Spring 2011

A modeling and simulation framework for electrokinetic nanoparticle treatment

James Phillips

A MODELING AND SIMULATION FRAMEWORK FOR ELECTROKINETIC NANOPARTICLE TREATMENT

by

James Phillips, B S , M S

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

COLLEGE OF ENGINEERING AND SCIENCE
LOUISIANA TECH UNIVERSITY

May 2011

LOUISIANA TECH UNIVERSITY

THE GRADUATE SCHOOL

04/15/2011

Date

We hereby recommend that the dissertation prepared under our supervision by James Ryan Phillips

entitled

A Modeling and Simulation Framework for Electrokinetic Nanoparticle Treatment

be accepted in partial fulfillment of the requirements for the Degree of

Doctor of Philosophy in Computational Analysis and Modeling

[illegible] 4/15/11

Supervisor of Dissertation Research

[illegible]

Head of Department

Computational Analysis and Modeling

Department

Recommendation concurred in

[illegible]

[illegible]

[illegible]

[illegible]

Advisory Committee

Approved

[illegible]

Director of Graduate Studies

[illegible]

Dean of the College

Approved

[illegible]

Dean of the Graduate School

GS Form 13a
(6/07)

ABSTRACT

The focus of this research is to model and provide a simulation framework for the packing of differently sized spheres within a hard boundary The novel contributions of this dissertation are the cylinders of influence (COI) method and sectoring method implementations The impetus for this research stems from modeling electrokinetic nanoparticle (EN) treatment, which packs concrete pores with differently sized nanoparticles We show an improved speed of the simulation compared to previously published results of EN treatment simulation while obtaining similar porosity reduction results We mainly focused on readily, commercially available particle sizes of 2 nm and 20 nm particles, but have the capability to model other sizes Our simulation has graphical capabilities and can provide additional data unobtainable from physical experimentation

The data collected has a median of 0 5750 and a mean of 0 5504 The standard error is 0 0054 at α = 0 05 for a 95% confidence interval of 0 5504 ± 0 0054 The simulation has produced maximum packing densities of 65% and minimum packing densities of 34% Simulation data are analyzed using linear regression via the R statistical language to obtain two equations one that describes porosity reduction based on all cylinder and particle characteristics, and another that focuses on describing porosity reduction based on cylinder diameter for 2 and 20 nm particles into pores of 100 nm height

Simulation results are similar to most physical results obtained from MIP and WLR Some MIP results do not fall within the simulation limits, however, this is expected as MIP has been documented to be an inaccurate measure of pore distribution and porosity of concrete Despite the disagreement between WLR and MIP, there is a trend that porosity reduction is higher two inches from the rebar as compared to the rebar-concrete interface The simulation also detects a higher porosity reduction further from the rebar This may be due to particles aggregating before reaching the rebar that can easily be seen in the graphical representation of the simulation cylinders

The dissertation author has created a web based framework to allow an interdisciplinary team to work in concert with access to the simulation and the results generated The results are stored into a MySQL database The database currently holds 271 simulation runs Simulation requests can be entered into a web interface and will automatically be processed in the order entered and the results stored into the database Results can also be retrieved from the database and filtered based on any simulation parameter Statistical analysis can be completed on the data points stored in the database by using version of Rweb modified by the dissertation author The result is a collaborative framework that can be extended to address future investigations into pore packing and chloride blocking

