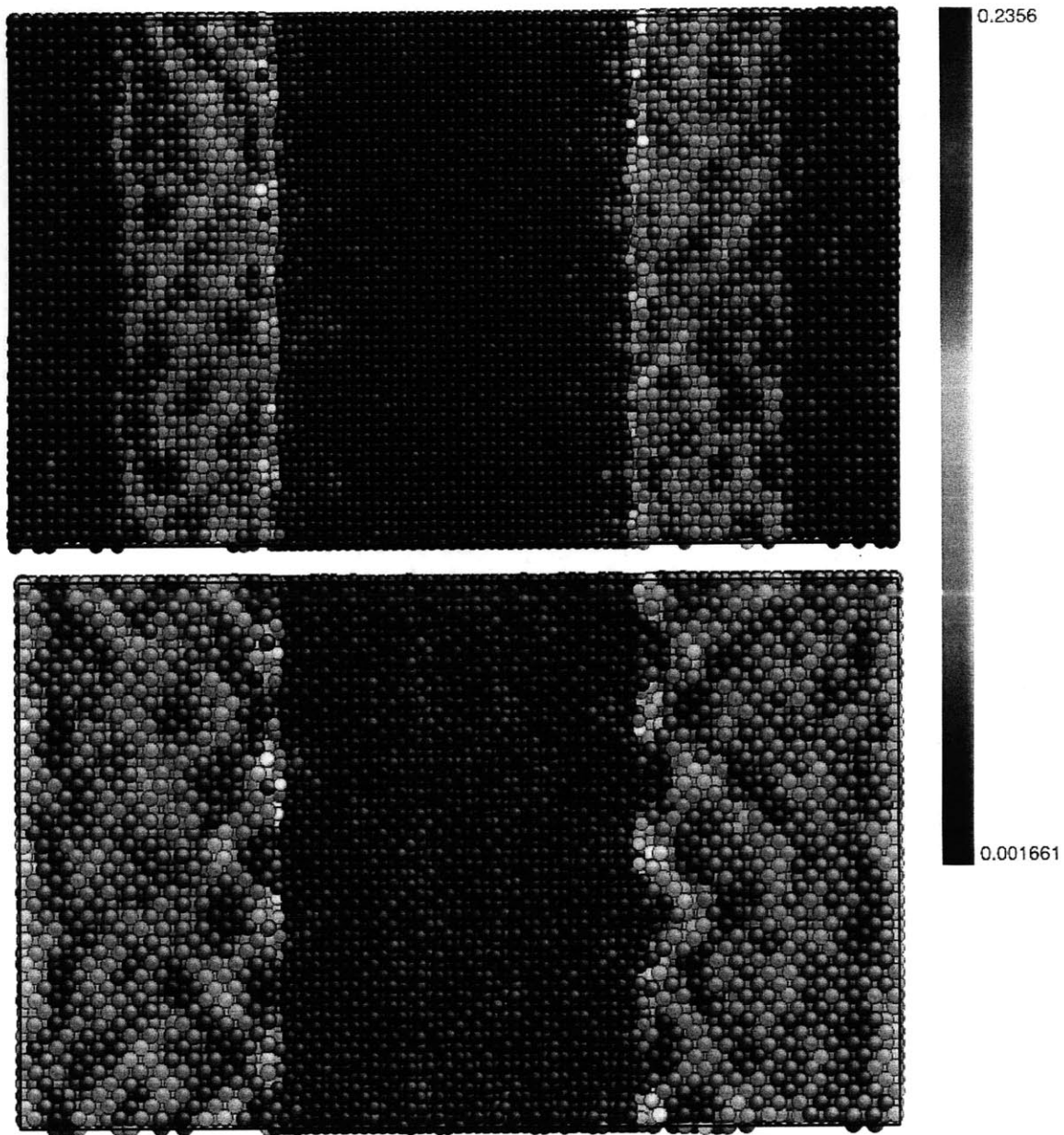


Figure 6-13 Von Mises strain evolution at (a) 1.5 psec (b) end of impact loading

As seen in Fig. 6-13, the localized shear strain invariants are also propagating with the shock waves from the impact surfaces toward the perimeter of the slab. Also, the shear strain in the Al slabs is much greater than the shear strain in the Ni slab. This means that aluminum is much less resistant under shock loading as is expected from its comparatively lower elastic constants and yield stress.

### 6.3. Evolutions of kinetic and potential energies after the impact loading: long time scale phenomena

After impact loading under shrinking boundary conditions, we performed the NVE (microcanonical ensembles) MD simulations to investigate the long timescale evolutions of the system. In the simulations, the time step is still 1 fsec and the total simulation time is 5 nsec (5 million time steps).

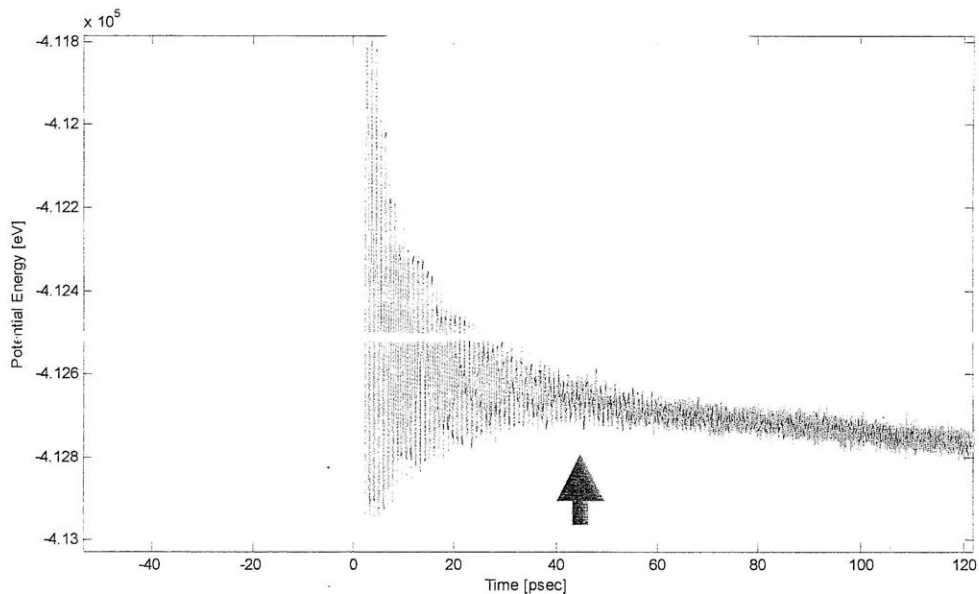


Figure 6-14 Total potential energy fluctuation in the initial stage of NVE simulation (Red arrow means the end of fluctuation by shock loading)

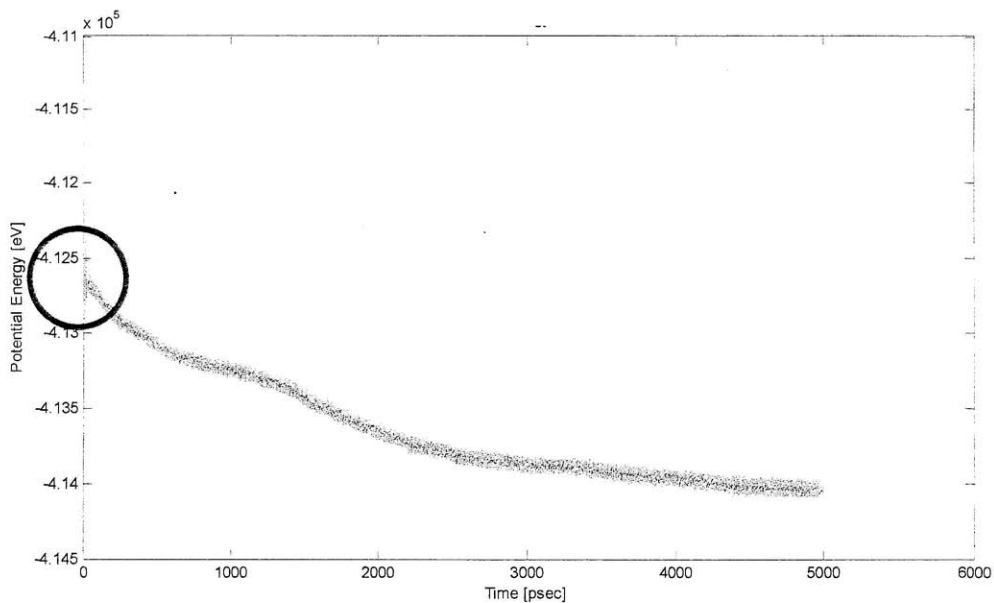




Figure 6-15 Total potential energy fluctuation in the long time scale NVE simulation

After completion of the impact loading, the pressure waves in the system go back and forth and, consequently, the total potential energy of the system fluctuates and stabilizes around 45 psec after the NVE simulation starts. However, after the fluctuation is stabilized, the total potential energy decreases again very slowly until 4.5 nsec, and goes to plateaus around 4.5 – 5 nsec.

Also, the temperature of Nickel and Aluminum slabs (i.e. kinetic energy) fluctuates during the shock wave relaxation at the initial stage of the NVE simulations after the impact loading completion. As the total potential energy decreases after 45 psec, the temperature of the Ni and Al slabs increase very slowly until 4.5 nsec, and goes to plateau around 4.5 – 5 nsec. Finally, the whole system is fully equilibrated to 750 K around 4.5 – 5 nsec during the NVE simulation.

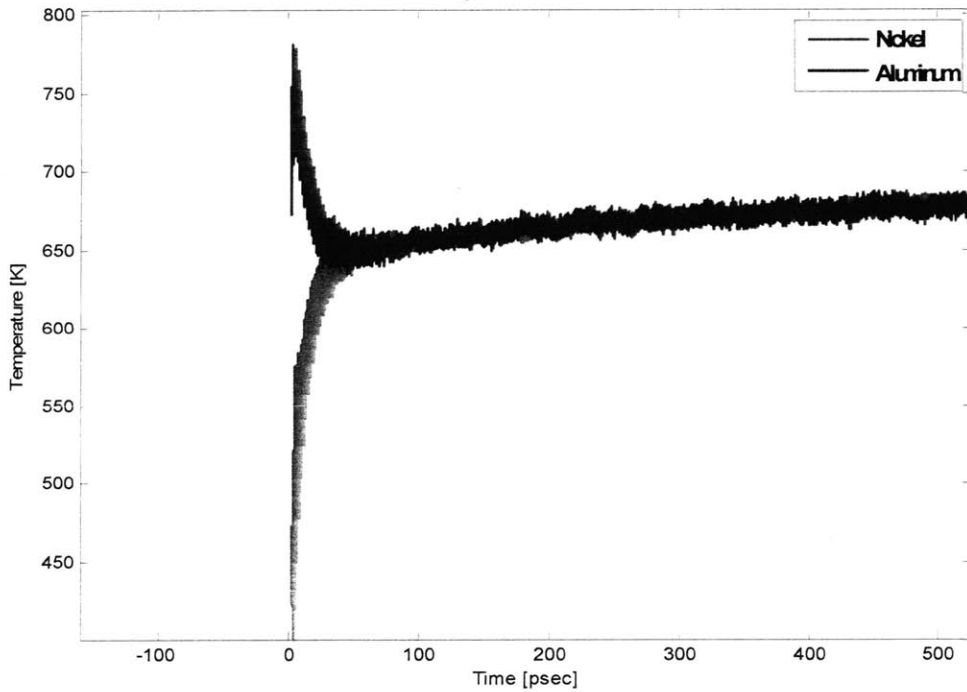


Figure 6-16 Temperature fluctuation in the initial stage of NVE simulation

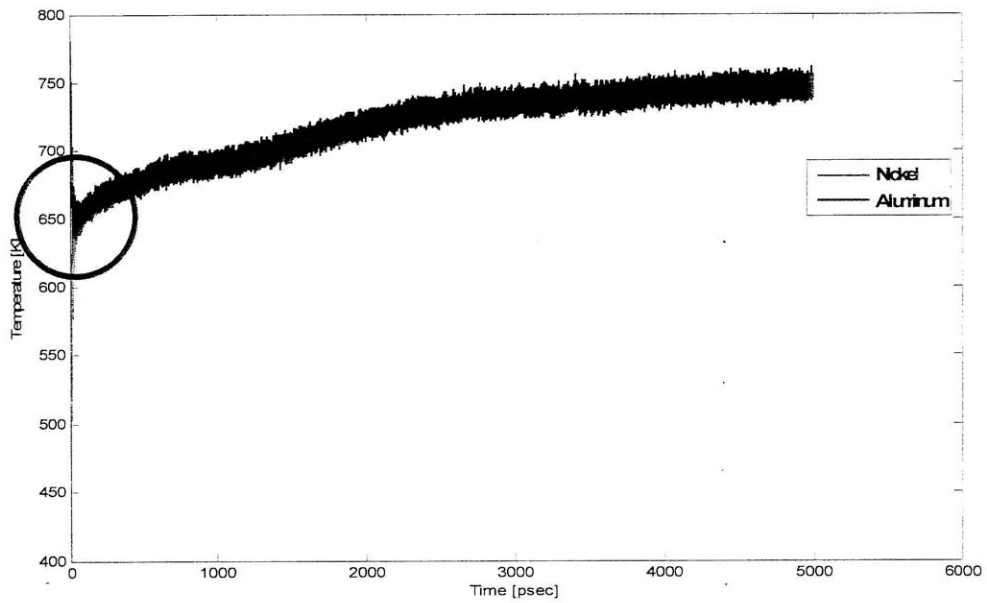


Figure 6-17 Temperature evolution in the long time scale NVE simulation

In Fig. 6-15 and 6-16, after 45 psec (shock loading stabilization time), we can clearly observe very slow and continuous exothermic phenomena until an equilibration time, 5 nsec. This exothermic reaction can be attributed to the slow intermixing between Ni and Al atoms at the interfaces, forming the new Ni and Al bonds. This intermixing process can be found in Fig. 6-19 and 6-20.

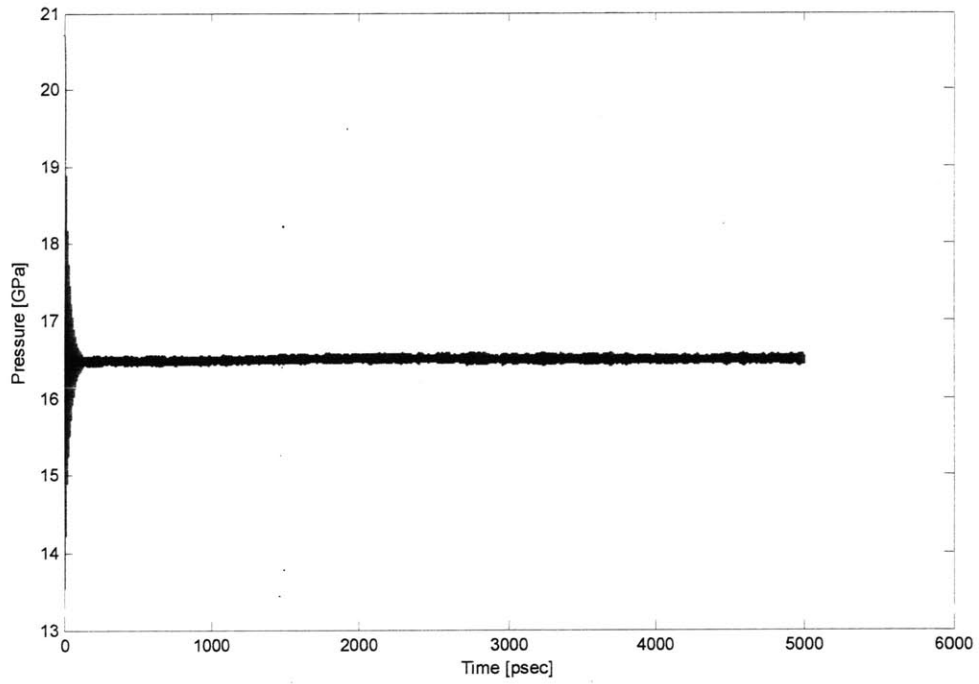


Figure 6-18 Global pressure evolution in the long time scale NVE simulation

Also, the total thermodynamic pressure of the system fluctuates until 45 psec during the shock wave relaxations, and stabilizes around 16.5 GPa.

Based on the results on the total potential energy and kinetic energy evolutions during the long timescale NVE simulation, we conclude there is change in the phase of the system after the relaxation of the shock waves. To quantify this, we will next analyze the atomic evolutions during the simulation.

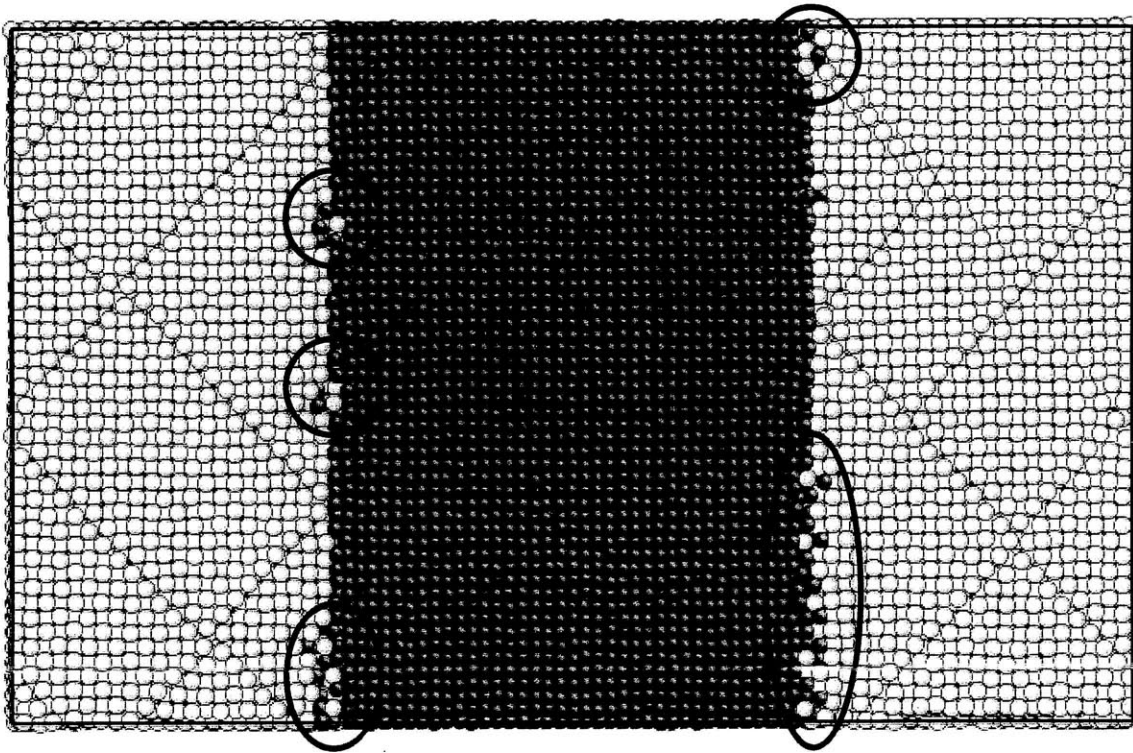


Figure 6-19 Atomic configuration at 4.5 psec after impact loading

As seen in Fig. 6-19, at 4.5 psec, intermixing between Ni and Al atoms is initiated locally. The intermixing between Ni and Al atoms propagates through the slabs very slowly and ceases around 4.5 – 5 nsec creating the clear intermetallic state in the Ni and Al interfaces.

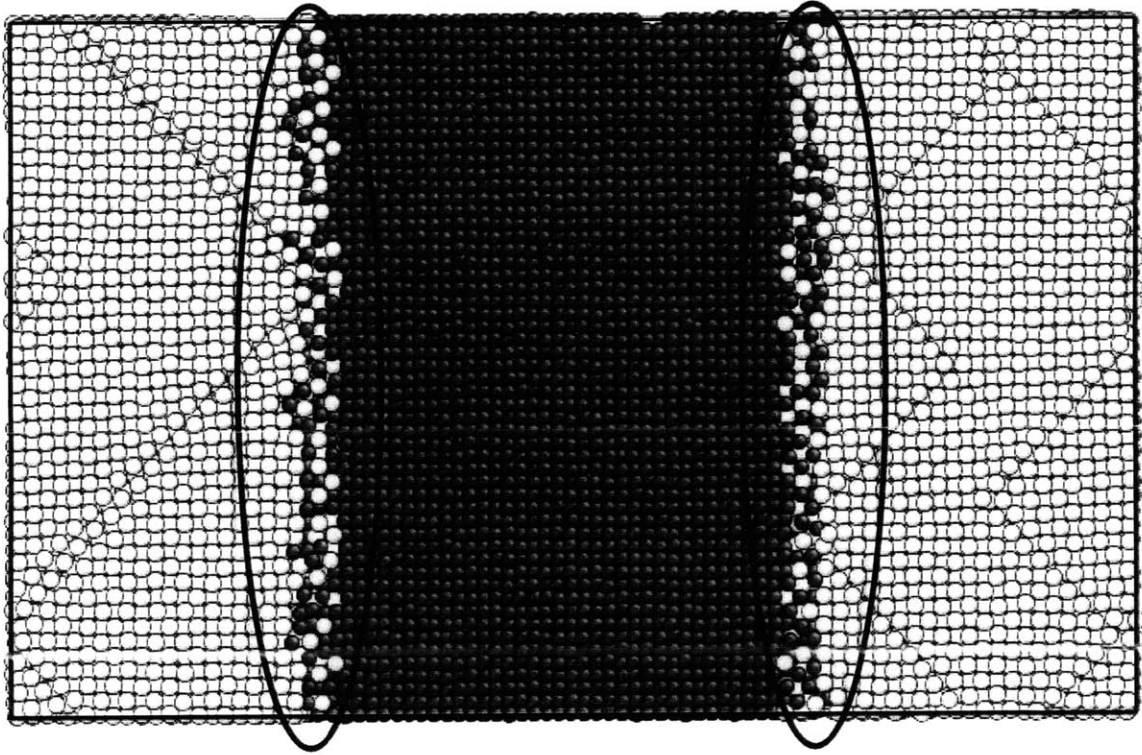


Figure 6-20 Atomic configuration at 4.5 nsec after impact loading

#### 6.4. Conclusion

In this chapter, we performed molecular dynamics simulations for the impact-induced alloying initiations, including a short-time impact loading process and a long-time impact induced process. For the combined and equilibrated Ni and Al nanoslabs (50234, 50378 atoms) with 1200 m/s impact velocity (0.243 eV per atom), we could observe clear impact induced alloying initiation at the interfaces between the two dissimilar materials, and consequent, propagation of slow alloying from the impact interfaces in the solid state around 750 K where the Ni and Al slabs are not liquid states. For 100 m/s to 1000 m/s impact velocities (under 0.2 eV per atom), we did not observe the interfacial mixing between the nanoslabs for the given size of the system. Based on the results, we conclude the parallel nanoslabs having zero curvature at the interfaces exhibit great resistance to shock loading and consequent intermixing processes.