APPROVAL FOR SCHOLARLY DISSEMINATION

TABLE OF CONTENTS

ABSTRACT iii

LIST OF TABLES x

LIST OF FIGURES xi

ACKNOWLEDGEMENTS xiii

PREFACE xiv

CHAPTER 1 INTRODUCTION 1

1 1 Objective 1

1 2 Research Tools 2

1 3 Methodology 3

1 4 Main Results 5

1 5 Overview 6

CHAPTER 2 BACKGROUND 7

2 1 Issues 7

2 1 1 Concrete 8

2 1 2 Properties of Concrete 8

2 1 3 Deterioration 9

2 1 4 Importance of Packing Density 11

2 2 Deterioration Mitigation Strategies 11

2 2 1 Concrete Mixtures 11

2 2 2 Electrochemical Chloride Extraction (ECE) 12

2 2 3 Electrokinetic Nanoparticle (EN) Treatment 12

2 3 Measurement Techniques 14

2 3 1 Weight Loss Reduction (WLR) Method 14

2 3 2 Mercury Intrusion Porosimetry (MIP) 15

CHAPTER 3 MODELING AND SIMULATION METHODS 17

3 1 Modeling 17

3 2 Sphere Packing 18

3 2 1 History 18

3 2 2 Categorization 19

3 3 EN Model 20

3 3 1 Coordinate System 20

3 3 2 Sphere Model 21

3 3 3 Partial Sphere Volume 22

3 3 4 Cylinder Model 23

3 3 5 Solution Percent Weight to Volume Conversion 24

3 4 Simulation 27

3 4 1 How Computer Simulation Can Help 28

3 4 2 Asymptotic Complexity 28

3 4 3 Time 31

3 4 4 Cylinder Boundary Collision Detection 32

3 4 5 Sphere Collision Detection 34

3 4 6 Sphere Introduction 35

3 4 7 Sphere Movement 36

3 5 Methods 39

3 5 1 General Simulation Method 40

3 5 2 Fixed Spheres 42

3 5 3 Detecting Simulation End 46

3 6 Broadphase 47

3 6 1 Brute Force 48

3 6 2 Cylinder of Influence (COI) Method 50

3 6 3 Sectoring Method 55

CHAPTER 4 SIMULATION RESULTS AND ANALYSIS 65

4 1 Sectoring and COI Results 65

4 2 Linear Regression 70

4 3 Comparison to Other Methods 73

4 4 Relationship to Experimental Results 75

4 5 Time Analysis 77

CHAPTER 5 CONCLUSION AND FUTURE WORK 86

5 1 Conclusion 86

5 2 Parallelism 87

5 2 1 Coarse Grain Parallelization 89

5 2 2 Fine Grain Parallelization 90

5 3 Particle Mechanics 91

5 3 1 Sımultaneous Movement of Dıfferently Sızed Partıcles 91

5 3 2 Addıtıonal Physıcal and Chemıcal Rules 92

5 3 3 Chlorıde Introductıon and Analysıs 92

5 4 Improved User Interface 95

APPENDIX A Command-Lıne Optıons 97

APPENDIX B MySQL Database Result Storage 100

APPENDIX C Sımulatıon Web Interface 103

APPENDIX D Compılatıon Instructıons 112

REFERENCES 114

LIST OF TABLES

Table 3 1 List of movement vectors 37

Table 3.2 Psuedocode for particle movement 39

Table 3 3 General simulation method 41

Table 3 4 Conditional probabilities of choosing a successful vector on the kth attempt 43

Table 3 5 Brute force broadphase collision detection 49

Table 3 6 COI broadphase collision detection 55

Table 3 7 Sector G and its neighbors by sector code 58

Table 3 8 Brute force broadphase collision detection 64

Table 4 1 Comparison of analytical, brute force and sectoring results for 20 nm height and diameter cylinders 74

Table 4 2 Comparison of analytical, brute force and sectoring results for 20 nm height and 50 nm diameter cylinders 74

Table 4 3 Search space for brute force, COI and sectoring methods 85

Table 4 4 Comparison of analytical, brute force and sectoring results for 20 nm height and 50 nm diameter cylinders 85

LIST OF FIGURES

Figure 3 1 Model co-ordinate system 21

Figure 3 2 Representation of a pore as a cylinder 24

Figure 3 3 Side view of a cylinder for containment 33

Figure 3 4 Sphere interaction determination 34

Figure 3 5 Particle introduction range 36

Figure 3 6 Sphere movement directions 37

Figure 3 7 Sphere A oscillating between two locations 44

Figure 3 8 Brute force linked list representation 49

Figure 3 9 COI data structure and matching packed cylinder 50

Figure 3 10 SOI and COI boundary 52

Figure 3 11 Search space of various queues in COI 54

Figure 3 12 Sector length compared to sphere diameter 56

Figure 3 13 Sector grid in relation to a cylinder with empty sector padding 56

Figure 3 14 Sector addressing relationship to axes 57

Figure 3 15 Sphere interactions with neighboring sectors 60

Figure 3 16 Two-dimensional representation of improved sectoring method 61

Figure 3 17 Sector grid overlays 63

Figure 4 1 COI porosity reduction for 2 nm and 20 nm spheres at 0 05 movement 66

Figure 4 2	Sectoring porosity reduction for 2 and 20 nm particles at 0 1 nm movement	67
Figure 4 3	Counts of 20 nm particles packed with COI at various cylinder sizes	68
Figure 4 4	Counts of 20 nm particles packed using the sectoring method	68
Figure 4 5	Graphical representation of a packed pore obtained from the implementation of the sectoring method	70
Figure 4 6	Scatterplot of 100 nm height cylinders packed with 2 and 20 nm spheres	71
Figure 4 7	Boxplots of porosity reduction given cylinder diameter	72
Figure 4 8	Comparison of physical results to simulation results	76
Figure 4 9	Cylindrical cross-section porosity reduction	77
Figure 4 10	Sphere counts relationship to cylinder height and cylinder diameter	79
Figure 4 11	Plot of count of 20 nm spheres in relation to cylinder diameter	80
Figure 4 12	Plot of count of 10 nm spheres in relation to cylinder diameter	81
Figure 5 1	Corrosion grid	94
Figure C-1	Web interface status screen	106
Figure C-2	Web interface simulation request screen	107
Figure C-3	Web interface single and multiple simulation entry forms	108
Figure C-4	Web interface result request screen	109
Figure C-5	Web interface result request dropdown box	109
Figure C-6	Web interface where clause entry	110
Figure C-7	Web interface results output	111

ACKNOWLEDGEMENTS

I would lıke to express my gratıtude to the many people who supported and contrıbuted to the development of thıs dıssertatıon I am enormously grateful to Dr Jınko Kanno for takıng the tıme from her busy schedule to guıde me through thıs process I gratefully acknowledge Dr Galen Turner, Dr Henry E Cardenas, Dr Danıela Maınardı, and Dr Andreı Paun, whose counsel and research were ınvaluable ın the preparatıon of thıs report Nıck Rıchardson and Kunal Kupwade-Patıl have kındly contrıbuted theır knowledge, enthusıasm, and advıce to help me ımprove the thoroughness and qualıty of thıs dıssertatıon I am also grateful to my wıfe, Casey Sloan, who encouraged me and proofread my materıal

PREFACE

The purpose of this work is significantly different from the work I completed on the Watson Project for the Louisiana Tech University under Mike O'Neal I learned about many elements of working on a large program in concert with many collaborators while developing the JavaScript lab used in Watson Lab The lab is still in use in Louisiana Tech University computer science classes today and is freely available at http //watson latech edu/ [1] It has also been used in Health Informatics and Information Management classes

Through teaching Computer Science courses at Louisiana Tech University over a four year period, I have honed my skills at creating effective and efficient programs using meaningful techniques A focus on programming and data structures has served me well in completing this simulation of electrokinetic nanoparticle (EN) treatment

Computer scientists are often involved in projects that do not fall directly under the field of computer science This is certainly the case in this EN simulation research as it requires the need for a strong background in engineering, chemistry and mathematics It is unlikely for any one person to have all the expertise necessary to complete a simulation of this nature For example, it certainly would be unreasonable to ask a computer scientist to create the design for a bridge, in the same way as it would be to ask an engineer to develop commercial software without assistance It is our goal to create a framework that can easily be used and understood by collaborators without a deep understanding of the programming principles that make it possible

In today's world, it is important to collaborate with many experts in various fields to solve complex problems The impetus for this research was to create a program to demonstrate the electrokinetic nanoparticle (EN) treatment process in real time For the purpose of this paper all terms related to the EN process will be identical to those used in [2]

The results of the simulation of the EN treatment of concrete and corresponding model were published at the 2008 World Multi-National Conference on Systemics, Cybernetics and Informatics [3] The presentation was solely provided by the dissertation author and received the "Session's Best Paper Award" Simulation results were also presented informally to representatives from Entergy Nuclear, Mississippi, who provided financial support for the research from January 1, 2008 to December 31, 2008 The target application was to reinforce the walls of nuclear plants so they would not have to be replaced Improvements to the simulation work continued after these presentations and the results are presented in this dissertation

CHAPTER 1

INTRODUCTION

In this dissertation, we present a simulation and model using computational analysis techniques obtained from a background in computer science and mathematics In this research, we use our model to describe the electrokinetic nanoparticle (EN) treatment for pores in concrete and run EN treatment simulations several hundred times The simulation provides a low-cost alternative to performing EN treatments and provides a level of detail for individual pores that is unobtainable in physical experimentation It has also been suggested that the simulation should be utilized in modeling the effect of nanoparticle treatments on human bones and teeth to treat a variety of medical conditions [4] These health treatments are beyond the scope of this dissertation This model and simulation will be useful in any scenario where there are many mobile particles that must flow from one area to another while respecting hard boundaries

In this chapter, we briefly cover the objective, research tools, methodology, and main results We also present an overview of this dissertation at the end of this chapter