However, if we decrease the size of the system, the alloying should be initiated even in the lower impact speeds. In the next chapter, we will investigate the impact-induced alloying initiation processes in

finite systems having the same number of atoms and impact speed. Also, for the same impact speed, we would discuss effect of size and shape of systems that are subject to impact loading.

## Chapter 7

# Non-equilibrium molecular dynamics simulations of Ni and Al nanosphere impacts

In this chapter, we focus on the impact-induced alloying initiation between the equilibrated Ni and Al nanoparticles. Unlike the nanoslab cases, it is not simple to create the impact loading for the nanoparticles of finite size. First we analyzed and selected the proper impact loading process for the nanoparticles by exploring several possible impact loading descriptions. Next, the system has been subject to the impact loading, and simulated with the NVE ensemble after the impact loading was initiated.

### 7.1. Shock loading within periodic boundary conditions (PBC): unexpected failure of the shock loading assumption

At first, we attempted to simulate impact loading using the same shrinking periodic boundary method as the nanoslab cases. To find the shrinking stop point, we investigate the time evolution of the total kinetic energy and mean drift velocities of the system. However, in these spherical systems, the compressive waves generated at the impact interfaces are not plane waves. That is, the compressive waves would not meet at the periodic boundaries over the lateral directions (orthogonal to the impact direction) at the same time. As a result, we instead examined the total kinetic energy evolution to find the shrinking boundary stop point. After finding the stop point, we performed the NVE simulation over 1 nsec.

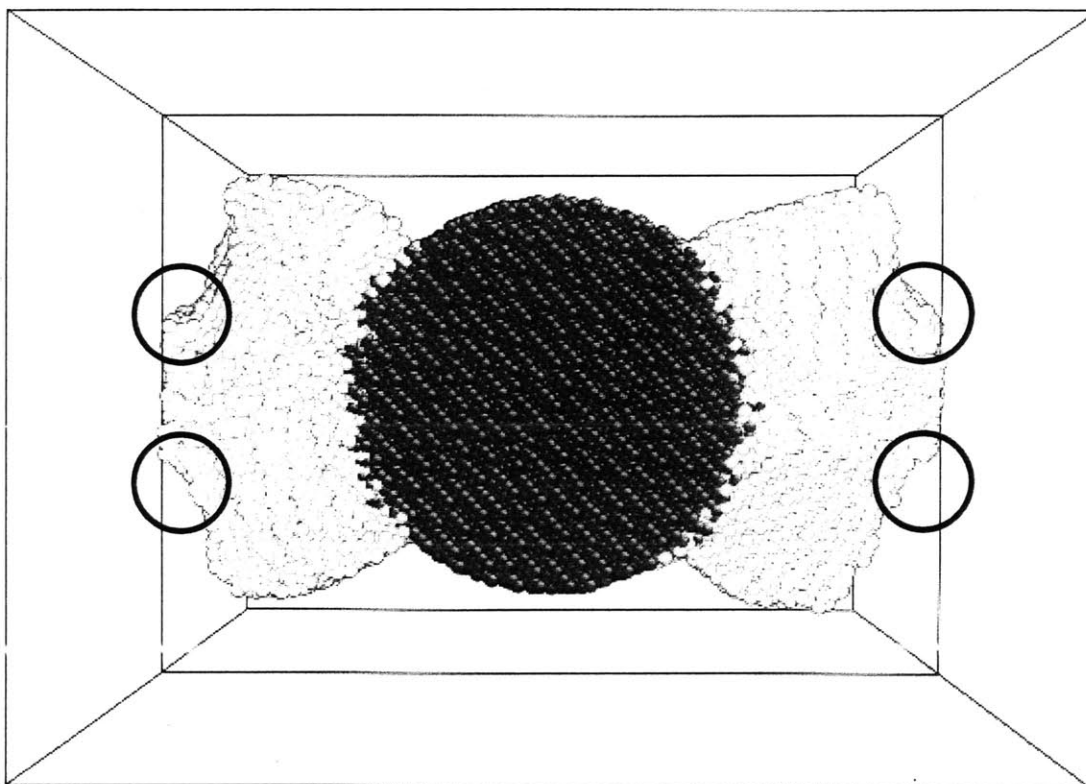


Figure 7-1 Failure of the nanoparticles at the periodic boundaries with the shrinking boundary method

However, with this approach, we observed unexpected failure at the periodic boundaries as seen in Fig. 7-1. That is, we concluded the shrinking boundary method for the finite system is not appropriate for impact of spherical particles.

## 7.2. Shock loading within non periodic boundary conditions

To find the proper impact loading process for the finite systems, we considered another method for the system. As described in Fig. 7-2, we took nanoparticles having half the diameter of the previous system; the left Al nanoparticle has been assigned with rigid velocity of 1200 nm/nsec over the thermal velocity of 300K. At the same time, the left boundary (non-periodic) is also assigned with the same rigid speed. After tracking the total kinetic energy of the system, we could find the stop point for the left boundary and fix the position of the boundary during the long-time simulations. That is, in this case, the left Al nanoparticle hits the right Ni particle with the free impact process, and the whole system is confined with the fixed size of



impact direction after the total kinetic energy minimum point (end of the free impact). It is necessary to restrict the whole system within the confined boundaries to prevent unexpected rigid motions of the nanoparticles after the impact loading. Also, all the boundaries are subject to the reflecting (momentum mirroring) conditions described in Chapter 6.

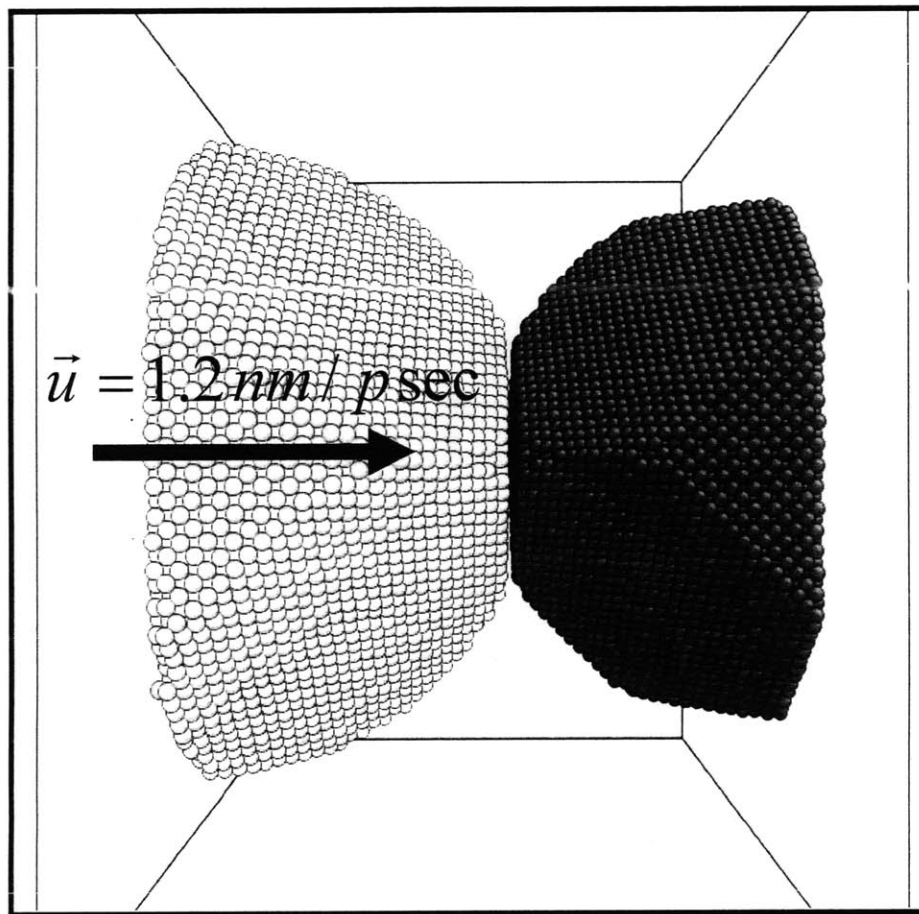


Figure 7-2 Atomic configuration of the facet nanoparticles before impact loading (Red: Ni, Yellow: Al)

We could find the impact loading end point around 5 fsec after the rigid motion, where the total kinetic energy is locally in minimum, and the total potential energy is locally in maximum. After this instant, the left boundary is fixed at the position, and the long timescale NVE simulation is performed to observe the phenomena after the impact loading that is confined within the boundaries.

### Simulation setup and sizes

Impact velocity: 1200 m/sec (0.234 eV/atom)

Initial temperature: 300K

Atoms in Ni slab: 25439

Atoms in Al Slab: 25348

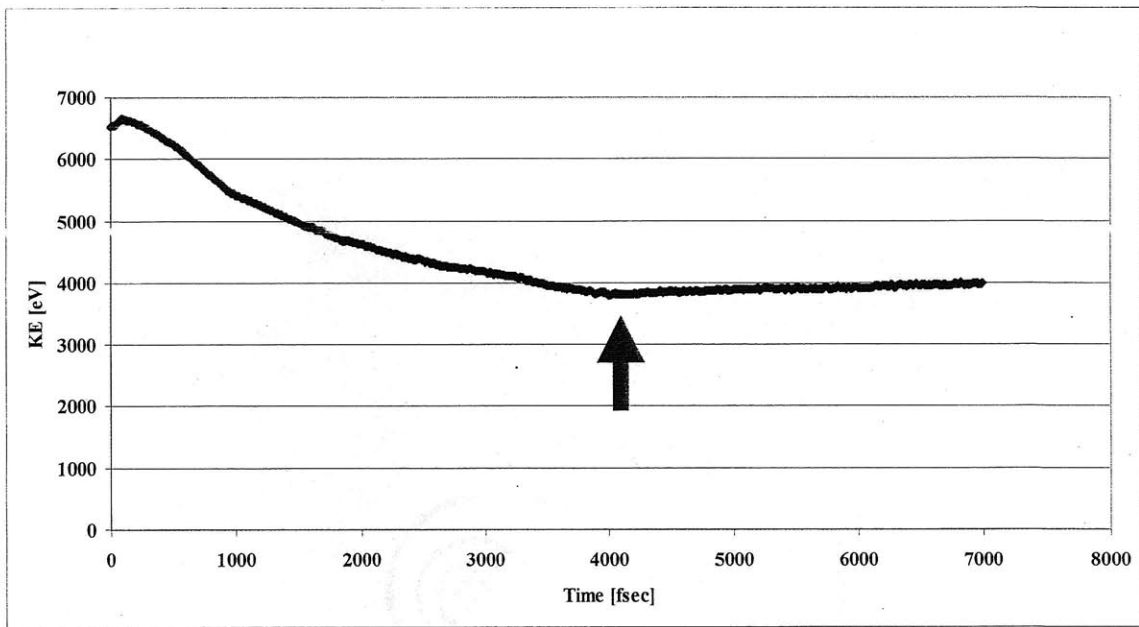


Figure 7-3 Total kinetic energy evolution during the impact loading (Arrow means the end of impact loading)

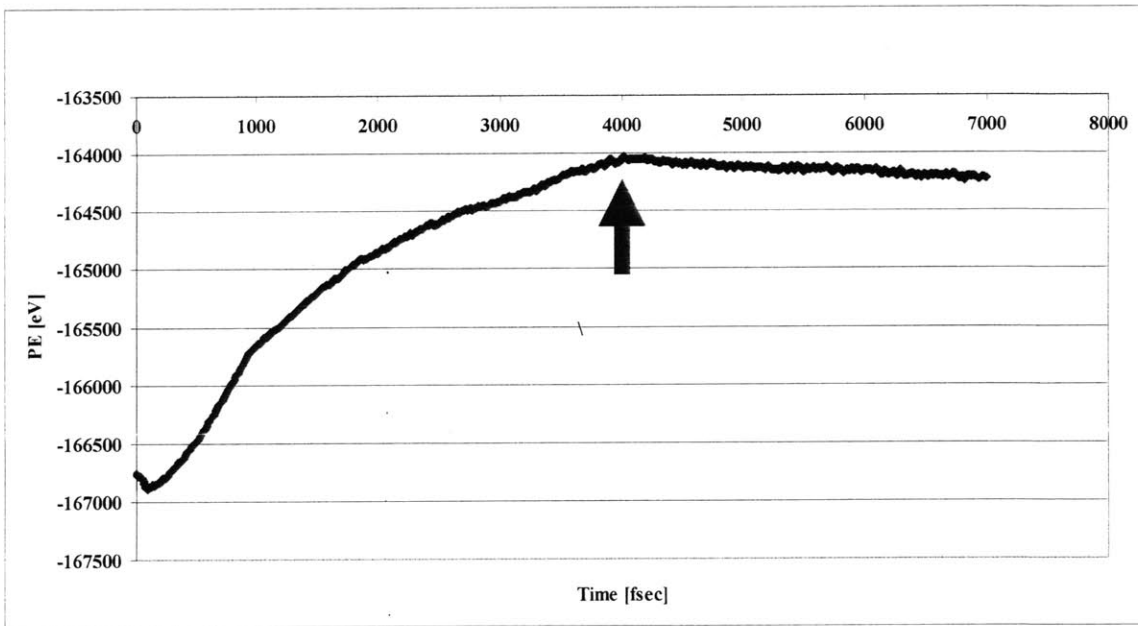


Figure 7-4 Total potential energy evolution during the impact loading (Arrow means the end of impact loading)

### 7.3. Evolutions of shock induced dynamic properties during the impact loading

To calculate the local dynamics properties in the system, the system has been divided to 30 pieces of local rectangular bins over the longitudinal direction. All the local properties have been averaged within each local bin. Also, the x directional (impact direction) size of the local bin has been also reduced during the impact loading.

(1) Shock temperature

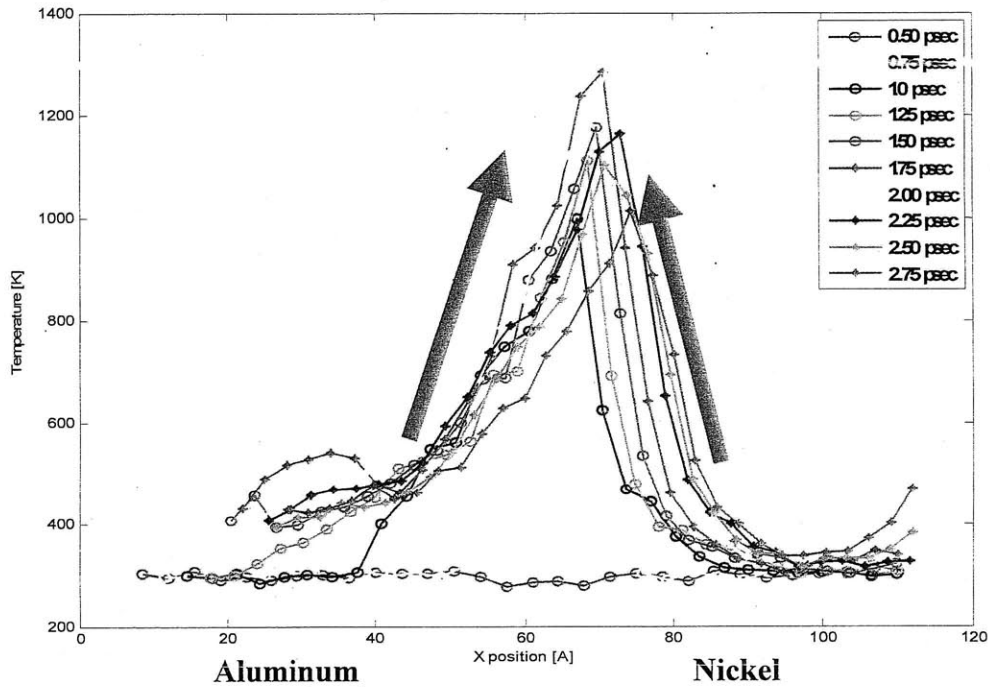


Figure 7-5 Instantaneous shock temperature evolution during impact from 0.5 psec to 2.75 psec

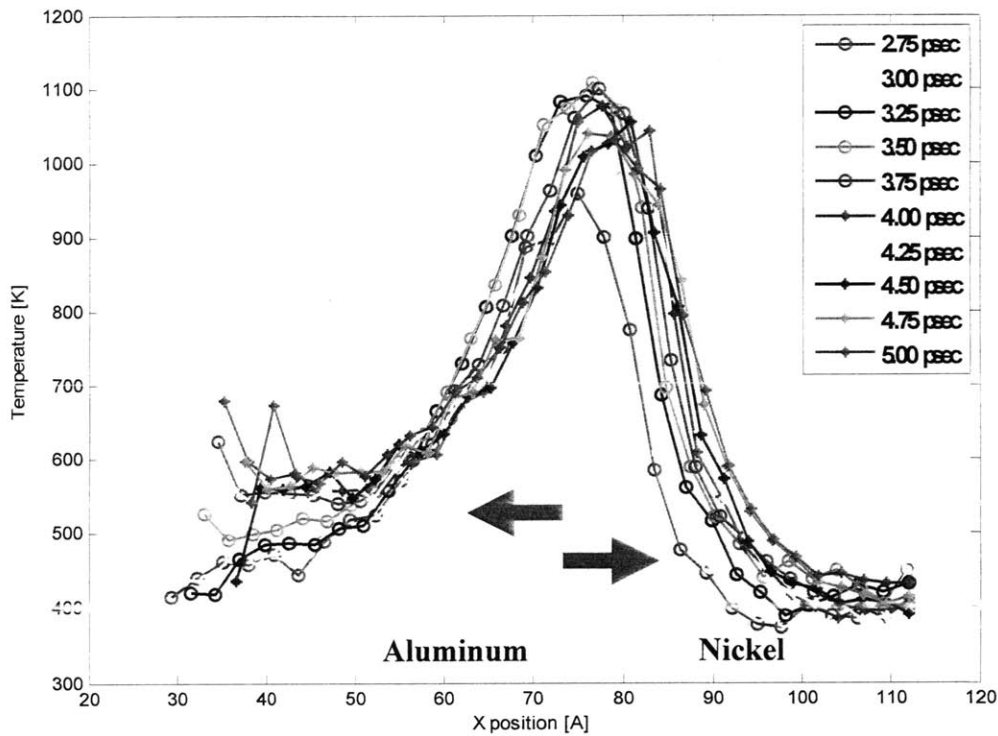


Figure 7-6 Instantaneous shock temperature evolution during impact from 2.75 psec to 5.0 psec

As expected, the shock temperature increases rapidly from the impact interface just after impact and the shock heating propagates through the nanoparticles. Also, the shock temperature significantly fluctuates at the free surfaces of the nanoparticles.

(2) Shock stresses and Von Mises shear invariants

Unlike the nanoslab cases, it is not straightforward to track the shock stress propagations during the impact loading. Also, in the nanoparticle cases, the instantaneous virial stresses fluctuate significantly during the impact loading because the nanoparticles of finite size respond to the impact loading more rapidly and the impact is extremely fast loading, that is nearly a point load applied to the impacted particles.