1.1 Objective

Our objective is to create a model and simulation that can be used to describe the electrokinetic nanoparticle (EN) treatment EN treatment is a technique that shows promise for increasing the longevity of concrete At the risk of gross oversimplification, EN treatment can be considered as the packing of spherical nanoparticles of various size into concrete pores

A functional requirement of our simulation was that all particles must flow to their packed locations as opposed to being immediately placed there It is desired that the simulation posses the capability to display the packing of pores in real time Furthermore, we require that the simulation run in a reasonable amount of time and memory, while maintaining accurate results We will show that our methods require significantly less time than previously published work on the EN treatment simulation presented at WMSCI while maintaining similar porosity reduction results

We will determine how spheres pack under cylindrical boundary conditions by collecting and analyzing simulation data We chose to represent the pore as a cylinder as it is the commonly assumed pore shape and also simplifies the problem [5] The research will be considered successful if the results of an electrokinetic nanoparticle (EN) treatment can be predicted An equation to predict the results based on input data such as the sphere sizes, ratios and cylinder heights and diameters is preferred We have obtained two such equations which are presented in Section 1 4 with more details in Chapter 4

1.2 Research Tools

Prototype versions of the simulation were coded in Sun Java and Microsoft Visual Studios C++ The current version is compiled with gcc in Linux and MingW in Windows in order to provide multi-platform support A MySQL database is used to store results and simulation requests An Apache web server serves a website with PHP server side code to access the MySQL database Linux BASH scripts are used to interface the simulation code with the MySQL database For statistical analysis, the R programming language was utilized [6] Rweb is utilized as a web interface to the R language Toodled was also embedded in the website to manage deadlines and goals for the research

1.3 Methodology

Our particular research is intended to simulate the electrokinetic nanoparticle (EN) treatment on concrete in detail for individual pores In our simulation, we model pores as cylinders and nanoparticles as spheres We focus on nanoparticles of diameters 2 and 20 nanometers (nm) Spheres will be randomly introduced into the cylinder at a constant rate Because concrete pores vary in height and diameter, we need to generate results for a range of cylinders with varying radii and heights

As part of this research, two algorithms have been developed to improve the speed at which the EN treatment can be simulated Cylinders of Influence (COI) and Sectoring Both of these algorithms share additional optimizations compared to work previously presented by the author at the World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI) in 2008 [3] Our algorithms simulate the flow of spheres from one end of a cylinder to the other while respecting the boundaries of the cylinder and all spheres The simulation programs utilized for this research require no manual interaction once they have been started Each simulation will automatically introduce spheres and will also detect when no more spheres can be packed into the cylinder Simulation results are stored in a database created and managed with the MySQL database management system (DBMS)

COI utilizes a custom data structure to reduce the time requirements of moving each particle during the simulation The custom data structure consisted of a collection of priority queues built on linked lists Each queue contained only one size of sphere, and iterators will keep tract of multiple points of interest in the structure This method takes advantage of the assumption that spheres do not move far in the linked list structure when they are moved This approach was developed under the assumption that cylinder diameter is relatively small compared to cylinder length As a result, this approach limits the impact of cylinder height increases on the time required to move an individual particle Changes to cylinder height generally have no effect on the time required to move an individual particle Larger simulation height does allow more particles to be packed which requires more simulation time, but this is unavoidable in our model The weakness of the COI method lies in diameter increases, and short cylinders with large diameter sizes quickly degenerate to the brute force method [7]

The sectoring method, formally known as a hierarchical grid method, was developed to simulate packing when cylinders were not limited by diameter [8] The sectoring method uses more memory, but reduces the number of calculations required to move a particle to a constant unaffected by cylinder height and diameter This method involves overlaying a three-dimensional grid over the cylinder We refer to each index-able area of the grid as a sector When a particle is moved it need only check nearby sectors, thus moving of individual particles can be done in a constant amount of time Sector sizes are automatically chosen by the simulation based off of the sizes of spheres to be packed

To validate the simulation, simulation runs were compared to live data and data from literature using statistical techniques Multivariate statistical techniques were used to analyze the data from numerous simulation runs over a wide variety of parameters [9]

Regressıon analyses were conducted to create equatıons that can be used to charactenze porosıty reductıon and predıct future results