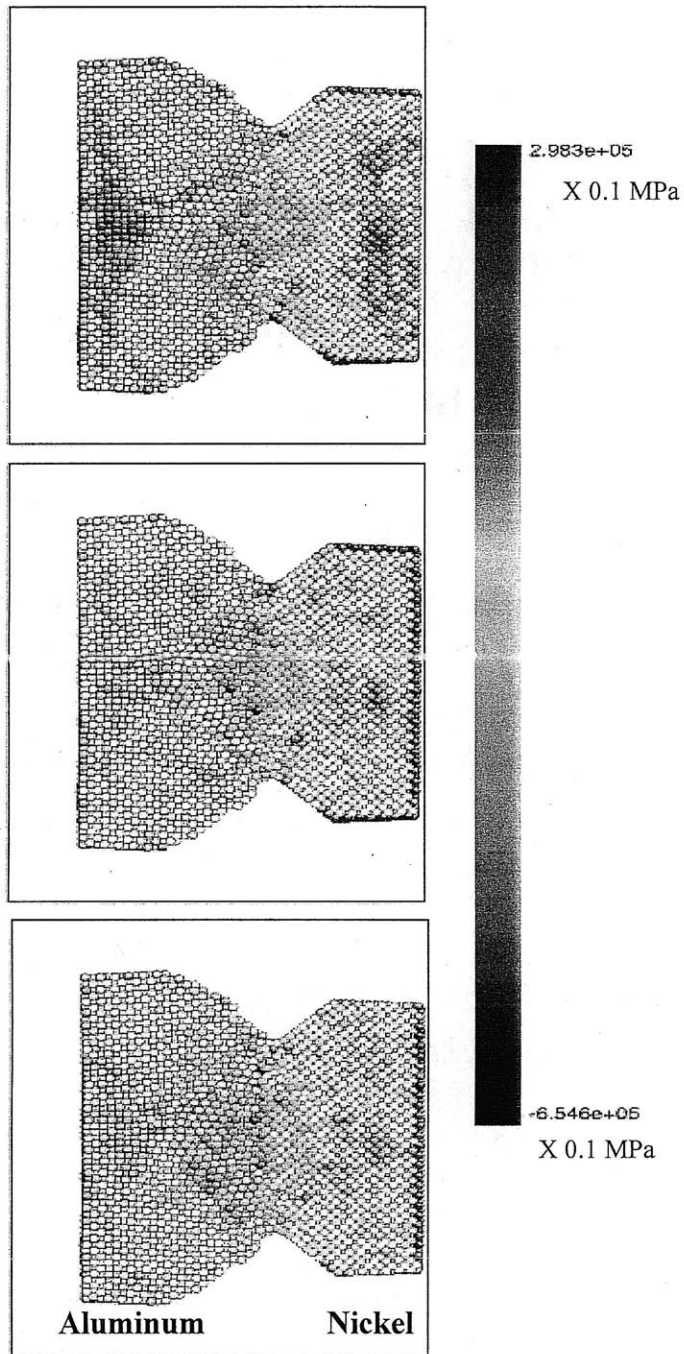


Figure 7-7 Instantaneous normal stresses (unit 0.1 MPa) at 2.75 psec after impact loading

As expected, the stress fluctuation over the nanoparticle is very wide. Also, at the right boundary, the stress fluctuation is specifically extreme because of the reflecting momentum conditions.

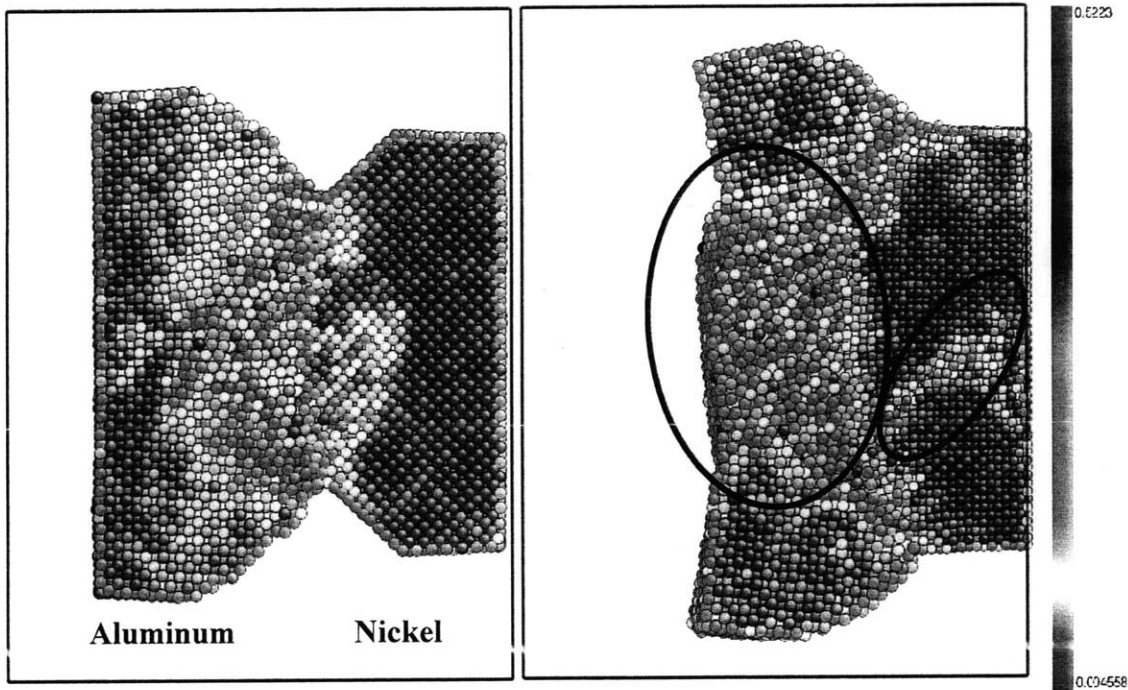


Figure 7-8 Von Mises shear strain invariants at the end of impact loading at 2.75 psec and 5.0 psec (end of impact loading, regions in circles yield)

After the end of impact loading (5.0 psec), most parts of the core region of the Al nanoparticle exhibits high Von Mises shear strain. However, in the Ni nanoparticle, a localized shear band is observed around the core region and other parts in the Ni nanoparticle still exhibit small shear strain.

(3) Potential energy

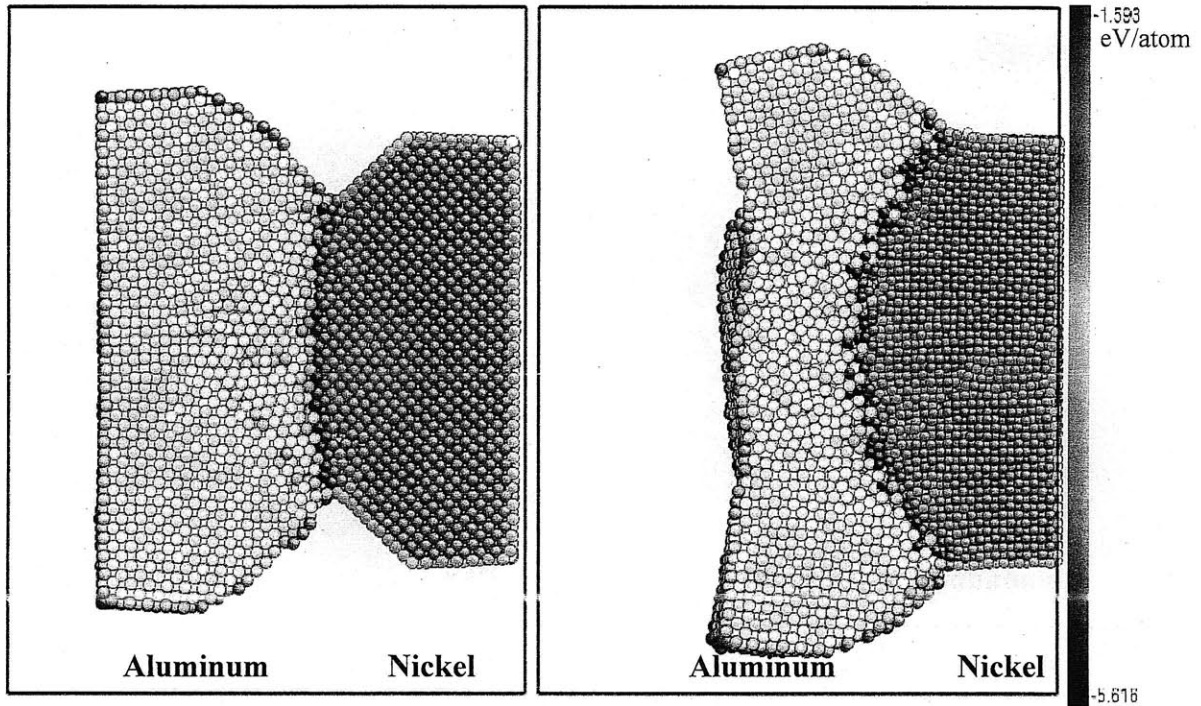


Figure 7-9 Potential energy (unit: eV/atom) evolution during the impact loading (a) 2.5 psec (2) 5 psec

As seen in the figures, around the end of the impact loading, 5 psec, intermixing between the Ni and Al atoms has been initiated. This is much faster initiation than the nanoslab cases with the same impact speed even though the instantaneous shock temperature peaks (around 1400K) at the interface are very similar in those two cases. Though the nanoparticles have the clear  $\{1\ 0\ 0\}$  facets at the impact direction (the same as the nanoslab cases), the alloying initiations could be accelerated because there exist many atoms having lower coordination numbers at the four edges of the facets. That is, the atoms having lower coordination numbers are acting as initial catalysts for the alloying activation.



#### 7.4. Dynamic properties during the long time scale simulation

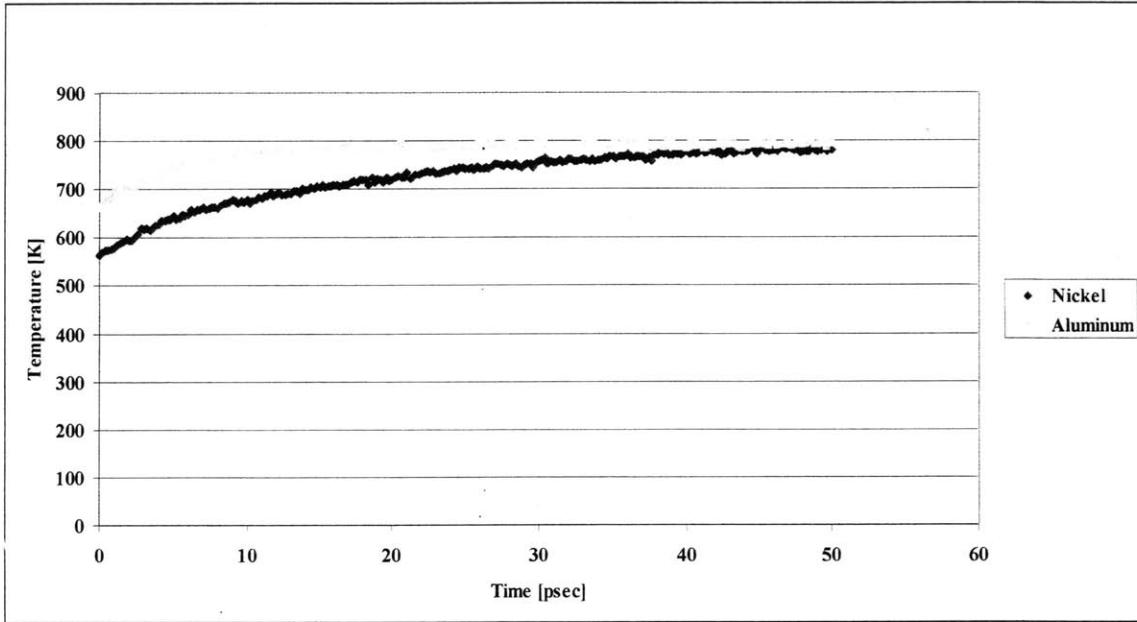


Figure 7-10 Temperature evolution of Ni and Al nanoparticles in the initial stage of NVE simulations

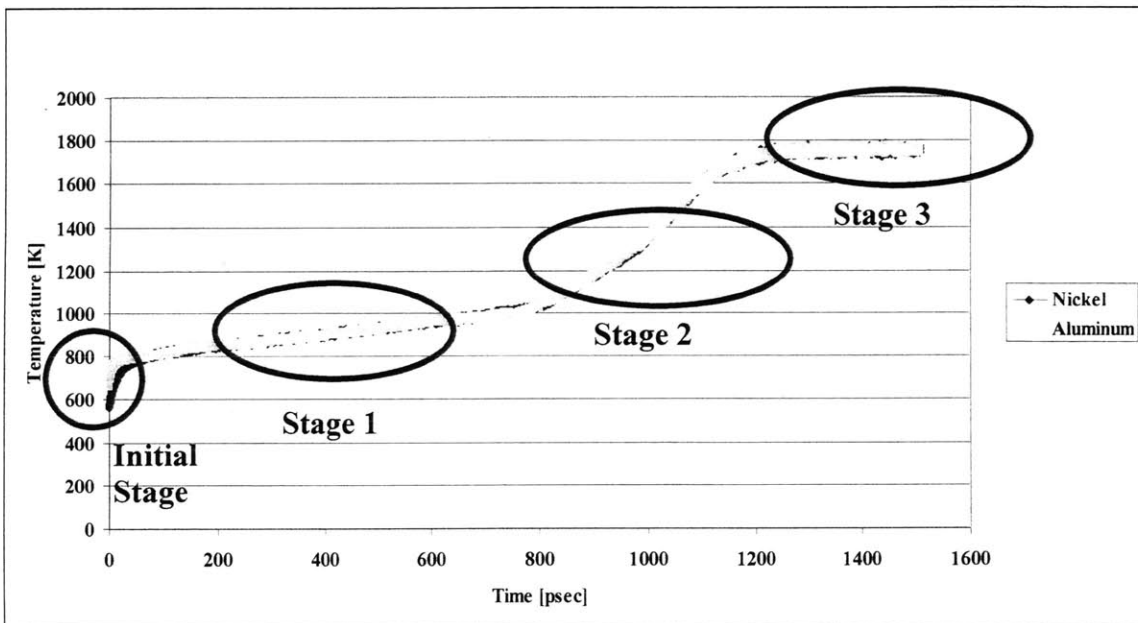


Figure 7-11 Temperature evolution of Ni and Al nanoparticles during the long timescale NVE simulations

(1) Initial stage

In the initial stage of the shock relaxation process after the completion of the impact loading, temperature in the Ni and Al nanoparticles converges around 800K. However, in the nanoslab case, the initial temperature convergence between the Ni and Al systems during the shock relaxation processes was between the temperature of the Ni and Al system with the extreme fluctuations. In the nanoparticles, the initial convergence temperature between the Ni and Al system is even over the temperature of the entire Al nanoparticle because the exothermic alloying reactions had been already initiated during the impact processes; consequently, there is energy of formation heating during the initial stage of the NVE simulation.

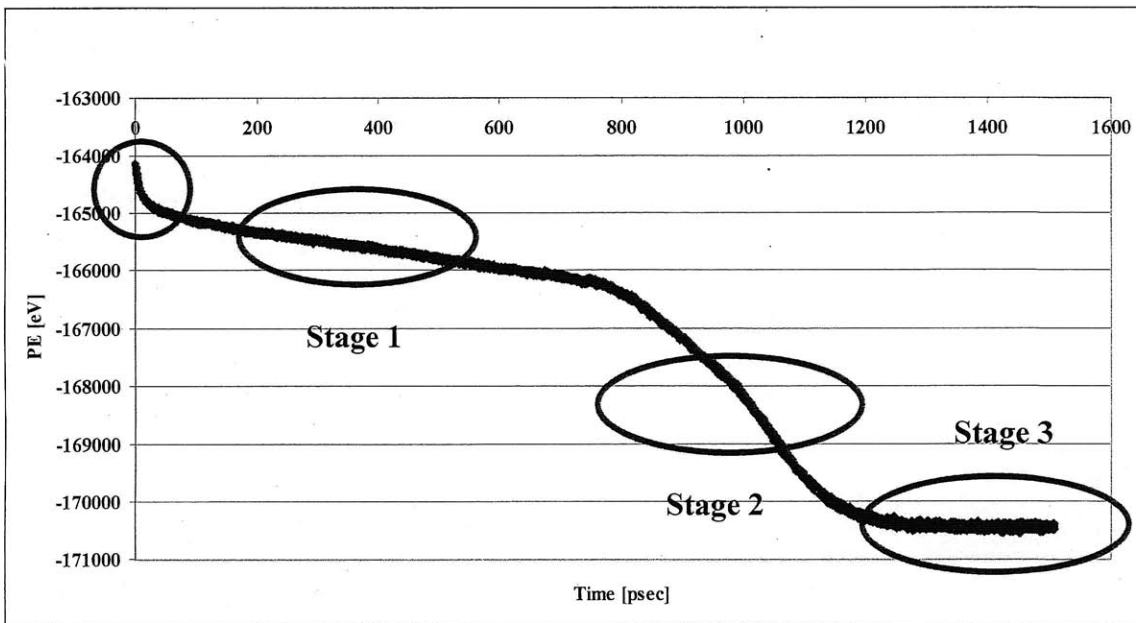


Figure 7-12 Total potential energy evolution during the long time scale NVE simulations

Also, during the initial stage, the total potential energy of the system decreases with the increased intermixing between Ni and Al atoms.

(2) Stage 2: intermixing + coalescence process

During stage 2, there is a slow intermixing process accompanied by the coalescence process, where the Al atoms at the surface diffuse to the Ni surface. This process is very similar to the coalescence process observed in the sintering process for the Ni and Al nanoparticles.

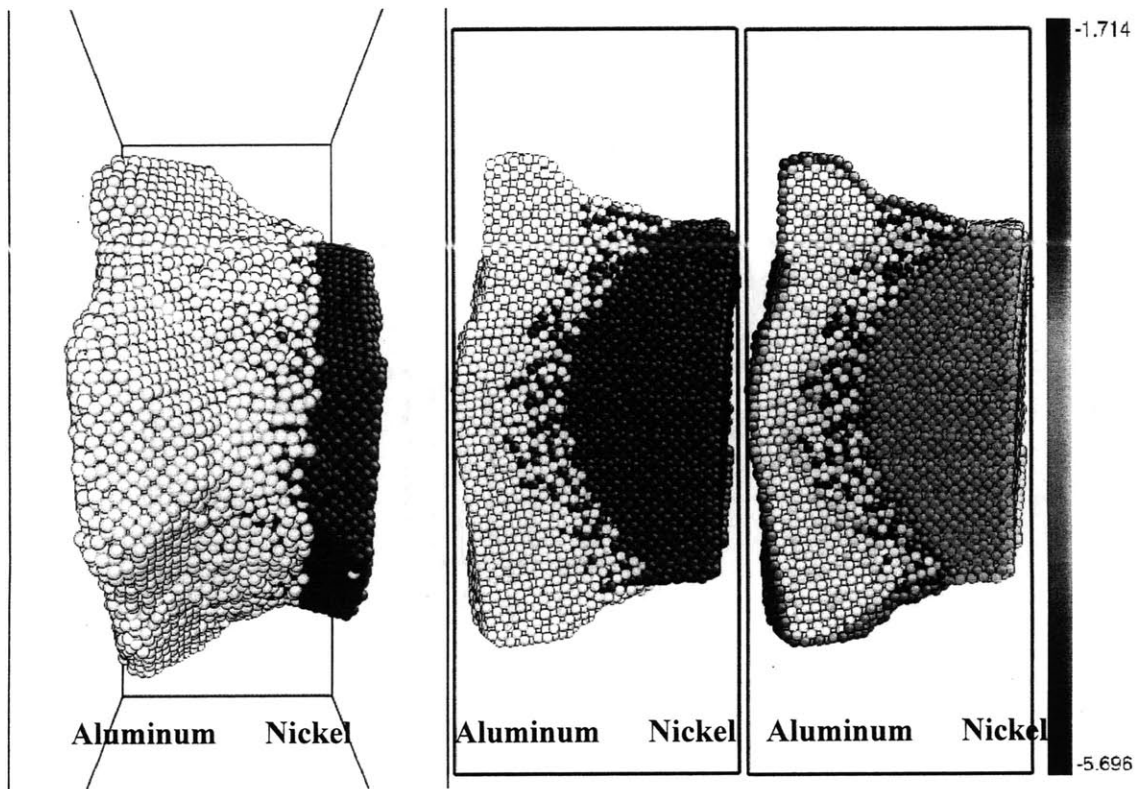


Figure 7-13 Atomic potential energy (unit: eV/atom) configuration at the stage 2, 500 psec

At the end of stage 2, around 950 K at 750 psec, the Al nanoparticle starts melting from its surface (experimental and the current EAM based melting points: 930K and 980K, respectively).

(3) Stage 3: melting of Al nanoparticle+fast coalescence process+fast intermixing+melting of Ni nanoparticle

During stage 3, after melting of the Al nanoparticle, the surface coalescence process and intermixing process have been accelerated because, currently, the Al atoms are in the liquid metal state. As seen in the figure, the temperature increases much faster than in stage 1. Also, the Ni nanoparticle starts melting from its surface around 1700K at 1100 psec (experimental and the current EAM based melting points: 1750K and 1830K, respectively). In the figure, the Ni atoms diffuse to the Al nanoparticle fast, and the intermixing between Ni and Al atoms is also very fast at the Ni and Al interfaces. However, the core of the Ni nanoparticle is still in the solid state.