1.4 Main Results

The maın results presented ın thıs dıssertatıon ınclude two equatıons obtaıned from lınear regressıons and dıscussed ın detaıl ın Sectıon 4 2 Equatıons 1 and 2 gıve porosıty reductıon (PR) of a packed cylınder gıven the charactenstıcs of the cylınder and spheres wıth whıch ıt ıs packed The cylınder has heıght C_H and dıameter C_D both gıven ın nanometers (nm) Partıcles have dıameter S_D ın nm As we have multıple sızes of partıcles, we sımply add a numencal subscnpt to denote dıfferently sızed partıcles In thıs case, the larger sıze of partıcle ıs S_1 and the smaller sıze ıs S_2 An analysıs was completed only on cylınders of heıght 100 packed wıth spheres of 2 and 20 nm to obtaın Equatıon 1

$$PR = 0\ 4757 - 0\ 00003\ C_D + 0\ 0207\ \ln(C_D) \tag{1}$$

Equatıon 1 ıs statıstıcally sıgnıficant, but only explaıns 59 percent of the vanance It does ındıcate that C_D has a better fit usıng a log transform Performıng a lınear regressıon analysıs ın R on all results obtaıned from the sımulatıon we obtaın Equatıon 2 where PR ıs the porosıty reductıon

$$PR = \ 4441 + 0\ 0081\ \ln(C_D) - 0\ 0003\ C_H + 0\ 0066\ S_{1D} - 0\ 0048\ S_{2D} \tag{2}$$

Equatıon 2 ıs statıstıcally sıgnıficant and explaıns 0 86 percent of the vanance For thıs paper, the coefficıents have been rounded to at most 5 decımal places, but the software ıs capable of 15 decımal poınt precısıon

1.5 Overview

Specific information on concrete and other background necessary to understand the issues involved are contained in Chapter 2 Chapter 3 contains a detailed explanation of the model and methodology used to simulate the EN treatment process This includes psuedocodes for the “brute force”, “cylinder of influence (COI)” and “sectoring” methods The fixing of spheres as described in Section 3 5 2 reduces the number of calculations required significantly The COI method is described in detail in Section 3 6 2 Specific details of the sectoring method can be found in Section 3 6 3 Chapter 4 contains the results and analyses of the simulation work including the two regression equations obtained Analyses were completed using the R statistical language Multiple linear regression was the primary statistical method utilized Chapter 5 summarizes the results and gives an overview of future research directions

CHAPTER 2

BACKGROUND

This chapter explains the background and motivation for this dissertation, including information on concrete and the electrokinetic nanoparticle (EN) treatment The main motivating factor for this research stems from the ingress of chloride ions into the concrete The chloride ions ultimately damage the concrete, which leads to the need for expensive repair or replacement In 2001, the "Annual direct cost of highway and bridge repairs (largely related to reinforcement corrosion) was $8 3 billion in the U S Alone" [10] Porosity reduction is the main deterrent used to combat the ingress of chloride ions To understand the mechanism and solution of concrete damage, we must discuss the nature of concrete, mechanisms of concrete deterioration and the common solutions to combat deterioration Given this information, we then describe the techniques of electrochemical chloride extraction (ECE) and electrokinetic nanoparticle (EN) treatment We will also discuss the methods used to evaluate the porosity of concrete weight loss ratio (WLR) and mercury intrusion porosimetry (MIP)

2.1 Issues

In order to understand the impetus and methods for this research we will give information on the nature of concrete, properties of concrete, deterioration of concrete, and the importance of packing density of concrete

2.1.1 Concrete

Concrete is composed of 12% cement The function of cement in concrete is to bind aggregates and sand together in a solid mass [11] A common type of cement is Portland cement Components of Portland cement include tricalcium silicate (C3S), dicalcium silicate (C2S), tricalcium aluminate (C3A), and calcium aluminoferrite (C4AF) [12] Modest variations in proportions of ingredients in cement can have a huge impact on the final product [11] The first step in the process of making cement is mixing the ingredients together and grinding them down into a fine powder referred to as clinker The size of the particles of cement clinker range from 5 to 50 µm, and the shape is spherical Cement paste is formed by mixing cement clinker with water The cement paste then hardens to form cement [12]

2.1.2 Properties of Concrete

The physical effects of chloride ingress and increased porosity of concrete are evident in the performance properties of concrete Properties of concrete which affect the performance of concrete include shrinkage, elasticity, strength and creep [13] Shrinkage refers to the decrease in volume of concrete during an extended time period which occurs due to changes in physio-chemical properties and moisture [14]

The shear resistance of concrete is affected by compressive strength, the ratio of shear span to depth and the ratio of embedded steel to concrete [15] Compressive strength refers to the amount of pressure concrete can withstand without breaking Greater levels of compressive strength in concrete positively correlate with tensile strength and negatively correlate with elasticity and creep [16]

Porosity is an intrinsic characteristic of cement mortar, concrete, and cement paste Pores allow for liquid and gas penetration Pores can be found in a variety of shapes with cylindrical pores being "the most commonly assumed shape" Performance properties of concrete are affected by pore structure Pore structure refers to pore size, pore connectivity, pore shape, total pore volume, pore distribution, surface area of pores, and number of pores [5] Increased porosity increases chloride permeability into concrete, which negatively impacts performance properties of concrete [13]

Types of pores include capillary pores, gel pores, air voids, and hollow-shell pores Gel pores tend to be the smallest pores, being only a few nanometers Gel pores are formed from C-S-H gel, which is the primary binder ingredient that gives microstructure and strength to cement paste Capillary pores are channels of interconnected pores, ranging from 2 nm to 10 μm in size Not all of the water added to the cement is used in the chemical reactions that occur during the hardening of concrete, The capillary pores form due to this "capillary water" Hollow-shell pores are the shape of a cement grain and range from 1 to 15 μm Hollow-shell pores form when a cement grain does not hydrate and the cement grain recedes, creating a hollow-shell pore Hollow-shell pores are difficult to identify using the mercury intrusion porosity, which is an often used technique for characterizing pore structures Air voids form due to air becoming trapped during formation of concrete Air voids may be several millimeters in size, but do not affect permeability of concrete structures [5]

2.1.3 Deterioration

Concrete structures may need to have a service life of a few years or a few centuries [17] Decreased durability and service life of concrete structures is often caused by steel corrosion Steel and other metals are used to reinforce concrete structures We generally refer to the iron reinforcement as rebar, which is short for "reinforcing bar" Steel corrosion leads to the deterioration of concrete structures [18] Costs of repairs and public safety represent the burden of corrosion of concrete [19] In 2001, the "Annual direct cost of highway and bridge repairs (largely related to reinforcement corrosion) was $8 3 billion in the U S Alone" [10] Financial cost of repairing damaged concrete structures may exceed the cost of primary construction [20]

Common causes of rebar corrosion include carbonation, chloride ingress, loss of alkalinity [18] Chloride ingress and sulfate attack are commonly thought to be serious threats to deterioration of concrete and decreased durability of concrete structures [21] The main determinant of durability of concrete structures is resistance to chloride ingress [22][23]

One of the most common causes of steel corrosion is chloride ingress Chloride ingress often occurs as a result of manufacturing materials being contaminated with chloride ions Manufacturing materials may become contaminated due to exposure to marine water or in cold climates when salt is placed on the concrete to melt ice [18] This is due to the chemical nature of salt, NaCl, and the disassociation of these atoms in water Prevention of chloride ions from moving through the cement and corroding steel is a major focus in improving the durability of concrete [19] Over time the porosity of concrete increases due the the mechanisms described above and can be measured indirectly through weight loss [3]

2.1.4 Importance of Packing Density

An important property of aggregates which affects performance of concrete is the particle size distribution Particle size distribution of aggregates affects the density at which particles can be packed [24] Increased particle density improves factors relating to performance of concrete, which include strength, creep, elasticity, and shrinkage Packing density can be improved by using admixtures However, the packing density of cement with different admixtures and various proportions of admixtures to cement is difficult to determine using current practices Current methods for determining particle size distribution do not predict the density at which particle aggregates pack [24]

2.2 Deterioration Mitigation Strategies

Decreasing permeability of concrete may decrease steel corrosion and improve durability of concrete Permeability of concrete is affected by type of cement, ratio of water to cement and chemical additives, such as silica fume, fly ash, and metakaolin, which are used to decrease pore size of cement [18] Decreasing the diameter of pore size decreases the diffusion of chloride ions into cement Additionally, connections between pores and total number of pores influence resistance of concrete to chloride ingress [25]

2.2.1 Concrete Mixtures

The proportion of chemical additives may be changed to enhance the effectiveness of additives in decreasing permeability [18]

In environments high in sulfates, admixtures are commonly recommended to decrease pore size, decrease proportion of C3A to silica, and decrease calcium hydroxide Additionally, in sulfate-rich environments, the durability of ordinary Portland cement mortar can be extended by decreasing the effects of sulfate attacks Researchers suggested that chloride ions may increase the amount of sulfate products that dissolve and increase the formation of Friedel's salt [21] Unfortunately the techniques described in reducing concrete pore sizes and porosity cannot be applied to already existing concrete structures, only to new structures