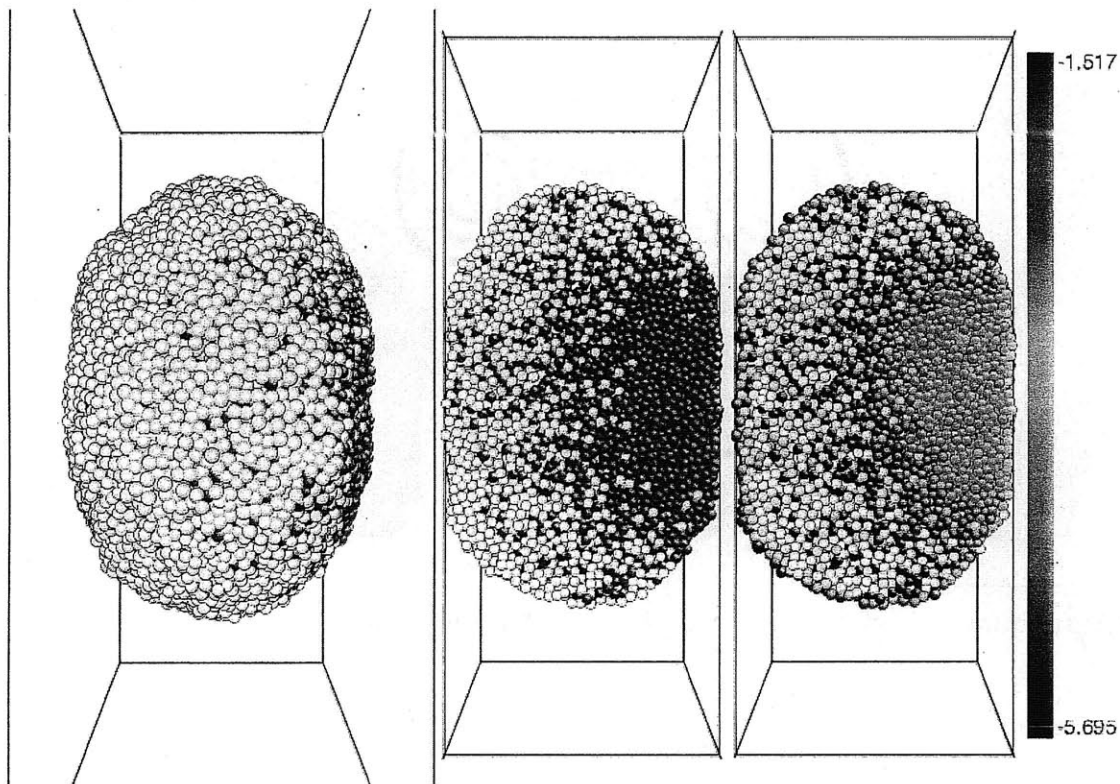


Figure 7-14 Atomic potential energy (unit: eV/atom) configuration at the stage 3, 1000 psec (1 nsec)

(4) Stage 4: completion of melting of Ni nanoparticle + coalescence + intermixing: fully equilibrated process for the given volume constraint

During stage 4, the Ni nanoparticle completely melts. All the processes including (1) intermixing between Ni and Al atoms (2) coalescence and (3) melting of Ni and Al nanoparticles are completed at this stage. Finally, the temperature and total potential energy of the system converge toward equilibrium values for the given volume (fixed X size) and pressure (0 external pressure) constraints. We clearly observe the plateaus of temperature and total potential energy at this stage in Fig. 7-11, 12.

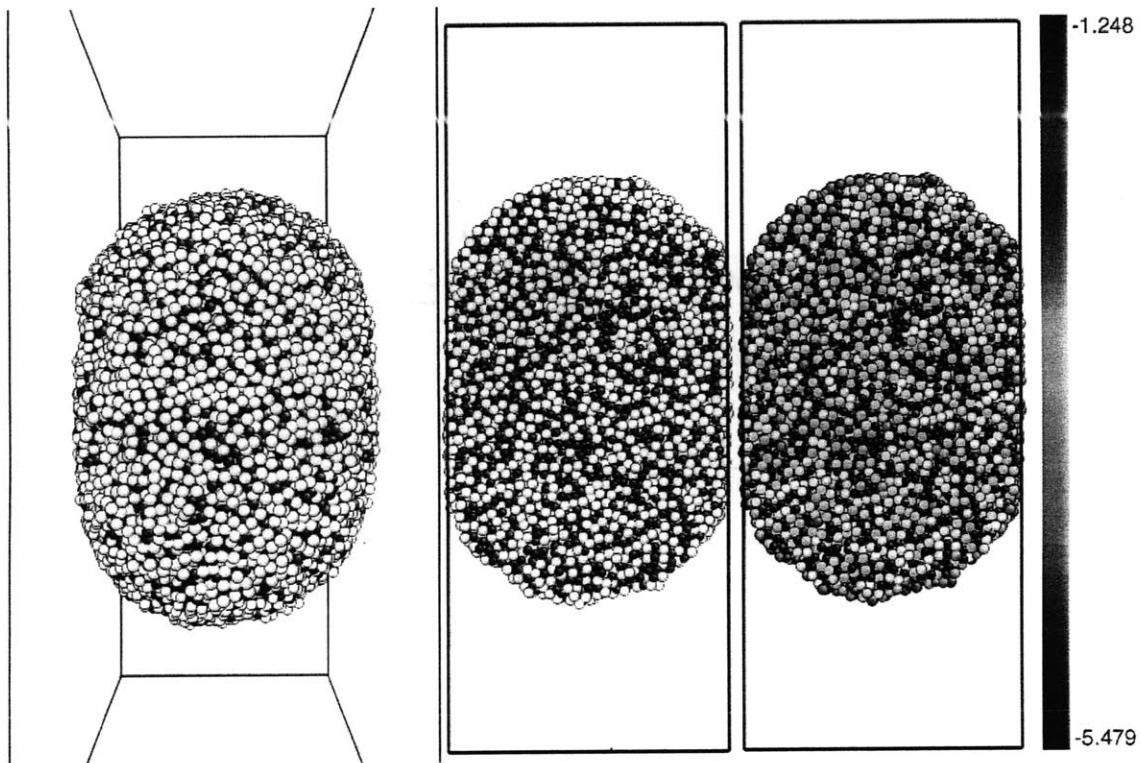


Figure 7-15 Atomic potential energy (unit: eV/atom) configuration at the stage 4, 1500 psec (1.5 nsec)

### 7.5. Impact between nanoparticle having rough surfaces

In this section, impact between nanoparticles which have rough surfaces (atomic islands on each facet) and the same numbers of atoms and impact velocity in the system as those in the previous sections are analyzed and compared to the previous faceted nanoparticle case.

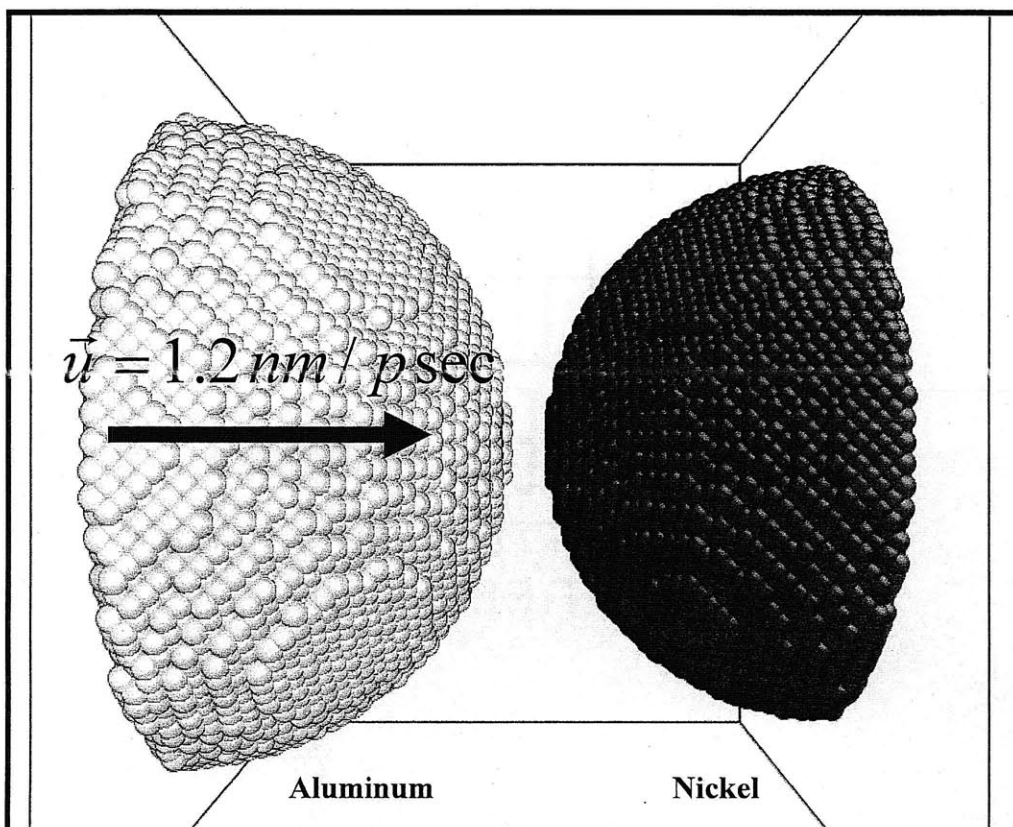


Figure 7-16 Atomic configuration of the non-facet nanoparticles before impact loading

Usually, with the synthesis processes of the nanoparticles, the surface roughness with the atomic islands on the facets can be controlled experimentally. The atoms on the islands have lower coordination numbers than the atoms on the corresponding faceted surfaces, and there are more such under-coordinated atoms in a spherical nanoparticle in Fig. 7-16 than a faceted nanoparticle of same radius. That is, by this characteristics, we hypothesized the atomic islands can catalyze the intermixing reactions between the Ni and Al nanoparticles under otherwise identical experimental conditions such as size of the nanoparticles and input impact velocities (energy).

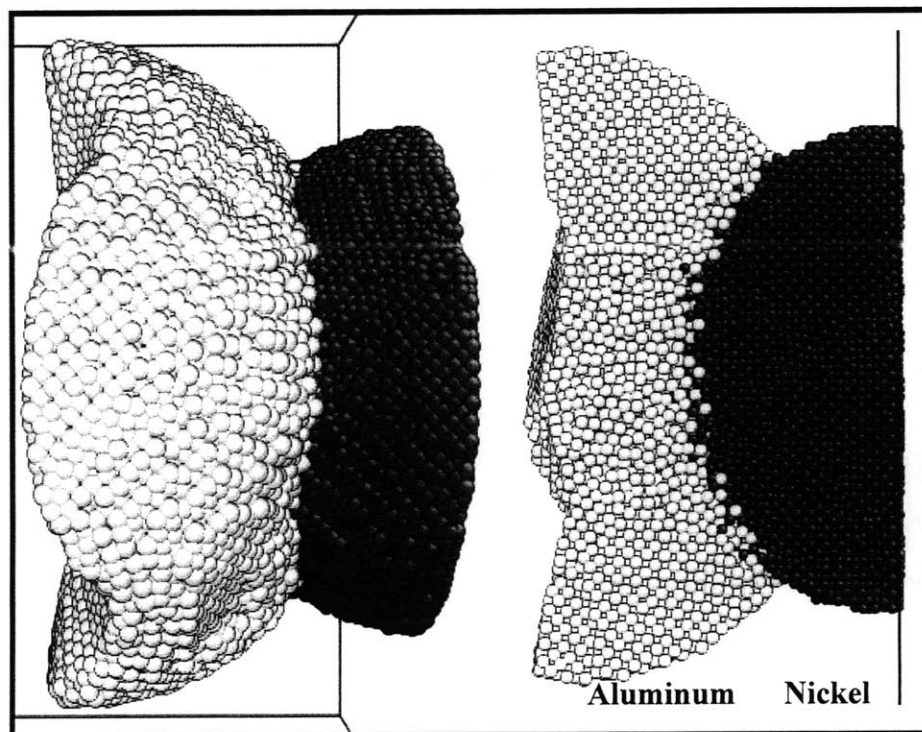
**Simulation setup and sizes**

Impact velocity: 1200 m/sec (0.234 eV/atom)

Initial temperature: 300K

Atoms in Ni slab: 25648

Atoms in Al Slab: 25583



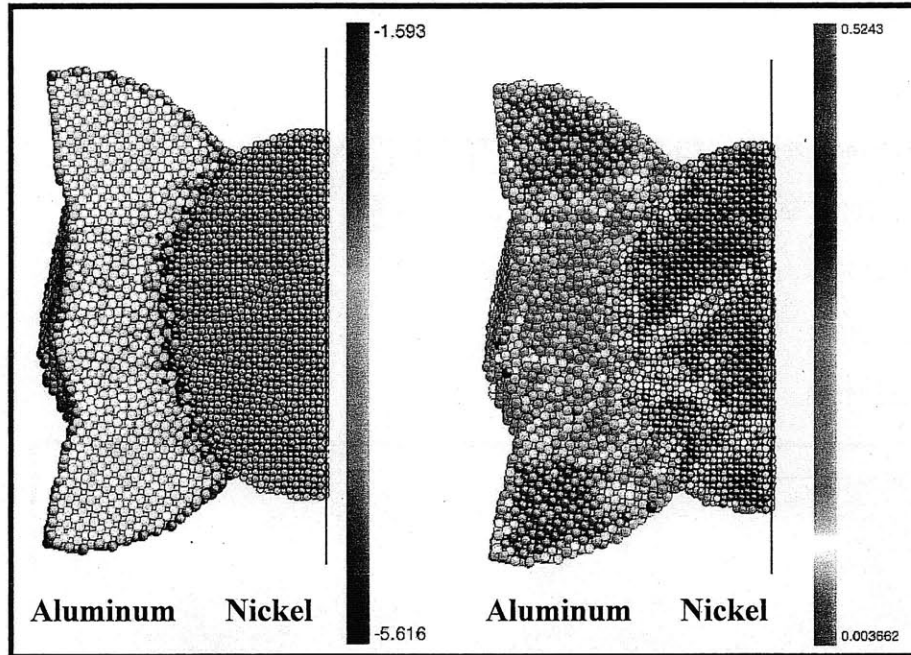


Figure 7-17 Atomic configuration at 1.5 psec after the impact loading (1) Atomic position (2) Atomic potential energy (unit: eV/atom) (3) Von Mises shear strain invariants

In the early stage of the impact loading, the alloying initiation has been observed around 1.5 psec. This is faster than the faceted cases. That is, during the impact loading, the lower coordination numbers and the corresponding lower binding energies at the impact surfaces accelerated the initial intermixing between Ni and Al atoms. Also, with the relative frictions between Ni and Al interfaces, the initial shock temperature has been observed at 50 – 60 K higher than the facet cases. Finally, we would address the total potential energy and temperature evolution during the whole duration of the simulation.



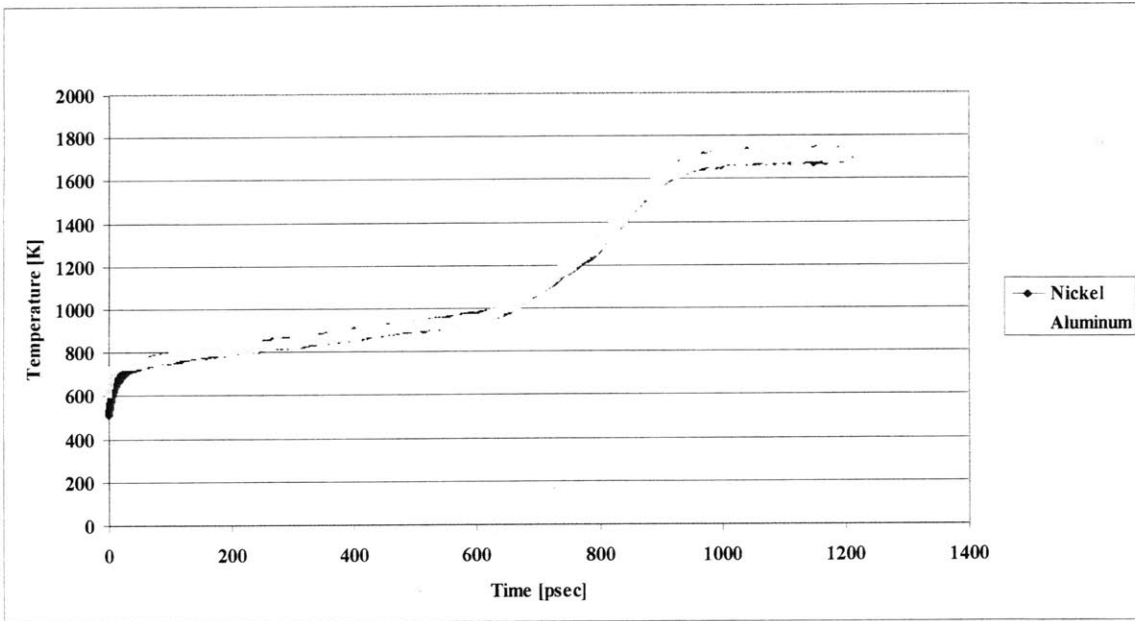


Figure 7-18 Temperature evolution of Ni and Al nanoparticles during the long time scale NVE simulations

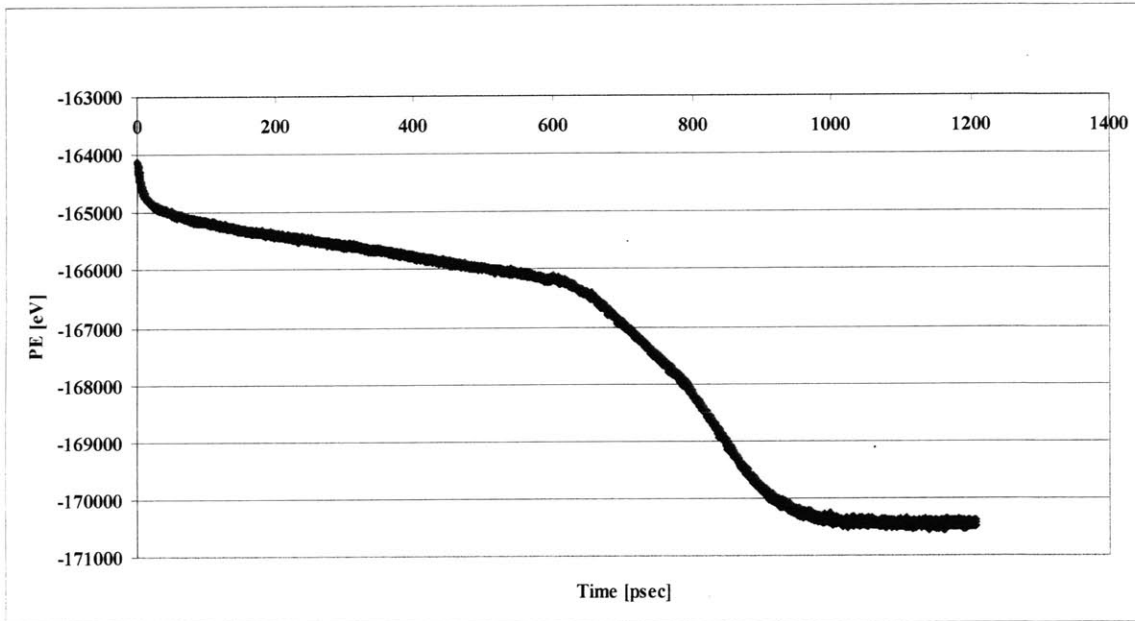


Figure 7-19 Total potential energy evolution of Ni and Al nanoparticles during the long time scale NVE simulations

As seen in Fig. 7-18, 19, all the time durations in the stages are shorter than the faceted cases. Also, the Al nanoparticle started melting around 910 K at 600 psec. Finally, the completion of the whole processes has been accelerated in comparison with the faceted cases.

#### **7.6. Effect of nanoparticle sizes**

For finite systems having large ratio of surface area to volume, the size of the system and surfaces can affect the physical and chemical properties of the systems [61]. As we increase the size of the finite system, the surface-volume ratio,  $R$  decreases. Also, for small  $R$  the properties of the system tend toward those of the bulk material. However, if we decrease the size of the finite system, the properties of the system are much different from the bulk systems. Especially, for the current issues on the melting, coalescence and intermixing processes, this size effect would be important for the whole systems. For example, for extremely small nanoparticles called nanocluster under 500 – 1000 atoms, the melting temperature of the nanocluster is much less than in the bulk [61].

In this section, we would perform the same simulations for the nanoparticles of larger diameter than the previous cases. Here, we would construct the nanoparticles having 25 % more atoms than the previous cases. However, the impact velocity (input energy per atom) and the ratio of Ni and Al atoms will be maintained constant.