2.2.2 Electrochemical Chloride Extraction (ECE)

Electrochemical chloride extraction (ECE) removes chlorides from reinforced concrete using an electric current [26] A direct current is induced between the rebar inside the concrete and a titanium mesh placed on the surface of the concrete The rebar (cathode) is negatively charged and the mesh (anode) is positively charged, causing positively charged particles to move from mesh to rebar, and negatively charged particles to move from rebar to mesh [26]

As chloride ions are negatively charged particles, they are removed from the reinforced concrete Chloride extraction generally removes up to 40% of the chlorides Only the free ions in the pore solution are extracted, while bound chloride ions are still left in the concrete The ECE process is usually performed over a period of 6-12 weeks [26]

2.2.3 Electrokinetic Nanoparticle (EN) Treatment

The process of introducing nanoparticles into concrete with the intent of reducing porosity is referred to as EN treatment The process results in decreased permeability and increased strength [2][26][27][28] Work from Dr Cardenas is the influencing factor for the modeling of the EN treatment In 2006, 20 nm silica particles and 2 nm alumina particles were utilized in EN treatments reducing the permeability of cement by a factor of three [28] The movement rate of a charged particle in solution under a direct current is referred to as electrophoretic velocity Electrophoretic velocity is dependent upon electric current, particle size, zeta potential and possibly other parameters [28] Although, the simulation requires the electrophoretic velocity to accurately simulate their movement, it does not calculate the electrophoretic velocity but takes the value as an input The exact calculation of electrophoretic velocity of particles is beyond the scope of this dissertation and is not a feature of the simulation at this time The electrophoretic velocity of particles utilized in EN treatment were recorded as 2 8 E-8 m/s for 20 nm silica particles and 1 3E−7 m/s for 2 nm colloidal alumina under a 39 V/m electric field [27]

Electroosmosis is the term used to describe particle movement through a porous material and will be used interchangeably with electrophoretic velocity or particle velocity in the EN treatment simulation of individual pores

The EN simulation will be compared with porosity reduction results from actual EN treatments performed by Kunal Kupwade-Patil Experiments were performed on "cylindrical reinforced concrete specimens" under a variety of conditions Typically, experiments involved saltwater exposure between control specimens and specimens treated with ECE and EN ECE and EN treatments typically only were performed for seven days for each experiment, but the saltwater exposure lasted for as long as three years Specimens were also examined after shorted intervals of saltwater exposures, but unfortunately the process of examination resulted in the destruction of the specimen as required

There is a large investment of time and effort required to perform these experiments A simulation of the experiment may be more time efficient and cost effective, while providing more detailed information at the pore level

2.3 Measurement Techniques

The control and treatment specimens of EN treatment experiments need to be measured in order to determine the effects of EN treatments Specimens may be tested for many properties described in Section 2 1 2, but for this dissertation, we are only concerned with porosity and pore distribution Many of the methods used to measure these properties are destructive, therefore multiple samples are taken from individual specimens to be tested The sample is usually a cross-section of concrete and can be taken from various distances from the rebar and at various sizes Although the accuracy of the results of these techniques are not without criticism, there needs to be some attempt at comparison between actual results and simulation results The two techniques utilized are weight loss ratio (WLR) and mercury intrusion porosimetry (MIP)

2.3.1 Weight Loss Reduction (WLR) Method

The WLR porosity measurement method is described by [26] and this section comes completely from that work Samples are taken from an experimental specimen and powdered to weights of 5 – 7 g and ground to pass a No 30 sieve In this method, the weight of a concrete sample saturated with water is compared with the weight of the sample when dry The samples are dried at 105°C The initial weight of the sample is W_1 and the dried weight is represented by W_2 Porosity in terms of percentage can be calculated as shown in Equation 3

$$porosity\ percentage\, of\ sample = \frac{(W_1 - W_2)}{W_1}\ 100 \qquad (3)$$

Thus, the WLR method is a very simple straightforward technique for obtaining porosity percentage of concrete

2.3.2 Mercury Intrusion Porosimetry (MIP)

Mercury intrusion porosimetry (MIP) is the most commonly used technique for determining pore distribution and pore size in cement structures [5] Several steps are involved in the procedure of MIP First, a small piece of cement is dried, draining the pores of fluid The cement sample is placed in a chamber, and mercury is placed around the sample Pressure is applied to wet the cement with mercury The most common model is of cylinder-shaped pores, all of which are equally accessible to mercury intrusion Pressure is increased at intervals, and the movement of mercury into the cement is monitored during each interval The data for calculating pore size distribution is based on the amount of pressure applied and the volume of mercury introduced into the sample [29]