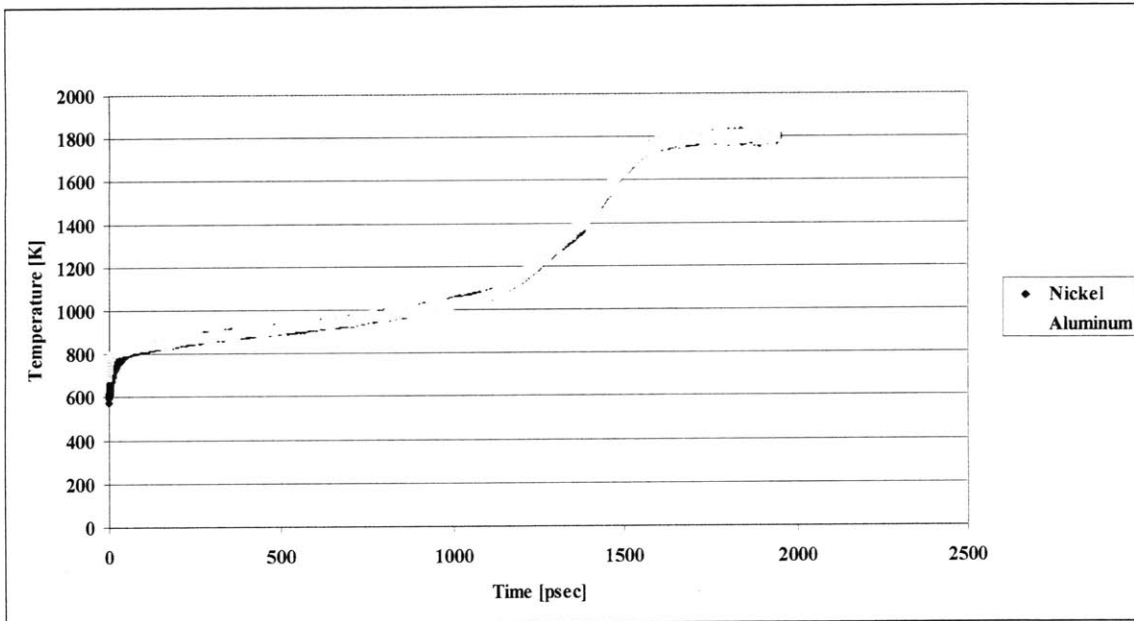


Figure 7-20 Temperature evolution of Ni and Al nanoparticles during the long timescale NVE simulations

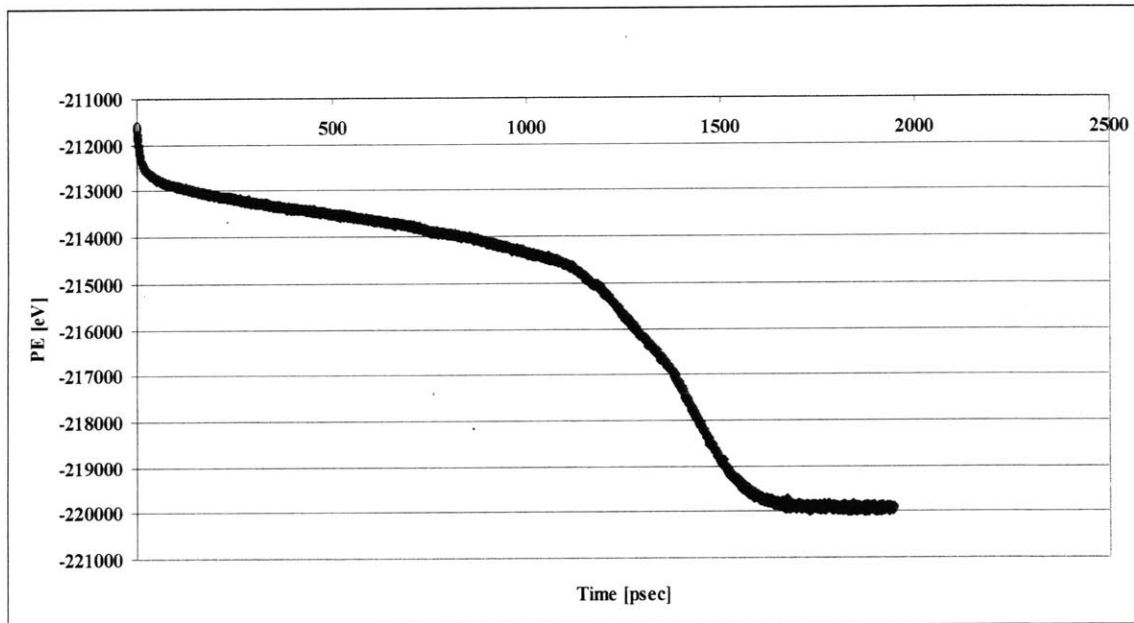


Figure 7-21 Total potential energy evolution of Ni and Al nanoparticles during the longtime scale NVE simulations

As seen in the temperature evolution for this case, stage 2 (melting of the Al particle) starts around 10 nsec (over 1000K). In the previous case (faceted case), stage 2 started around 8 nsec (over 930K). This means the total evolution rate during the whole processes of melting, coalescence, and intermixing has been decreased by increasing the size of the system. Also, in the nanoscale finite systems, the melting temperature of the system is very sensitive to the size. That is, in our simulations, the melting temperature of the current size is greater by 50 – 70 K than the previous size. However, final temperature at the equilibrium state at the end of simulations is very similar to the previous case (around 1800K).

### 7.7. Conclusion

In this chapter, we performed molecular dynamics simulations for the Ni and Al nanoparticle impacts. For the same size of the Ni and Al systems as the previous nanoslab cases, we could clearly observe the early stage of the Ni and Al alloying initiation, and consequent melting of Al and Ni nanoparticles, coalescence processes, and full mixing between the nanoparticles. That is, even though the starting shock temperatures after the impact loading in the nanoslab cases and nanoparticle cases for the same size and impact energy are very similar, we could observe complete and self-sustained alloying propagations and full mixed state of the Ni and Al systems in the nanoparticles that was absent in the nanoslab cases. Due to the high number of undercoordinated atoms in the surfaces, the alloying initiation has been accelerated in the initial stage of the impact loading. Consequently, the alloying propagation speed was also accelerated within the nanoparticles. That is, even though the impact direction  $\{1\ 0\ 0\}$  is the same at the nanoslab and nanoparticle cases, those atoms surrounding  $\{1\ 0\ 0\}$  surfaces (facet) had lower coordination number and atomic potential energy (lower bond strength in Ni-Ni and Al-Al atoms), and catalyzed the initial alloying initiations. Also, for the finite system with the shock heated environments, the melting of the system was achieved more easily. Also, for the nanoparticle cases, we examined the effect of particle diameter on the whole processes by increasing the number of atoms in the system by 25 %. As expected, in the larger particle, the total process time has been increased. Finally, the final temperature at the equilibrium state for the nanoparticle cases is around 1800K [45]. This is very close to the effective adiabatic temperature for the synthesis of the Ni and Al alloy systems, which had been used in chapter 2 on continuum simulations of the self-sustained alloying propagations. In the continuum simulations, we used the initial peak temperature at

the boundary in the effective adiabatic temperature for the alloying processes, which we found to be 1830K to initiate the alloying reactions.

## Chapter 8

### Implementation of Reactive Force Field (ReaxFF) potential for future research

LAMMPS which has been used through this research stands for Large-Scale Atomistic and Molecular Massively Parallel Simulator, which is an object oriented classical molecular dynamics simulation code designed to run effectively on parallel computers. For the last several years, various force field potentials such as the Embedded Atomic Method (EAM), Morse, Tersoff and the Lennard-Jones potentials have been developed for the classical molecular dynamics simulations of various material systems as internal classes in LAMMPS. However, the current force field potentials in LAMMPS are not capable of describing chemical reactions with complex molecular structures. To make practical the molecular dynamics simulations of large scale complex systems with a variety of chemical reactions in LAMMPS, the implementation of the reactive force field is essential. The general reactive force field (ReaxFF) was originally developed at California Institute of Technology (CalTech) in 2001. Goddard and Van Duin et al., initially developed and implemented ReaxFF with Fortran for hydrocarbon systems with chemical reactions, and have extended ReaxFF to other complex material systems such as C, H, N, O systems [54]. To make it possible to use ReaxFF for the reactive molecular dynamics simulations in LAMMPS, we developed one force field class which uses Caltech's ReaxFF Fortran code as a library linked to LAMMPS. This work was completed in part during Hansohl Cho's internship at Sandia National Laboratories (June 15 – August 30, 2008) and was completed over the following fall (Sept 1 – December 31, 2008). Research at Sandia was led and supervised by Dr. S. J. Plimpton and Dr. A. P. Thompson; Dr. Plimpton is the original author of LAMMPS [62]. We tested the force field class by calculating chemically high energy material systems such as RDX and TATB. Section 8.1 describes the theoretical overview of the reactive force field. Section 8.2 describes the implementation of the ReaxFF in LAMMPS. Section 8.4 presents the application of the ReaxFF to the high energy material systems, RDX and TATB. Based on the implemented parallelized

ReaxFF in LAMMPS, we would perform the future research on the oxidation of the Ni and Al nanoparticles, and the polymer composites, and their interfacial and mixed state in the polymer nanocomposites.

### **8.1. Theoretical backgrounds in ReaxFF: fully reactive Carbon Hydrate systems**

The accuracy and speed of modern quantum chemistry (QC) methods allow the geometries, energies and vibration energies to be predicted quite accurately for small molecules. However, QC is not yet practical for studying the dynamic properties of larger molecules and solid state systems with chemical reactions. Consequently, it is useful to have accurate force fields (FF) to quickly evaluate the forces and other dynamics properties such as the effects of mechanical shock waves or diffusion of small molecules in polymer. Generic FF such as DREIDING and the universal FF allow predictions for broad classes of compounds. However, in general, these force fields do not describe chemical reactivity. An exception is the Brenner potential, which leads to accurate geometries for ground states of hydrocarbons, but is formulated in such a way that can describe bond breaking. However, the Brenner potential does not include the van der Waals and Coulomb interactions that are very important in predicting the structures and properties of many systems. In addition, the actual potential curves for bond breaking and reactions are poorly described with the Brenner potential. Also, though other bond order dependent force field methods such as the bond energy bond order (BEBO) method and the valence bond (VALBOND) method were proposed to describe the chemical reactions, these methods do not fully address the need to have full chemistry of breaking and forming bonds, in addition to a proper description the fully bonded equilibrium geometry of complex molecules.

The more advanced and generalized reactive force field (ReaxFF) which has general bond order dependent potential terms with the van der Waals and Coulomb terms as non-bonded interactions to predict and describe the complex chemical systems with reactions was developed and proposed by Adri C. T. van Duin and William A. Goddard III in 2001 based on the reaction curves derived from QC calculations. Similar to empirical nonreactive force fields, the reactive force field divides the system energy up into various partial energy contributions as demonstrated by the equation (8-1).

$$E_{System} = E_{bond} + E_{over/under} + E_{lp} + E_{pen} + E_{coa} + E_{hb} + E_{tors} + E_{conj} + E_{val} + E_{vdW} + E_{Coul} \quad (8-1)$$

A fundamental assumption of the ReaxFF is that the bond order between a pair of atoms can be obtained directly from the interatomic distance and consists of three exponential terms: (1) the sigma bond, (2) the first pi bond, and (3) the second pi bond. Also, all bonded terms in the equation (1) are made dependent on the calculated bond order. Each energy term is described as below.

$E_{bond}$  : Bond energy. Bond energy is described as a function of the total bond order consisting of the sigma bond, the first pi bond, and the second pi bond in Hydrocarbon systems only including C-C, C-H, and H-H [1]. However, in systems including N and O, the bond energy is calculated from the separate energy contributions of the sigma, the first pi, and the second pi bonds [2].

$E_{over/under}$  : Atomic over-/under coordination energy. From the valence theory of bonding, we know that the total bond order of C should not exceed 4 and that of H should not exceed 1, except in hypervalent cases. However, even after correction of the original bond orders, a degree of over-coordination may remain in the molecule. To handle this, an over-/under coordination penalty term is added.

$E_{lp}$  : Lone pair electron energy. Lone pair electrons generally play little role in hydrocarbon chemistry, but lone pairs on such heteroatoms as oxygen and nitrogen are important in the response of these atoms to over-/under coordination.

$E_{val}$  : Valence angle energy. Just as for bond energy terms, it is important that the energy contribution from valence angle terms goes to zero as the bond orders in the valence angles goes to zero. Also, we need to ensure that dependence of the energy of valence angle accounts properly for the bond order greater than 1.

$E_{pen}$  : Penalty energy. To reproduce the stability of systems with two double bonds sharing an atom in a valence angle, an additional energy penalty is imposed for such systems.

$E_{hb}$  : Hydrogen bond energy

$E_{conj}$  : Conjugate energy. The contribution of conjugation effects to the molecular energy

$E_{tors}$  : Torsion angle energy. Just as with valence angle terms we need to ensure that dependence of the energy of torsion angle accounts properly as the bond order goes to zero and for the bond order greater than 1.

$E_{coa}$  : Valence angle conjugation energy. In simple valence bond theory, this is described in terms of a formal charge transfer. Also, the conjugation term is keyed into torsion angle. This accounts for the effect of conjugation on rotational barriers.

$E_{vdW}$  : Nonbonded van der Waals interactions.

$E_{Coul}$  : Nonbonded Coulomb interactions. In addition to the valence interactions, there are repulsive interactions at short interatomic distances due to Pauli principle and attraction energies at long distances due to dispersion. These interactions, comprised of van der Waals and Coulomb forces, are induced for all atom pairs, thus avoiding awkward alterations in the energy description during bond dissociation. In the ReaxFF, for the van der Waals interactions, a distance corrected Morse potential was used and for the Coulomb interactions, the charge equilibration (QEq) scheme proposed by Raffe and Goddard, [55] was used.



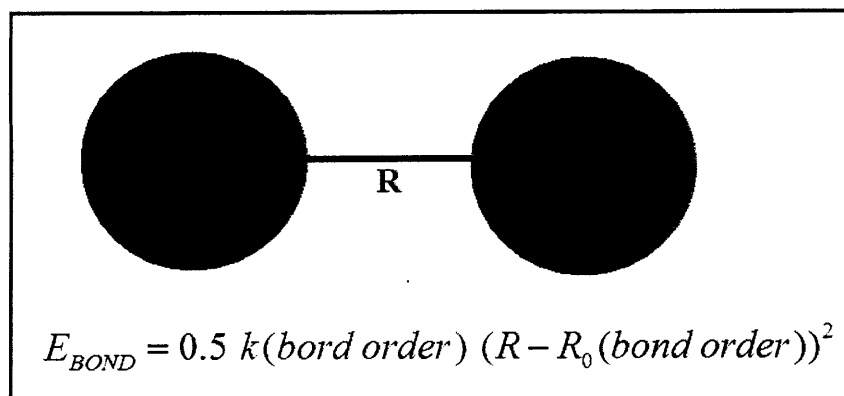


Figure 8-1 Schematic of the bond order potentials

LAMMPS has been developed and updated based on the object oriented programming with C++. Thus, we designed and implemented the ReaxFF as one pair potential class in LAMMPS. This class provides an interface between LAMMPS and the ReaxFF energy functions. The latter are based on the original ReaxFF code.

As described in Fig. 8-2, we directly used the ReaxFF to calculate the reactive energy terms in LAMMPS by compiling the ReaxFF Fortran subroutines as a library and linking it to the PairReax class in LAMMPS. The PairReax class in LAMMPS has been implemented to allow serial execution and also parallel execution using the MPI message passing library and a spatial-decomposition of the simulation domain.

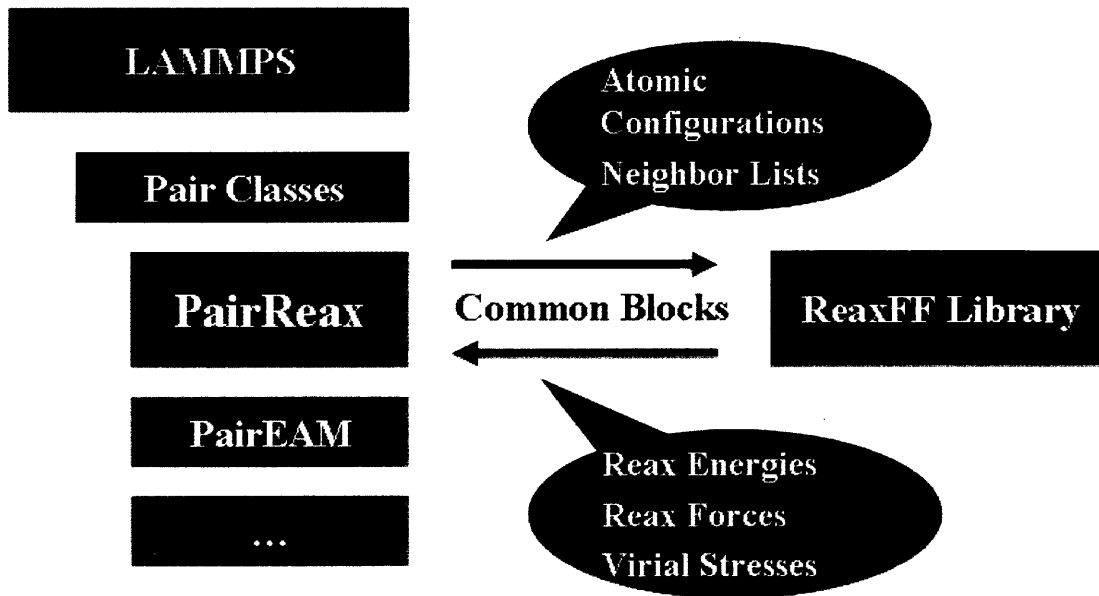


Figure 8-2 Description of structures and flows of data of ReaxFF in LAMMPS

A simple description of calculating the systems using the PairReax class in LAMMPS is given below,

#### A. Initial Atomic Configuration

By reading the initial atomic configurations of systems defined in LAMMPS input data files, the atomic positions of the current system are saved in LAMMPS data structures for the current configuration.

#### B. Neighbor List for the ReaxFF Calculations

After constructing the proper neighbor lists based on NSQ or BIN neighboring style defined and provided in LAMMPS, the Verlet neighbor list which is directly used to calculate the ReaxFF is made from the currently constructed neighbor lists. Finally, for the pair of atoms in the Verlet neighbor list, it is checked whether the pair is within the distance of the maximum bond radius defined in the ReaxFF or not as described in Fig. 3. Here, we used the upper taper radius (10 Å) for the non-bonded interactions as the minimum cutoff radius of the neighbor lists.

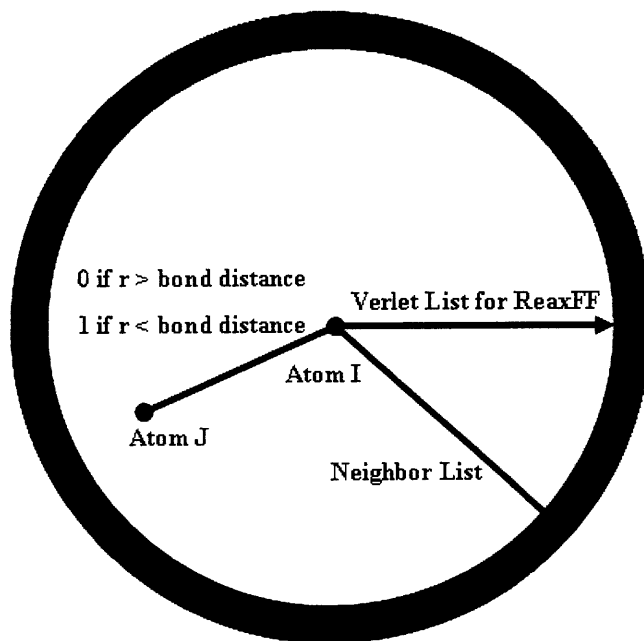


Figure 8-3 Neighbor list for ReaxFF

### C. Data Transferring to the ReaxFF Fortran Library

To use the ReaxFF Fortran code in calculating the ReaxFF energy terms in LAMMPS, we compiled and encapsulated the current ReaxFF Fortran code into a C++ library compatible with the current PairReax class in LAMMPS. After linking the ReaxFF library to the PairReax class in LAMMPS, the constructed Verlet lists with the current atomic configuration in the PairReax class are transferred to the ReaxFF library via Fortran common blocks, which appears as 'extern struct' objects in the C++ code.

### D. Calculation of the ReaxFF Energies

The reactive force field energy terms and related properties such as force on each atom and virial stresses are calculated in the ReaxFF library and transferred back into the PairReax class in LAMMPS via Fortran common blocks.

### E. Interactions of Ghost Atoms

Unlike other non-reactive force field potentials in LAMMPS, extra interactions between ghost atoms of the corresponding local processor should be considered for the purpose of correctly calculating the forces on bonded atoms which are located near borders of two different processors. For the extra

interactions between ghost atoms, construction of the half neighbor list of each ghost atom has been implemented in the PairReax class

## **8.2. Worked examples of RDX and TATB: comparisons with GRASP results**

In this section, we report calculation examples of RDX [C<sub>3</sub>H<sub>6</sub>N<sub>6</sub>O<sub>6</sub>, Cyclotrimethylenetrinitramine] and TATB [C<sub>6</sub>H<sub>6</sub>N<sub>6</sub>O<sub>6</sub>, Triaminotrinitrobenzene] with the implemented PairReax class in LAMMPS. Also, to verify the PairReax class in LAMMPS, the comparison with the calculation results of the same systems using GRASP is presented.

### **8.2.1. NVE Ensemble Simulation of RDX [C<sub>3</sub>H<sub>6</sub>N<sub>6</sub>O<sub>6</sub>].**

Cyclotrimethylenetrinitramine, also known as RDX or T4 is an explosive nitroamine widely used in industrial and military applications. It is material of high chemical energy and, as such, is used as a major component of many plastic-bonded explosive.

#### **Simulation settings**

A single RDX molecule consisting of 21 atoms with periodic boundary conditions

Microcanonical ensemble (Fixed N, V, and E)

Initial temperature: 0K

Step size: 1 femtosecond

Number of steps: 25

Number of processors: 8

Time evolution of potential energy and total energy

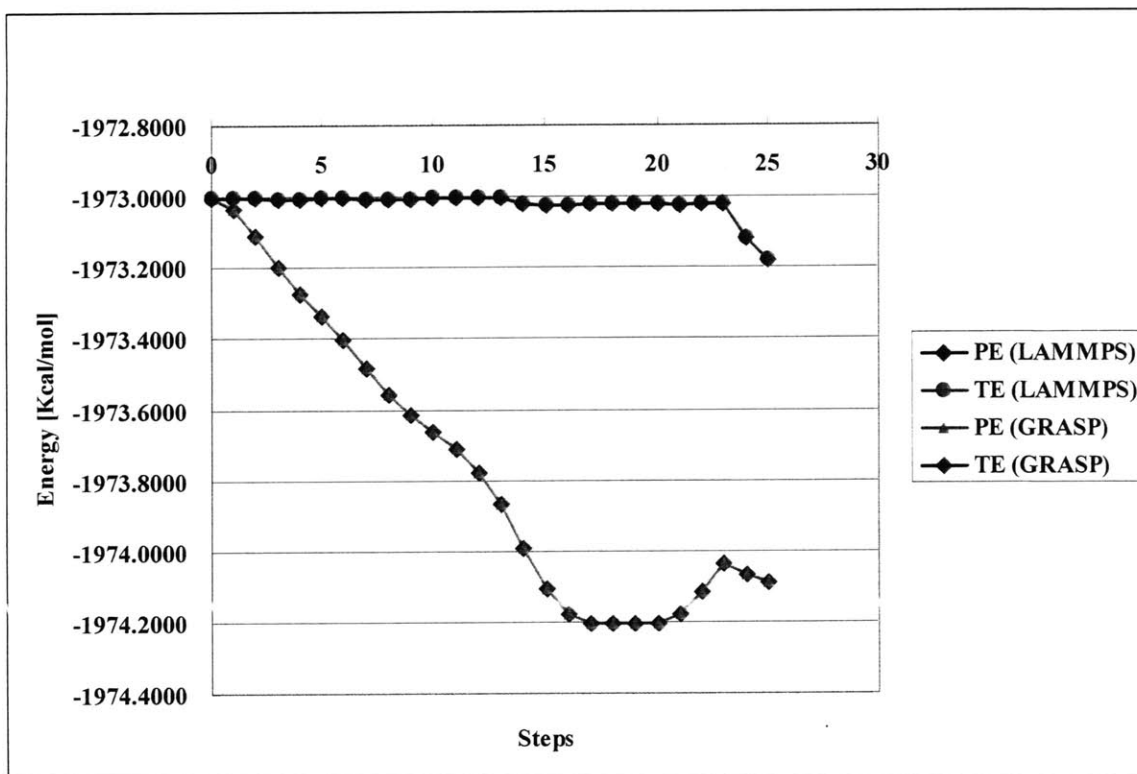


Figure 8-4 Time evolution of potential and total energy of RDX in LAMMPS and GRASP

Step	LAMMPS		GRASP		PE Difference	TE Difference
	PE [Kcal/mol]	TE [Kcal/mol]	PE [Kcal/mol]	TE [Kcal/mol]		
0	-1973.0020538	-1973.0020538	-1973.0020538	-1973.0020538	0.000000000000%	0.000000000000%
1	-1973.0336928	-1973.0031742	-1973.0336928	-1973.0031742	-0.000000000005%	0.000000000000%
2	-1973.1117284	-1973.0055554	-1973.1117284	-1973.0055554	-0.000000000015%	0.000000000000%
3	-1973.2002239	-1973.0071917	-1973.2002239	-1973.0071917	-0.000000000025%	0.000000000000%
4	-1973.2741925	-1973.0069843	-1973.2741925	-1973.0069843	-0.000000000025%	0.000000000000%
5	-1973.3359324	-1973.0057735	-1973.3359324	-1973.0057735	-0.000000000025%	0.000000000000%
6	-1973.4026273	-1973.0053011	-1973.4026273	-1973.0053011	-0.000000000041%	0.000000000000%
7	-1973.4808226	-1973.0061677	-1973.4808226	-1973.0061677	-0.000000000051%	-0.000000000005%
8	-1973.5576228	-1973.0072141	-1973.5576228	-1973.0072141	-0.000000000051%	0.000000000000%
9	-1973.6175083	-1973.0070660	-1973.6175083	-1973.0070660	-0.000000000046%	0.000000000000%
10	-1973.6624605	-1973.0057963	-1973.6624605	-1973.0057963	-0.000000000041%	0.000000000000%
11	-1973.7111667	-1973.0047012	-1973.7111667	-1973.0047012	-0.000000000061%	0.000000000000%
12	-1973.7799912	-1973.0048060	-1973.7799912	-1973.0048060	-0.000000000091%	0.000000000000%
13	-1973.8673964	-1973.0059848	-1973.8673964	-1973.0059848	-0.000000000111%	0.000000000000%
14	-1973.9924069	-1973.0227011	-1973.9924069	-1973.0227011	-0.000000000162%	-0.000000000056%
15	-1974.1083367	-1973.0248196	-1974.1083367	-1973.0248196	-0.000000000127%	-0.000000000051%
16	-1974.1799386	-1973.0250227	-1974.1799386	-1973.0250227	-0.000000000076%	-0.000000000051%
17	-1974.2040128	-1973.0233564	-1974.2040128	-1973.0233564	-0.000000000030%	-0.000000000051%
18	-1974.2050258	-1973.0217627	-1974.2050258	-1973.0217627	-0.000000000020%	-0.000000000051%
19	-1974.2064339	-1973.0219824	-1974.2064339	-1973.0219824	-0.000000000015%	-0.000000000051%
20	-1974.2052504	-1973.0235102	-1974.2052504	-1973.0235102	0.000000000010%	-0.000000000056%
21	-1974.1792541	-1973.0242713	-1974.1792541	-1973.0242713	0.000000000076%	-0.000000000051%
22	-1974.1172840	-1973.0231034	-1974.1172840	-1973.0231034	0.000000000127%	-0.000000000051%
23	-1974.0366823	-1973.0209642	-1974.0366823	-1973.0209642	0.000000000132%	-0.000000000051%
24	-1974.0658423	-1973.1195759	-1974.0658423	-1973.1195759	0.000000000106%	-0.000000000030%
25	-1974.0895267	-1973.1806062	-1974.0895267	-1973.1806062	0.000000000041%	-0.000000000035%

In the table, we can see the initial potential and total energies of the system in LAMMPS are the same as those in GRASP, exactly. However, as step increases, slight energy differences occur between LAMMPS and GRASP. The differences are under  $1 \times 10^{-10}$ % of the energy as a machine error. Also, it does not increase as step increases. The difference might be accumulated from the difference of the simulation schemes in LAMMPS and GRASP every time step. Also, though GRASP constructs the neighbor lists of the ghost atoms to calculate the interactions between ghost atoms on the given neighboring steps, LAMMPS counts and calculates the interactions every time step. Based on these analyses, we conclude the ReaxFF calculations of a single RDX molecule in LAMMPS are reliable, compared with those in GRASP.

### 8.2.2. NVE Ensemble simulation of TATB [C<sub>6</sub>H<sub>6</sub>N<sub>6</sub>O<sub>6</sub>]

Triaminotrinitrobenzene, also known as TATB is a powerful explosive, but it is extremely insensitive to shock, vibration and impact. Because it is so difficult to detonate by accident, even under severe environments, it has become preferred for applications where extreme safety is required.

#### Simulation settings

16 TATB molecules consisting of 384 atoms with periodic boundary conditions

Microcanonical ensemble (Fixed N, V, and E)

Initial temperature: 0K

Step size: 0.0625 femtosecond

Number of steps: 25

Number of processors: 8

#### Time evolution of potential energy and total energy

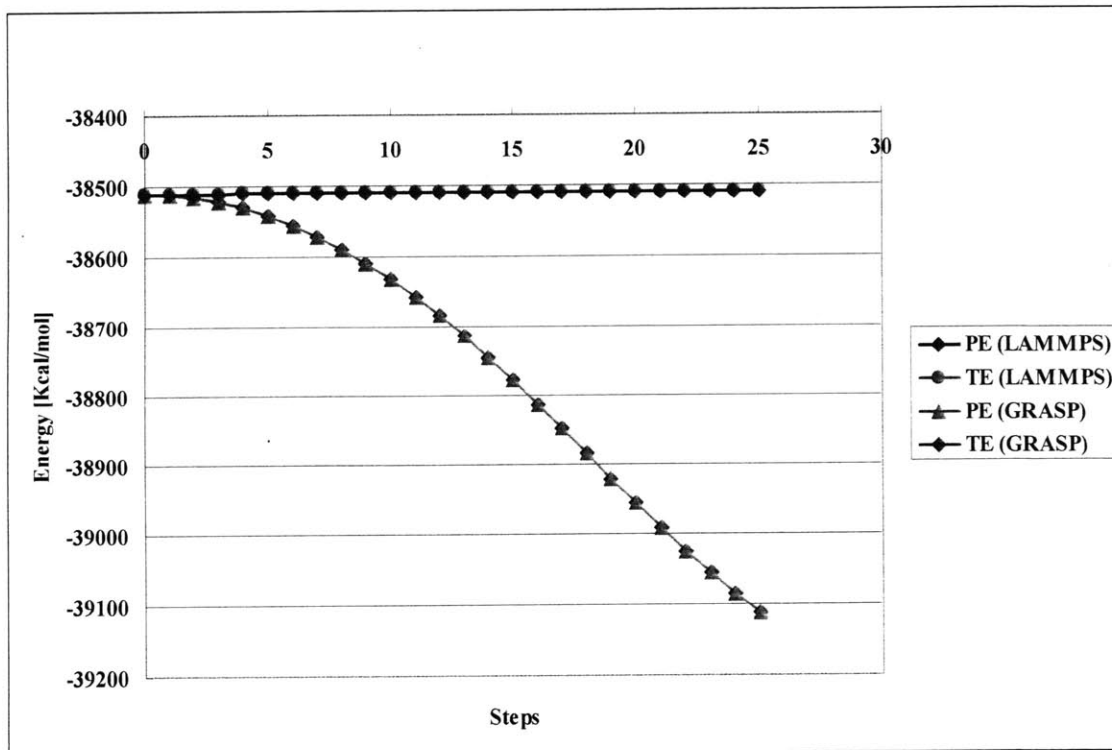


Figure 8-5 Time evolution of potential and total energy of TATB in LAMMPS and GRASP

Step	LAMMPS		GRASP		PE Difference	TE Difference
	PE [Kcal/mol]	TE [Kcal/mol]	PE [Kcal/mol]	TE [Kcal/mol]		
0	-38511.0566045	-38511.0566045	-38511.0566045	-38511.0566045	0.000000000000%	0.000000000000%
1	-38512.3015035	-38511.0567514	-38512.3015035	-38511.0567514	-0.000000000013%	0.000000000001%
2	-38516.0338511	-38511.0571913	-38516.0338511	-38511.0571913	-0.000000000047%	0.000000000000%
3	-38522.2466138	-38511.0579227	-38522.2466138	-38511.0579227	-0.000000000108%	0.000000000001%
4	-38530.7093372	-38510.8401722	-38530.7093371	-38510.8401722	-0.000000000190%	-0.000000000001%
5	-38541.6246079	-38510.6228058	-38541.6246077	-38510.6228058	-0.000000000296%	0.000000000000%
6	-38555.1900952	-38510.6244332	-38555.1900951	-38510.6244332	-0.000000000424%	-0.000000000002%
7	-38571.1613612	-38510.6264065	-38571.1613609	-38510.6264065	-0.000000000572%	0.000000000000%
8	-38589.5074321	-38510.6288012	-38589.5074318	-38510.6288012	-0.000000000745%	-0.000000000001%
9	-38610.1911250	-38510.6317605	-38610.1911246	-38510.6317605	-0.000000000937%	0.000000000001%
10	-38632.7777298	-38510.2472355	-38632.7777294	-38510.2472355	-0.000000001149%	0.000000000000%
11	-38657.7733620	-38510.0384060	-38657.7733614	-38510.0384060	-0.000000001378%	-0.000000000002%
12	-38684.9684461	-38509.8705381	-38684.9684455	-38509.8705381	-0.000000001622%	0.000000000000%
13	-38714.3977400	-38509.8803653	-38714.3977393	-38509.8803653	-0.000000001881%	-0.000000000001%
14	-38745.7476759	-38509.8939941	-38745.7476750	-38509.8939941	-0.000000002146%	0.000000000000%
15	-38778.8229716	-38509.9134001	-38778.8229706	-38509.9134001	-0.000000002411%	-0.000000000003%
16	-38813.3529168	-38509.9396245	-38813.3529158	-38509.9396245	-0.000000002667%	-0.000000000004%
17	-38848.2356173	-38509.2334103	-38848.2356162	-38509.2334103	-0.000000002903%	-0.000000000003%
18	-38884.4978776	-38509.2710595	-38884.4978764	-38509.2710595	-0.000000003103%	-0.000000000005%
19	-38921.0395136	-38509.4706956	-38921.0395123	-38509.4706956	-0.000000003259%	-0.000000000005%
20	-38956.9712970	-38509.5027320	-38956.9712957	-38509.5027320	-0.000000003359%	-0.000000000007%
21	-38991.9074280	-38509.5390831	-38991.9074267	-38509.5390831	-0.000000003396%	-0.000000000006%
22	-39025.3862120	-38509.6393572	-39025.3862107	-38509.6393572	-0.000000003372%	-0.000000000004%
23	-39056.5033776	-38509.3384553	-39056.5033763	-38509.3384553	-0.000000003289%	-0.000000000005%
24	-39085.5333790	-38509.2370981	-39085.5333778	-38509.2370981	-0.000000003155%	-0.000000000003%
25	-39111.9896646	-38509.0370844	-39111.9896635	-38509.0370844	-0.000000002989%	0.000000000002%

In the above table, we can see the initial potential energy and total energies of the system in LAMMPS are the same as those in GRASP, exactly. However, as step increases, there exist less than  $1 \times 10^{-10}\%$  slight energy differences occur between LAMMPS and GRASP. This discrepancy also does not increase as time step increases. As discussed in the RDX example, the difference might come from the differences in the simulation schemes in LAMMPS and GRASP. Based on these analyses, we can conclude the ReaxFF calculations of TATB molecules in LAMMPS are reliable, compared with those in GRASP.



### **8.3. Conclusion: Further research focus on the ReaxFF on Nickel, Aluminum, and Oxygen systems**

Currently, the ReaxFF parameters on the single phase of Ni and Al, and their interactions with Oxygen are under development. However, to our knowledge, the interaction between Ni and Al hybrid systems has not been developed at all. Based on this SM thesis research, we will focus on the oxidation processes of Ni and Al nanoparticles with oxygen-rich environments to create the native oxide of Ni and Al nanoparticles based on the ReaxFF. Also, with ReaxFF, we will model the polymeric systems such as epoxy networks that will contain the oxide nanoparticles as fillers. After the construction of such fully combined systems of Ni and Al oxide nanoparticles, and polymer matrix based on the ReaxFF, we will investigate the physical and chemical behaviors of the whole systems under the extreme thermal and mechanical environments such as shock loading. For the near-term research for the polymer nanocomposite systems, the currently implemented and parallelized version of the ReaxFF would play a key role and directly used in the atomistic and molecular modeling on it.

As a result of the work described in this chapter, the currently implemented ReaxFF potentials with the parallelized charge equilibration algorithms for the non-bonded interactions were officially issued to the public via the LAMMPS official website. (<http://lammps.sandia.gov/authors.html>)

## Chapter 9

### Summary and Conclusion

#### 9.1. Summary

In this thesis, the following were investigated and analyzed using a variety of continuum and molecular dynamics simulations.

A. Self-sustained intermixing exothermic reaction propagations in the Nickel and Aluminum laminates were modeled and numerically simulated with the coupled partial differential equations for the reactive heat conduction and mass diffusion. By imposing the initial peak temperature over the effective adiabatic temperature, 1830K at the boundary, the reaction was initiated and propagated through the laminates with the self-sustained manner. The reaction front velocity, reaction zone width, affects of the premixed region and ambient temperature calculated from the numerical simulations were well matched to current experimental data.

B. Embedded Atomic Method interatomic potentials for the Nickel and Aluminum hybrid systems and ReaxFF for C, H, O, N and Si systems were implemented and parallelized as internal classes of LAMMPS in C++ language, and distributed to public via LAMMPS official website (<http://lammps.sandia.gov>).

C. Based on the implemented EAM potentials for the Nickel and Aluminum systems, the Nickel and Aluminum nanoslabs were equilibrated for the given experimental conditions. After the equilibration processes, the impact loading simulations for the Nickel and Aluminum nanoslabs were performed under high speed impact velocities. With the impact simulations, over 1200 m/sec impact velocity (0.234 eV per atom) and tens of GPa pressure, the intermixing between the Ni and Al nanoslabs was initiated even in the solid state at 800K temperature below.

D. For the same size of the system, Nickel and Aluminum nanoparticles having large surface area to volume ratios were constructed and equilibrated. After the equilibration processes, the impact loading simulations for the nanoparticles were performed with the same impact speed as in the nanoslab cases. With the impact simulations, over 1200 m/sec impact velocity (0.234 eV per atom), the Ni and Al nanoparticles were fully mixed. The complete processes included melting of the Ni and Al nanoparticles and coalescence process. Also, the effect of the nanoparticle size and surface roughness were analyzed in terms of the whole process time. It was observed that the final temperature after the whole process of the intermixing initiation, melting, and coalescence settled to 1800K, which is very close to the adiabatic temperature used for the intermixing initiation in the continuum calculations.

## 9.2. Future research plan

Our final objective for this project is to construct a complete set of computational models for reactive polymer nanocomposites, and suggest design principles based on the constructed computational models in a wide range of time and length scales for various applications and demands. In this thesis, preliminary research on the Nickel and Aluminum systems was completed. Based on this preliminary research, we will focus on the following:

- A. Oxidation processes for the Nickel and Aluminum nanoparticles and construct their native oxide nanoparticles based on the ReaxFF potential.
- B. Impact loading processes for the Nickel and Aluminum oxide nanoparticles: cohesive energy between oxygen and metal is very high. Therefore, the oxide layers surrounding the core nanoparticles are predicted as chemomechanical shields for the intermixing between the Ni and Al atoms. Also, before considering the intermixing between the metals, it would be required to investigate the surface properties of the oxide nanoparticles in terms of melting points and mechanical properties.
- C. Computational models for the interfaces between three different phases of polymer, oxygen, and Nickel/Aluminum; as discussed, the nanoparticle (or oxidized nanoparticle) would be filled into the polymer

matrices such as epoxy networks. Based on the ReaxFF potentials, the complex properties of the interfacial phases would be investigated.

D. Design principles of the reactive polymer nanocomposites: based on the computational models for the reactive polymer nanocomposites with the multi-scale approaches, we will suggest the design principles for the nanocomposites for various demands and applications, especially those defined by the extreme thermal and mechanical stimuli.

## APPENDIX A. C++ Class for the ReaxFF implementation in LAMMPS

### Pair\_reax.h

```
#ifndef PAIR_REAX_H
#define PAIR_REAX_H

#include "pair.h"

namespace LAMMPS_NS {

class PairREAX : public Pair {
public:
  PairREAX(class LAMMPS *);
  ~PairREAX();

  void compute(int, int);
  void settings(int, char **);
  void coeff(int, char **);
  void init_style();
  double init_one(int, int);
  double memory_usage();

  int pack_comm(int, int *, double *, int, int *);
  void unpack_comm(int, int, double *);
  int pack_reverse_comm(int, int, double *);
  void unpack_reverse_comm(int, int *, double *);

private:
  double cutmax;
  double rcutvsq,rcutbsq;
  int iprune,ihb;
  double hbcut,swb;
  double swa;
  double swc0, swc1, swc2, swc3, swc4, swc5, swc6, swc7;
  double precision;
  int packflag;

  struct ff_params {
    double rcutsq;
    int np;
    double *params;
  };
  ff_params *param_list;
  int *map;

  int nentries;
  double chpot;
  int *arow_ptr,*acol_ind;
  double *ch,*elcvec;
  double *rcg,*wcg,*pcg,*poldcg,*qcg;
  double *aval;
  int nmax,matmax;

  void allocate();
  void read_files(char *, char *);
  void neigh_f2c(int, int *, int *, int **);
  void neigh_c2f(int, int *, int *, int **);

  void write_reax_positions();
  void write_reax_vlist();
  void read_reax_forces();
  void read_reax_atom_virial();

  void taper_setup();
  double taper_E(const double &, const double &);
  double taper_F(const double &, const double &);

  void compute_charge(double &);

```

```

void sparse_product(const int &, const int &, const int &, double[],
                  int[], int[], double[], double[]);
void cg_solve(const int &, const int &, double[], int[],
              int[], double[], double[]);
void charge_reax(const int &, const int &, double[],
                 double[], int[], int[], double[]);
void output_itemized_energy(double);
};

}

#endif

```

## Pair\_reax.cpp

```

/* -----
   Contributing authors: Aidan Thompson (Sandia, athomps@sandia.gov)
                       Hansohl Cho (MIT, hansohl@mit.edu)
   LAMMPS implementation of the Reactive Force Field (ReaxFF) is Ardi Van Duin's original ReaxFF code
   ----- */

#include "mpi.h"
#include "math.h"
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "pair_reax.h"
#include "pair_reax_fortran.h"
#include "atom.h"
#include "update.h"
#include "force.h"
#include "comm.h"
#include "neighbor.h"
#include "neigh_list.h"
#include "neigh_request.h"
#include "memory.h"
#include "error.h"

using namespace LAMMPS_NS;

#define MIN(a,b) ((a) < (b) ? (a) : (b))
#define MAX(a,b) ((a) > (b) ? (a) : (b))
#define SMALL 0.0001

/* ----- */

PairREAX::PairREAX(LAMMPS *Imp) : Pair(Imp)
{
  single_enable = 0;
  one_coeff = 1;
  no_virial_compute = 1;

  cutmax = 0.0;
  hbcut = 6.0;
  iprune = 4;
  ihb = 1;
  chpot = 0;

  nmax = 0;
  arow_ptr = NULL;
  ch = NULL;
  elcvec = NULL;
  rcg = NULL;
  wcg = NULL;
  pcg = NULL;
  poldcg = NULL;
  qcg = NULL;

  matmax = 0;

```

```

aval = NULL;
acol_ind = NULL;

comm_forward = 1;
comm_reverse = 1;

precision = 1.0e-6;
}

/* -----
   free all arrays
   check if allocated, since class can be destructed when incomplete
   ----- */

PairREAX::~PairREAX()
{
  if (allocated) {
    memory->destroy_2d_int_array(setflag);
    memory->destroy_2d_double_array(cutsq);

    for (int i = 1; i <= atom->ntypes; i++)
      delete [] param_list[i].params;
    delete [] param_list;

    delete [] map;
  }

  memory->sfree(arrow_ptr);
  memory->sfree(ch);
  memory->sfree(elcvec);
  memory->sfree(rcg);
  memory->sfree(wcg);
  memory->sfree(pcg);
  memory->sfree(poldcg);
  memory->sfree(qcg);

  memory->sfree(aval);
  memory->sfree(acol_ind);
}

/* ----- */

void PairREAX::compute(int eflag, int vflag)
{
  int i,j;
  double evdw,ecoul;
  double energy_charge_equilibration;

  evdw = ecoul = 0.0;
  if (eflag || vflag) ev_setup(eflag,vflag);
  else evflag = vflag_fdotr = 0;

  if (vflag_global) FORTRAN(cbkvirial, CBKVIRIAL).Lvirial = 1;
  else FORTRAN(cbkvirial, CBKVIRIAL).Lvirial = 0;

  if (vflag_atom) FORTRAN(cbkvirial, CBKVIRIAL).Latomvirial = 1;
  else FORTRAN(cbkvirial, CBKVIRIAL).Latomvirial = 0;

  // reallocate charge equilibration and CG arrays if necessary

  if (atom->nmax > nmax) {
    memory->sfree(rcg);
    memory->sfree(wcg);
    memory->sfree(pcg);
    memory->sfree(poldcg);
    memory->sfree(qcg);

    nmax = atom->nmax;
    int n = nmax+1;
  }
}

```

```

arow_ptr = (int *) memory->smalloc(n*sizeof(int),"reax:arow_ptr");
ch = (double *) memory->smalloc(n*sizeof(double),"reax:ch");
elcvec = (double *) memory->smalloc(n*sizeof(double),"reax:elcvec");
rcg = (double *) memory->smalloc(n*sizeof(double),"reax:rcg");
wcg = (double *) memory->smalloc(n*sizeof(double),"reax:wcg");
pcg = (double *) memory->smalloc(n*sizeof(double),"reax:pcg");
poldcg = (double *) memory->smalloc(n*sizeof(double),"reax:poldcg");
qcg = (double *) memory->smalloc(n*sizeof(double),"reax:qcg");
}

// calculate the atomic charge distribution

compute_charge(energy_charge_equilibration);

// transfer LAMMPS positions and neighbor lists to REAX

write_reax_positions();
write_reax_vlist();

// determine whether this bond is owned by the processor or not

FORTRAN(srtbon1, SRTBON1>(&iprune, &ihb, &hbcut);

// communicate with other processors for the atomic bond order calculations

FORTRAN(cbkabo, CBKABO).abo;

// communicate local atomic bond order to ghost atomic bond order

packflag = 0;
comm->comm_pair(this);

FORTRAN(molec, MOLEC());
FORTRAN(encalc, ENCALC());
FORTRAN(mdsav, MDSAV>(&comm->me);

// read forces from ReaxFF Fortran

read_reax_forces();

// extract global and per-atom energy from ReaxFF Fortran
// compute_charge already contributed to eatom

if (eflag_global) {
evdwl += FORTRAN(cbkenergies, CBKENERGIES).eb;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).ea;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).elp;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).emol;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).ev;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).epen;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).ecoa;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).ehb;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).et;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).eco;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).ew;
evdwl += FORTRAN(cbkenergies, CBKENERGIES).efi;

ecoul += FORTRAN(cbkenergies, CBKENERGIES).ep;
ecoul += energy_charge_equilibration;

eng_vdwl += evdwl;
eng_coul += ecoul;
}

if (eflag_atom) {
int ntotal = atom->nlocal + atom->nghost;
for (i = 0; i < ntotal; i++)
eatom[i] += FORTRAN(cbkcd,CBKCD).estrain[i];
}

```



```

// extract global and per-atom virial from ReaxFF Fortran

if (vflag_global) {
virial[0] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[0];
virial[1] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[1];
virial[2] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[2];
virial[3] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[3];
virial[4] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[4];
virial[5] = -FORTRAN(cbkvirial, CBKVIRIAL).virial[5];
}

if (vflag_atom) {
int ntotal = atom->nlocal + atom->nghost;
j = 0;
for (i = 0; i < ntotal; i++) {
vatom[i][0] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+0];
vatom[i][1] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+1];
vatom[i][2] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+2];
vatom[i][3] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+3];
vatom[i][4] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+4];
vatom[i][5] = -FORTRAN(cbkvirial, CBKVIRIAL).atomvirial[j+5];
j += 6;
}
}
}

/* ----- */

void PairREAX::write_reax_positions()
{
double xtmp, ytmp, ztmp;
int j, jx, jy, jz, jia;

double **x = atom->x;
double *q = atom->q;
int *type = atom->type;
int *tag = atom->tag;
int nlocal = atom->nlocal;
int nghost = atom->nghost;

FORTRAN(rsmall, RSMALL).na = nlocal+nghost;
FORTRAN(rsmall, RSMALL).na_local = nlocal;

jx = 0;
jy = ReaxParams::nat;
jz = 2*ReaxParams::nat;
jia = 0;

j = 0;
for (int i = 0; i < nlocal+nghost; i++, j++) {
FORTRAN(cbkcc, CBKCC).c[j+jx] = x[i][0];
FORTRAN(cbkcc, CBKCC).c[j+jy] = x[i][1];
FORTRAN(cbkcc, CBKCC).c[j+jz] = x[i][2];
FORTRAN(cbkch, CBKCH).ch[j] = q[i];
FORTRAN(cbkia, CBKIA).ia[j+jia] = map[type[i]];
FORTRAN(cbkia, CBKIA).iag[j+jia] = map[type[i]];
FORTRAN(cbkcc, CBKCC).itag[j] = tag[i];
}
}

/* ----- */

void PairREAX::write_reax_vlist()
{
int ii, jj, i, j, iii, jjj;
double xitmp, yitmp, zitmp;
double xjtmp, yjtmp, zjtmp;
int itag, jtag;
int nvpair, nvself, nvpairmax;
int nbond;

```

```

int inum,jnum;
int *ilist;
int *jlist;
int *numneigh,**firstneigh;
double delr2;
double delx, dely, delz;
double rtmp[3];

double **x = atom->x;
int *tag = atom->tag;
int nlocal = atom->nlocal;
int nghost = atom->nghost;

nvpairmax = ReaxParams::nneighmax * ReaxParams::nat;

nvpair = 0;
nvlself = 0;
nbond = 0;

inum = list->inum;
ilist = list->ilist;
numneigh = list->numneigh;
firstneigh = list->firstneigh;

for (ii = 0; ii < inum; ii++) {
    i = ilist[ii];
    xitmp = x[i][0];
    yitmp = x[i][1];
    zitmp = x[i][2];
    itag = tag[i];
    jlist = firstneigh[i];
    jnum = numneigh[i];

    for (jj = 0; jj < jnum; jj++) {
        j = jlist[jj];
        xjtmp = x[j][0];
        yjtmp = x[j][1];
        zjtmp = x[j][2];
        jtag = tag[j];

        delx = xitmp - xjtmp;
        dely = yitmp - yjtmp;
        delz = zitmp - zjtmp;

        delr2 = delx*delx+dely*dely+delz*delz;

        if (delr2 <= rcutvsq) {
            if (i < j) {
                iii = i+1;
                jjj = j+1;
            } else {
                iii = j+1;
                jjj = i+1;
            }
            if (nvpair >= nvpairmax) break;

            FORTRAN(cbkpairs, CBKPAIRS).nv11[nvpair] = iii;
            FORTRAN(cbkpairs, CBKPAIRS).nv12[nvpair] = jjj;
            FORTRAN(cbknvlbo, CBKNVLBO).nvlbo[nvpair] = 0;

            if (delr2 <= rcutbsq) {
                FORTRAN(cbknvlbo, CBKNVLBO).nvlbo[nvpair] = 1;
                nbond++;
            }

            FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 0;

            if (j < nlocal)
                FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 1;
            else if (itag < jtag)

```

```

        FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 1;
    else if (itag == jtag) {
        if (delz > SMALL)
            FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 1;
        else if (fabs(delz) < SMALL) {
            if (dely > SMALL)
                FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 1;
            else if (fabs(dely) < SMALL && delx > SMALL)
                FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 1;
        }
    }
    nvpair++;
}
}
}

```

```
int ntotal = nlocal + nghost;
```

```

for (int i = nlocal; i < ntotal; i++) {
    xitmp = x[i][0];
    yitmp = x[i][1];
    zitmp = x[i][2];
    itag = tag[i];

```

```

    for (int j = i+1; j < ntotal; j++) {
        xjtmp = x[j][0];
        yjtmp = x[j][1];
        zjtmp = x[j][2];
        jtag = tag[j];

```

```

        delx = xitmp - xjtmp;
        dely = yitmp - yjtmp;
        delz = zitmp - zjtmp;

```

```
delr2 = delx*delx+dely*dely+delz*delz;
```

```
// don't need to check the double count since i < j in the ghost region
```

```

if (delr2 <= rcutvsq) {
    iii = i+1;
    jjj = j+1;

```

```
    if (nvpair >= nvpairmax) break;
```

```

    FORTRAN(cbkpairs, CBKPAIRS).nv11[nvpair] = iii;
    FORTRAN(cbkpairs, CBKPAIRS).nv12[nvpair] = jjj;
    FORTRAN(cbknvlbo, CBKNVLBO).nvlbo[nvpair] = 0;

```

```

    if (delr2 <= rcutbsq) {
        FORTRAN(cbknvlbo, CBKNVLBO).nvlbo[nvpair] = 1;
        nbond++;
    }

```

```

    FORTRAN(cbknvlown, CBKNVLOWN).nvlown[nvpair] = 0;
    nvpair++;
}
}
}

```

```

FORTRAN(cbkpairs, CBKPAIRS).nvpair = nvpair;
FORTRAN(cbkpairs, CBKPAIRS).nvself = nvself;
}

```

```
/* ----- */
```

```
void PairREAX::read_reax_forces()
```

```
{
    double ftmp[3];
```

```
    double **f = atom->f;
```

```

int ntotal = atom->nlocal + atom->nghost;

int j = 0;
for (int i = 0; i < ntotal; i++) {
    ftmp[0] = -FORTRAN(cbkd, CBKD).d[j];
    ftmp[1] = -FORTRAN(cbkd, CBKD).d[j+1];
    ftmp[2] = -FORTRAN(cbkd, CBKD).d[j+2];
    f[i][0] = ftmp[0];
    f[i][1] = ftmp[1];
    f[i][2] = ftmp[2];
    j += 3;
}
}
/* ----- */

void PairREAX::allocate()
{
    allocated = 1;
    int n = atom->ntypes;

    setflag = memory->create_2d_int_array(n+1,n+1,"pair:setflag");
    cutsq = memory->create_2d_double_array(n+1,n+1,"pair:cutsq");

    param_list = new ff_params[n+1];
    for (int i = 1; i <= n; i++)
        param_list[i].params = new double[5];

    map = new int[n+1];
}

/* -----
   global settings
   ----- */

void PairREAX::settings(int narg, char **arg)
{
    if (narg != 0 && narg != 2) error->all("Illegal pair_style command");

    if (narg == 2) {
        hbcut = atof(arg[0]);
        precision = atof(arg[1]);

        if (hbcut <= 0.0 || precision <= 0.0)
            error->all("Illegal pair_style command");
    }
}

/* -----
   set coeffs for one or more type pairs
   ----- */

void PairREAX::coeff(int narg, char **arg)
{
    if (!allocated) allocate();

    if (narg != 3 + atom->ntypes)
        error->all("Incorrect args for pair coefficients");

    // insure I,J args are * *
    if (strcmp(arg[0],"*") != 0 || strcmp(arg[1],"*") != 0)
        error->all("Incorrect args for pair coefficients");

    // insure filename isffield.reax
    if (strcmp(arg[2],"ffield.reax") != 0)
        error->all("Incorrect args for pair coefficients");

    // read args that map atom types to elements in potential file

```

```

// map[i] = which element the lth atom type is, -1 if NULL

for (int i = 3; i < narg; i++) {
  if (strcmp(arg[i], "NULL") == 0) {
    map[i-2] = -1;
    continue;
  }
  map[i-2] = atoi(arg[i]);
}

int n = atom->ntypes;

int count = 0;
for (int i = 1; i <= n; i++)
  for (int j = i; j <= n; j++) {
    setflag[i][j] = 1;
    count++;
  }

if (count == 0) error->all("Incorrect args for pair coefficients");
}

/* -----
init specific to this pair style
----- */

void PairREAX::init_style()
{
  if (atom->tag_enable == 0)
    error->all("Pair style reax requires atom IDs");
  if (force->newton_pair == 0)
    error->all("Pair style reax requires newton pair on");
  if (strcmp(update->unit_style, "real") != 0 && comm->me == 0)
    error->warning("Not using real units with pair reax");

  int irequest = neighbor->request(this);
  neighbor->requests[irequest]->newton = 2;

  FORTRAN(readc, READC)();
  FORTRAN(reaxinit, REAXINIT)();
  FORTRAN(ffinpt, FFINPT)();
  FORTRAN(tap7th, TAP7TH)();

  // turn off read_in by fort.3 in REAX Fortran

  int ngeofor_tmp = -1;
  FORTRAN(setngeofor, SETNGEOFOR>(&ngeofor_tmp);
  if (comm->me == 0) FORTRAN(readgeo, READGEO)();

  // initial setup for cutoff radius of VLIST and BLIST in ReaxFF

  double vlbora;

  FORTRAN(getswb, GETSWB>(&swb);
  cutmax=MAX(swb, hbcut);
  rcutvsq=cutmax*cutmax;
  FORTRAN(getvlbora, GETVLBORA>(&vlbora);
  rcutbsq=vlbora*vlbora;

  // parameters for charge equilibration from ReaxFF input, fort.4
  // verify that no LAMMPS type to REAX type mapping was invalid

  int nelements;
  FORTRAN(getnso, GETNSO>(&nelements);

  FORTRAN(getswa, GETSWA>(&swa);
  double chi, eta, gamma;
  for (int itype = 1; itype <= atom->ntypes; itype++) {
    if (map[itype] < 1 || map[itype] > nelements)
      error->all("Invalid REAX atom type");

```

```

chi = FORTRAN(cbkchb, CBKCHB).chi[map[itype]-1];
eta = FORTRAN(cbkchb, CBKCHB).eta[map[itype]-1];
gamma = FORTRAN(cbkchb, CBKCHB).gam[map[itype]-1];
param_list[itype].np = 5;
param_list[itype].rcutsq = cutmax;
param_list[itype].params[0] = chi;
param_list[itype].params[1] = eta;
param_list[itype].params[2] = gamma;
param_list[itype].params[3] = swa;
param_list[itype].params[4] = swb;
}

taper_setup();
}

/* -----
init for one type pair i,j and corresponding j,i
----- */

double PairREAX::init_one(int i, int j)
{
return cutmax;
}

/* ----- */

int PairREAX::pack_comm(int n, int *list, double *buf, int pbc_flag, int *pbc)
{
int i,j,m;

m = 0;

if (packflag == 0) {
for (i = 0; i < n; i++) {
j = list[i];
buf[m++] = FORTRAN(cbkabo, CBKABO).abo[j];
}
} else {
for (i = 0; i < n; i++) {
j = list[i];
buf[m++] = wcg[j];
}
}

return 1;
}

/* ----- */

void PairREAX::unpack_comm(int n, int first, double *buf)
{
int i,m,last;

m = 0;
last = first + n;

if (packflag == 0) {
for (i = first; i < last; i++)
FORTRAN(cbkabo, CBKABO).abo[i] = buf[m++];
} else {
for (i = first; i < last; i++)
wgc[i] = buf[m++];
}
}

/* ----- */

int PairREAX::pack_reverse_comm(int n, int first, double *buf)

```

```

{
  int i,k,m,last,size;

  m = 0;
  last = first + n;
  for (i = first; i < last; i++)
    buf[m++] = wcg[i];

  return 1;
}

/* ----- */

void PairREAX::unpack_reverse_comm(int n, int *list, double *buf)
{
  int i,j,k,m;

  m = 0;
  for (i = 0; i < n; i++) {
    j = list[i];
    wcg[j] += buf[m++];
  }
}

/* ----- */
charge equilibration routines
/* ----- */

/* ----- */

void PairREAX::taper_setup()
{
  double swb2,swa2,swb3,swa3,d1,d7;

  d1=swb-swa;
  d7=pow(d1,7);
  swa2=swa*swa;
  swa3=swa2*swa;
  swb2=swb*swb;
  swb3=swb2*swb;

  swc7= 20.0e0/d7;
  swc6= -70.0e0*(swa+swb)/d7;
  swc5= 84.0e0*(swa2+3.0e0*swa*swb+swb2)/d7;
  swc4= -35.0e0*(swa3+9.0e0*swa2*swb+9.0e0*swa*swb2+swb3)/d7;
  swc3= 140.0e0*(swa3*swb+3.0e0*swa2*swb2+swa*swb3)/d7;
  swc2=-210.0e0*(swa3*swb2+swa2*swb3)/d7;
  swc1= 140.0e0*swa3*swb3/d7;
  swc0=(-35.0e0*swa3*swb2*swb2+21.0e0*swa2*swb3*swb2+
        7.0e0*swa*swb3*swb3+swb3*swb3*swb)/d7;
}

/* ----- */

double PairREAX::taper_E(const double &r, const double &r2)
{
  double r3=r2*r;
  return swc7*r3*r3+r+swc6*r3*r3+swc5*r3*r2+swc4*r2*r2+swc3*r3+swc2*r2+
        swc1*r+swc0;
}

/* ----- */

double PairREAX::taper_F(const double &r, const double &r2)
{
  double r3=r2*r;
  return 7.0e0*swc7*r3*r3+6.0e0*swc6*r3*r2+5.0e0*swc5*r2*r2+
        4.0e0*swc4*r3+3.0e0*swc3*r2+2.0e0*swc2*r+swc1;
}

```

```

/* -----
compute current charge distributions based on the charge equilibration
----- */

void PairREAX::compute_charge(double &energy_charge_equilibration)
{
double xitmp, yitmp, zitmp;
double xjtmp, yjtmp, zjtmp;
int itype, jtype, itag, jtag;
int ii, jj, i, j;
double delr2, delr_norm, gamt, hulp1, hulp2;
double delx, dely, delz;
double qsum,qi;
int nmatentries;
double sw;
double rtmp[3];
int inum,jnum;
int *ilist;
int *jlist;
int *numneigh,**firstneigh;

double **x = atom->x;
double *q = atom->q;
int *type = atom->type;
int *tag = atom->tag;

int nlocal = atom->nlocal;
int nghost = atom->nghost;

inum = list->inum;
ilist = list->ilist;
numneigh = list->numneigh;
firstneigh = list->firstneigh;

// realloc neighbor based arrays if necessary

int numneigh_total = 0;
for (ii = 0; ii < inum; ii++)
    numneigh_total += numneigh[ilist[ii]];

if (numneigh_total + 2*nlocal > matmax) {
    memory->sfree(aval);
    memory->sfree(acol_ind);
    matmax = numneigh_total + 2*nlocal;
    aval = (double *) memory->smalloc(matmax*sizeof(double),"reax:aval");
    acol_ind = (int *) memory->smalloc(matmax*sizeof(int),"reax:acol_ind");
}

// build linear system

nmatentries = 0;

for (ii = 0; ii < inum; ii++) {
    i = ilist[ii];
    xitmp = x[i][0];
    yitmp = x[i][1];
    zitmp = x[i][2];
    itype = type[i];
    itag = tag[i];
    jlist = firstneigh[i];
    jnum = numneigh[i];

    arow_ptr[i] = nmatentries;
    aval[nmatentries] = 2.0*param_list[itype].params[1];
    acol_ind[nmatentries] = i;
    nmatentries++;

    aval[nmatentries] = 1.0;
    acol_ind[nmatentries] = nlocal + nghost;
    nmatentries++;
}

```



```

for (jj = 0; jj < jnum; jj++) {
  j = jlist[jj];
  xjtmp = x[j][0];
  yjtmp = x[j][1];
  zjtmp = x[j][2];
  jtype = type[j];
  jtag = tag[j];

  delx = xitmp - xjtmp;
  dely = yitmp - yjtmp;
  delz = zitmp - zjtmp;

  delr2 = delx*delx+dely*dely+delz*delz;

  // avoid counting local-ghost pair twice since
  // ReaxFF uses half neigh list with newton off

  if (j >= nlocal) {
    if (itag > jtag) {
      if ((itag+jtag) % 2 == 0) continue;
    } else if (itag < jtag) {
      if ((itag+jtag) % 2 == 1) continue;
    } else {
      if (zjtmp < zitmp) continue;
      if (zjtmp == zitmp && yjtmp < yitmp) continue;
      if (zjtmp == zitmp && yjtmp == yitmp && xjtmp < xitmp) continue;
    }
  }

  // rcutvsq = cutmax*cutmax, in ReaxFF

  if (delr2 <= rcutvsq) {
    gamt = sqrt(param_list[itype].params[2]*param_list[jtype].params[2]);
    delr_norm = sqrt(delr2);
    sw = taper_E(delr_norm, delr2);
    hulp1=(delr_norm*delr2+(1.0/(gamt*gamt*gamt)));
    hulp2=sw*14.40/cbrt(hulp1);
    aval[nmatentries] = hulp2;
    acol_ind[nmatentries] = j;
    nmatentries++;
  }
}

// in this case, we don't use Midpoint method
// so, we don't need to consider ghost-ghost interactions
// but, need to fill the arow_ptr[] arrays for the ghost atoms

for (i = nlocal; i < nlocal+nghost; i++)
  arow_ptr[i] = nmatentries;
arow_ptr[nlocal+nghost] = nmatentries;

// add rhs matentries to linear system

for (ii=0; ii<inum; ii++) {
  i = ilist[ii];
  itype = type[i];
  elcvec[i] = -param_list[itype].params[0];
}

for (i = nlocal; i < nlocal+nghost; i++) elcvec[i] = 0.0;

// assign current charges to charge vector

qsum = 0.0;
for (ii = 0; ii < inum; ii++) {
  i = ilist[ii];
  qi = q[i];
  ch[i] = qi;
}

```

```

    if (i < nlocal) qsum += qi;
}

for (i = nlocal; i < nlocal+nghost; i++) {
    qi = q[i];
    ch[i] = qi;
}

double qtot;
MPI_Allreduce(&qsum,&qtot,1,MPI_DOUBLE,MPI_SUM,world);
elcvec[nlocal+nghost] = 0.0;
ch[nlocal+nghost] = chpot;

// solve the linear system using CG solver

charge_reax(nlocal,nghost,ch,aval,acol_ind,arow_ptr,elcvec);

// calculate the charge equilibration energy

energy_charge_equilibration = 0;

// have already updated charge distributions for the current structure

for (ii = 0; ii < inum; ii++) {
    i = ilist[ii];
    itype = type[i];

    // 23.02 is the ReaxFF conversion from eV to kcal/mol
    // should really use constants.evfactor ~23.06
    // but that would break consistency with serial ReaxFF code
    // NOTE: this hard-wired real units
    // if want other units would have to change params[] in file

    qi = 23.02 * (param_list[itype].params[0]*ch[i]+
                 param_list[itype].params[1]*ch[i]*ch[i]);
    energy_charge_equilibration += qi;
    if (eflag_atom) eatom[i] += qi;
}

// copy charge vector back to particles from the calculated values

for (i = 0; i < nlocal+nghost; i++) q[i] = ch[i];
chpot = ch[nlocal+nghost];
}

/* ----- */

void PairREAX::charge_reax(const int & nlocal, const int & nghost,
                          double ch[], double aval[], int acol_ind[],
                          int arow_ptr[], double elcvec[])
{
    double chpottmp, suma;
    double sumtmp;

    cg_solve(nlocal,nghost,aval,acol_ind,arow_ptr,ch,elcvec);
}

/* -----
CG solver for linear systems
----- */

void PairREAX::cg_solve(const int & nlocal, const int & nghost,
                        double aval[], int acol_ind[], int arow_ptr[],
                        double x[], double b[])
{
    double one, zero, rho, rho_old, alpha, beta, gamma;
    int iter, maxiter;
    int n;
    double sumtmp;

```

```

// parallel CG method by A. P. Thompson
// distributed (partial) vectors: b, r, q, A
// accumulated (full) vectors: x, w, p
// r = b-A.x
// w = r      (ReverseComm + Comm)

double *r = rcg;
double *w = wcg;
double *p = pcg;
double *p_old = poldcg;
double *q = qcg;

n = nlocal+nghost+1;

one = 1.0;
zero = 0.0;
maxiter = 100;

for (int i = 0; i < n; i++) w[i] = 0;

// construct r = b-Ax

sparse_product(n, nlocal, nghost, aval, acol_ind, arow_ptr, x, r);

// not using BLAS library

for (int i=0; i<n; i++) {
    r[i] = b[i] - r[i];
    w[i] = r[i];
}

packflag = 1;
comm->reverse_comm_pair(this);
comm->comm_pair(this);

MPI_Allreduce(&w[n-1], &sumtmp, 1, MPI_DOUBLE, MPI_SUM, world);
w[n-1] = sumtmp;
rho_old = one;

for (iter = 1; iter < maxiter; iter++) {
    rho = 0.0;
    for (int i=0; i<nlocal; i++) rho += w[i]*w[i];

    MPI_Allreduce(&rho, &sumtmp, 1, MPI_DOUBLE, MPI_SUM, world);
    rho = sumtmp + w[n-1]*w[n-1];
    if (rho < precision) break;

    for (int i = 0; i < n; i++) p[i] = w[i];

    if (iter > 1) {
        beta = rho/rho_old;
        for (int i = 0; i < n; i++) p[i] += beta*p_old[i];
    }

    sparse_product(n, nlocal, nghost, aval, acol_ind, arow_ptr, p, q);

    gamma = 0.0;
    for (int i=0; i<n; i++) gamma += p[i]*q[i];
    MPI_Allreduce(&gamma, &sumtmp, 1, MPI_DOUBLE, MPI_SUM, world);

    gamma = sumtmp;
    alpha = rho/gamma;

    for (int i=0; i<n; i++) {
        x[i] += alpha*p[i];
        r[i] -= alpha*q[i];
        w[i] = r[i];
    }

    comm->reverse_comm_pair(this);

```

```

comm->comm_pair(this);

MPI_Allreduce(&w[n-1], &sumtmp, 1, MPI_DOUBLE, MPI_SUM, world);
w[n-1] = sumtmp;

for (int i=0; i<n; i++) p_old[i] = p[i];
rho_old = rho;
}
}

/* -----
sparse maxtrix operations
----- */

void PairREAX::sparse_product(const int &n, const int &nlocal,
                             const int &nghost,
                             double aval[], int acol_ind[], int arow_ptr[],
                             double *x, double *r)
{
    int i,j,jj;

    for (i=0; i<n; i++) r[i] = 0.0;

    for (i=0; i<nlocal; i++) {
        r[i] += aval[arow_ptr[i]]*x[i];
        for (j=arow_ptr[i+1]; j<arow_ptr[i+1]; j++) {
            jj = acol_ind[j];
            r[i] += aval[j]*x[jj];
            r[jj] += aval[j]*x[i];
        }
    }

    for (i=nlocal; i<nlocal+nghost; i++)
        for (j=arow_ptr[i]; j<arow_ptr[i+1]; j++) {
            jj = acol_ind[j];
            r[i] += aval[j]*x[jj];
            r[jj] += aval[j]*x[i];
        }
}

/* -----
memory usage of local atom-based arrays
----- */

double PairREAX::memory_usage()
{
    double bytes = nmax * sizeof(int);
    bytes += 7 * nmax * sizeof(double);
    bytes += matmax * sizeof(int);
    bytes += matmax * sizeof(double);
    return bytes;
}

/* -----
print out the itemized energies to the log file
this is not currently called, but should
be made available to the user in some way
----- */

void PairREAX::output_itemized_energy(double energy_charge_equilibration)
{
    double etmp[14], etmp2[14];

    etmp[0] = FORTRAN(cbkenergies, CBKENERGIES).eb;
    etmp[1] = FORTRAN(cbkenergies, CBKENERGIES).ea;
    etmp[2] = FORTRAN(cbkenergies, CBKENERGIES).elp;
    etmp[3] = FORTRAN(cbkenergies, CBKENERGIES).emol;
    etmp[4] = FORTRAN(cbkenergies, CBKENERGIES).ev;
    etmp[5] = FORTRAN(cbkenergies, CBKENERGIES).epen;
    etmp[6] = FORTRAN(cbkenergies, CBKENERGIES).ecoa;

```

```

etmp[7] = FORTRAN(cbkenergies, CBKENERGIES).ehb;
etmp[8] = FORTRAN(cbkenergies, CBKENERGIES).et;
etmp[9] = FORTRAN(cbkenergies, CBKENERGIES).eco;
etmp[10] = FORTRAN(cbkenergies, CBKENERGIES).ew;
etmp[11] = FORTRAN(cbkenergies, CBKENERGIES).ep;
etmp[12] = FORTRAN(cbkenergies, CBKENERGIES).efi;
etmp[13] = energy_charge_equilibration;

MPI_Allreduce(etmp,etmp2,14,MPI_DOUBLE,MPI_SUM,world);

```

```

if (comm->me == 0) {
  fprintf(screen,"eb = %g \n",etmp2[0]);
  fprintf(screen,"ea = %g \n",etmp2[1]);
  fprintf(screen,"elp = %g \n",etmp2[2]);
  fprintf(screen,"emol = %g \n",etmp2[3]);
  fprintf(screen,"ecoa = %g \n",etmp2[4]);
  fprintf(screen,"epen = %g \n",etmp2[5]);
  fprintf(screen,"ecoa = %g \n",etmp2[6]);
  fprintf(screen,"ehb = %g \n",etmp2[7]);
  fprintf(screen,"et = %g \n",etmp2[8]);
  fprintf(screen,"eco = %g \n",etmp2[9]);
  fprintf(screen,"ew = %g \n",etmp2[10]);
  fprintf(screen,"ep = %g \n",etmp2[11]);
  fprintf(screen,"efi = %g \n",etmp2[12]);
  fprintf(screen,"eqeq = %g \n",etmp2[13]);
  fprintf(logfile,"eb = %g \n",etmp2[0]);
  fprintf(logfile,"ea = %g \n",etmp2[1]);
  fprintf(logfile,"elp = %g \n",etmp2[2]);
  fprintf(logfile,"emol = %g \n",etmp2[3]);
  fprintf(logfile,"ecoa = %g \n",etmp2[4]);
  fprintf(logfile,"epen = %g \n",etmp2[5]);
  fprintf(logfile,"ecoa = %g \n",etmp2[6]);
  fprintf(logfile,"ehb = %g \n",etmp2[7]);
  fprintf(logfile,"et = %g \n",etmp2[8]);
  fprintf(logfile,"eco = %g \n",etmp2[9]);
  fprintf(logfile,"ew = %g \n",etmp2[10]);
  fprintf(logfile,"ep = %g \n",etmp2[11]);
  fprintf(logfile,"efi = %g \n",etmp2[12]);
  fprintf(logfile,"eqeq = %g \n",etmp2[13]);
}
}

```

## REFERENCES

- [1] S. Srinivasan, M. Baskes, G. Wagner, “*Atomistic simulations of shock induced microstructural evolution and spallation in single crystal Nickel*”, Journal of Applied Physics, 101, 43504 (2007)
- [2] Y. Wang, Z. Pan, Q. Wei, A. Du, “*Impact energy dependence of Al13 cluster deposition on Ni(0 0 1) surface*”, Surface Science, 512, 128 (2002)
- [3] S. Alavi, D. Thompson, “*Molecular Dynamics Simulations of the Melting of Aluminum Nanoparticles*”, Journal of Chemical Physics, 134, 2345 (2006)
- [4] V. Shutthanandan, A. Saleh, R. Smith., “*Alloy formation at the Ni Al interface for nickel films deposited on Al(110) surfaces*”, Surface Science. 450, 204 (2000)
- [5] Y.M. Wang, E.M. Bringa, J.M. McNaney, “*Deforming nanocrystalline nickel at ultrahigh strain rates*”, Applied Physics Letters, 88, 62917 (2006)
- [6] J. Xu, J.B. Luo, X.C. Lu, L.L. Wang, “*Atomic scale deformation in the solid surface induced by nanoparticle impacts*”, Nanotechnology, 16, 859 (2005)
- [7] H. Huang, J.R. Asay, “*Compressive strength measurements in aluminum for shock compression over the stress range of 4 to 22 Gpa*”, Journal of Applied Physics, 98, 33524 (2005)
- [8] M. Parrinello, “*Crystal structure and pair potentials: a molecular dynamics study*”, Physical Review Letters, 45, 1196 (1980)
- [9] M. Zacharia, “*Effect of coalescence energy release on the temporal shape evolution of nanoparticles*”, Physical Review B., 64, 205402 (2001)
- [10] N. Hoover, “*Constant temperature equations of motions*”, Physical Review A., 34, 2399 (1986)
- [11] B.L. Holian, “*Large-Scale Molecular Dynamics Simulations of Three-Dimensional Ductile Failure*”, Physical Review Letters, 78, 30907 (1997)
- [12] H. Hwang, “*Molecular dynamics simulations of energetic aluminum cluster deposition*”, Computational Material Science, 23, 105 (2002)
- [13] R. J. Miller., “*Atomic-scale simulations of nanoindentation-induced plasticity in copper crystals with nanometer-sized nickel coatings*”, Acta Materialia, 54, 33 (2006)
- [14] D. Wolf, V. Yamakov, S.R. Phillpot, A. Mukherjee , “*Molecular dynamics simulations of the preparation and deformation of nanocrystalline copper*”, Acta Materialia, 52, 5105 (2004)

- [15] W.A. Goddard III, A.C.T. van Duin, “*Melting and crystallization in Ni nanoclusters: The mesoscale regime*”, *Journal of Chemical Physics*, 115, 385 (2002)
- [16] V. Yang, “*Effect of Particle Size on Melting of Aluminum at Nano Scales*”, *Journal of Physical Chemistry C.*, 111, 11776 (2007)
- [17] U. Landman, “*Disordering and melting of Aluminum surfaces*”, *Physical Review Letters*, 61, 440 (1998)
- [18] E. Bringa et al., “*Ultrahigh Strength in Nanocrystalline Materials Under Shock Loading*”, *Science*, 309, 1838 (2005)
- [19] K. Kado, “*Microscopic View of Structural Phase Transitions Induced by Shock Waves*”, *Science*, 296, 1681 (2002)
- [20] B.L. Holian, “*Modeling of shock wave deformation via molecular dynamics*”, *Physical Review A.*, 37, 2562 (1988)
- [21] H. Zhou, “*Sintering processes of two nanoparticles: a study by molecular dynamics*”, *Philosophical Magazine Letters*, 73, 27 (1996)
- [22] Y. Wen et al., “*Size effects on the melting of nickel nanowires: a molecular dynamics study*”, *Physica E*, 25, 47 (2005)
- [23] F. Delogu, “*Structural and energetic properties of unsupported Cu nanoparticles from room temperature to the melting point: Molecular dynamics simulations*”, *Physical Review B.*, 72, 205418 (2005)
- [24] J. Diao et al., “*Surface-stress-induced phase transformation in metal nanowires*”, *Nature Materials*, 2, 656 (2003)
- [25] M. Parrinello, “*Unified approach for molecular dynamics and density functional theory*”, *Physical Review Letters*, 55, 2471 (1985)
- [26] G.I. Kanel, S.V. Razorenov, K. Baumung, “*Dynamic yield and tensile strength of aluminum single crystals at temperatures up to the melting point*”, *Journal of Applied Physics*, 90, 136 (2001)
- [27] V. Portnoy, “*Formation of nickel aluminides by mechanical alloying and thermodynamics of interaction*”, *Journal of Alloy and Compounds*, 336, 196 (2002)

- [28] D. Eakins et al., “*Shock-induced reaction in a flake nickel / spherical aluminum powder mixture*”, Journal of Applied Physics, 100, 113521 (2006)
- [29] R. Graham, “*High-pressure shock activation and mixing of nickel-aluminium powder mixtures*”, Journal of Material Science, 28, 2903 (1993)
- [30] J. Zimmerman, “*Calculation of stress in atomistic simulation*”, Modeling and Simulations in Material Science and Engineering, 12319 (2004)
- [31] D. Tsi, “*The virial theorem and stress calculation in MD*”, Journal of Chemical Physics, 70, 1375 (1979)
- [32] M. Tuckerman, “*Non-Hamiltonian molecular dynamics: Generalizing Hamiltonian phase space principles to non-Hamiltonian systems*”, Journal of Chemical Physics, 115, 1678 (2001)
- [33] S. Nose, “*A unified formulation of the constant temperature molecular dynamics*”, Journal of Chemical Physics 81, 511(1984)
- [34] A. Hoover, “*Constant pressure equations of motion*”, Physical Review A, 31, 1695 (1985)
- [35] M. Klein, “*Constant pressure molecular dynamics algorithms*”, Journal of Chemical Physics, 101, 4177 (1994)
- [36] H. Berendsen, “*MD with coupling to an external bath*”, Journal of Chemical Physics, 81, 3684 (1984)
- [37] M. Tuckerman, “*Understanding modern molecular dynamics*”, Journal of Physical Chemistry B., 104, 1678 (2000)
- [38] C.A. Schuh, A.C. Lund, “*Atomistic basis for the plastic yield criterion*”, Nature Materials, 2, 449 (2003)
- [39] A. Nakano et al., “*MD simulations of oxidation of an Al nanoparticle*”, Journal of Physical Chemistry B., 110, 3727 (2006)
- [40] M. Baskes, S. Foil, “*EAM functions for the fcc metals*”, Physical Review B, 33, 7983 (1986)
- [41] Y. Mishin, “*Embedded atom potential for B2 NiAl*”, Physical Review B, 65, 224114 (2002)
- [42] A. Duckham et al., “*Reactive nanostructured foils used as a heat source for joining process*”, Journal of Applied Physics, 96, 1521 (2004)



- [43] S. Jayaraman, O.M. Knio, A.B. Mann, T.P. Weihs, “*Numerical predictions of oscillatory combustion in reactive multilayers*”, *Journal of Applied Physics*, 86, 800 (1999)
- [44] J.C. Trenkle, T.P. Weihs, “*Microstructural study of an oscillatory formation reaction in nanostructured reactive multilayer foils*”, *Applied Physics Letters*, 87, 153108 (2005)
- [45] K. Morsi, “*Review-reaction synthesis of Ni and Al*”, *Material Science and Engineering A*, 229, 1 (2001)
- [46] K. Kelly, *Selected Values of Thermodynamic Properties of Metals and Alloys*, Wiley, 1963
- [47] A. Allen, D. Tildesley, *Computer Simulation of Liquids*, Oxford Press, 1986
- [48] D. Frenkel, F. Smit, *Understanding molecular simulations*, Academic Press, 2002
- [49] R. Hardy, “*Formulae for local properties in molecular dynamics: shock waves*”, *Journal of Chemical Physics* 76, 622(1982)
- [50] M.W. Finnis, J.E. Sinclair, “*A simple empirical N-body potential for transition metals*”, *Philosophical Magazine A*, 50, 1938(1984)
- [51] C. Kittel, *Introduction to solid state physics*, Wiley, 2004
- [52] C. Toh, “*Surface relaxation, stress and disordering in Pb*”, *Physical Review B*, 50, 17507 (1994)
- [53] S. Zhao, A.J. Strachan, “*Molecular dynamics simulation of dynamical response of perfect and porous Ni/Al nanolaminates under shock loading*”, *Physical Review B*, 76, 14103 (2007)
- [54] A.C.T. van Duin et al., “*ReaxFF: A Reactive Force Field for Hydrocarbons*”, *Journal of Physical Chemistry A*, 105, 9396 (2001)
- [55] A.K. Rappe, W.A. Goddard III, “*Charge equilibration for molecular dynamics simulations*”, *Journal of Physical Chemistry B*, 95, 3358 (1991)
- [56] K. Huang, *Introduction to Statistical Physics*, CRC Press, 1999
- [57] S. Thornton., *Classical Dynamics of Particles and Systems*, Brooks and Cole, 2002
- [58] K. Dill, *Molecular Driving Forces*, Garland Science, 2006
- [59] T.P. Orlando, P.L. Hegelstein, *Introduction to Applied Quantum and Statistical Physics*, Wiley, 2008
- [60] L. anand, *Mechanics of Solid Materials 2.071 Lecture Notes*, MIT, 2008, Unpublished
- [61] V. Yang, “*Effect of particle size on melting of Aluminum at nanoscales*”, *Journal of Physical Chemistry C*, 111, 11776 (2007)

[62] LAMMPS User Manual, Sandia National Laboratories, <http://lammps.sandia.gov>

[63] Atomeyes User Manual, J. Li, <http://mt.seas.upenn.edu/Archive/Graphics/A/>