MIP is useful for providing information about pore size, but does not provide accurate data about pore size distributions Mercury is unable to access all pores due to shape of some pores and only a small percentage of pores are open directly to the outside of hydrated cement Therefore, pore size distribution can not be accurately estimated based on MIP [29] This method does not allow for differentiation between mercury intruding into one long pore or into many short pores It has been found that the model does not accurately demonstrate pore size distribution MIP indicates that pores are smaller than they actually are, due to mercury being unable to access pores that are very small or isolated Models have indicated that some pores are 10 to 100 times smaller than the actual size of the pores [29] Pore size distribution is typically determined by mercury porosimetry (MIP) Pore size distribution is difficult to determine based on MIP [30] MIP does not provide an accurate characterization of pore structure or total porosity [29][30][31]

All mercury intrusion porosimetry (MIP) data utilized in this paper were performed by Kunal Kupwade-Patil on powdered samples using a Micromeritics Autopore IV 9500 A maximum pressure of 226 MPa was applied in the MIP measurement Currently the Micromeritics Autopore IV 9500 is no longer functional

CHAPTER 3

MODELING AND SIMULATION METHODS

In this chapter we will cover the broad topics of modeling and simulation For modeling, we will discuss the history of sphere packing and the chemistry that is involved in our models We will also discuss previous methods and proofs related to this packing problem

3.1 Modeling

Modeling is used to describe a process or event in a formally written way When examining models of processes, it is important to understand that the model is an illustration of the process and not a determinant of it For example, if we correctly implement all the rules that we are aware of in regards to a given process, but our model results do not conform to the analytical results, we can infer that there is a component missing from our understanding The process being modeled may be simple or complex, but even simple models can provide insight into the examined processes Examples of processes in which modeling was used include the equation for gravity, and how the HIV virus causes AIDS in patients [32]

We model the electrokinetic nanoparticle (EN) treatment to study the reduction of porosity in concrete at a pore level Information gained from modeling may reduce time requirements, increase porosity reduction and improve cost efficiency for ultimately electrokinetic nanoparticle treatments [27]

3.2 Sphere Packing

Sphere packing, as the name suggests, is the packing of spheres into either bounded or unbounded spaces We will give a short history of the history of sphere packing as well as the categorization of sphere packing methods

3.2.1 History

Kepler's conjecture proposes that the maximum packing density of mono-sized hard spheres in three-dimensional space is given in Equation 4

$$\frac{\pi}{\sqrt{18}} \simeq 0\ 74048 \qquad (4)$$

The Kepler conjecture has been proven to be true if the spheres are arranged in a regular lattice [33] That is, a pattern is repeated exactly to pack a given space Thus the only possible counterexample would be a packing without a repeating pattern In the twentieth century, Claude Ambrose Rodgers was able to prove an upper bound of 78% Although a higher bound than Kepler's conjecture, we can be certain of this absolute bound In 2003, Hales published a computational proof of Keplers conjecture in the Annals of Mathematics A computational proof of the analysis indicates that the proof is correct The flyspeck project was started to formalize the proof completely [34]

As a result, any packing density for mono-sized spheres greater than 78% should be rejected outright and packing densities higher than 74 048% should be viewed with great suspicion Furthermore, we are primarily considering sphere packing with two sets of spheres Each sphere in a given set is the same size, but the sets do not contain spheres of the same size To clarify, we are packing with two different sizes of spheres In this case, we can be assured of an absolute bound of 78%+22%*78%=95 16% That is the packing density of a large sphere size of 78% and the packing density of the unpacked space, 22%, with a smaller sphere size Considering Kepler's Conjecture instead, we obtain an upper bound of 74 048%*25 952%*74 048%=93 265% This bound holds no matter how small the size of the smaller sphere set In our simulations, we expect to obtain much lower bounds for two reasons First, we are packing a bounded space, thus the spheres may not be able to fit in the available space due to topological reasons, even if the mathematical calculations indicate there is enough total space available Second, we will not be using a structured packing in a non-bounded space We have bounds that are not allowed to intersect the spheres, thus we do not count partial spheres as many previous methods Furthermore, we are randomly packing, so there is no effort made to consciously pack spheres as close as possible There are two types of packing found in molecular structures that approach the maximum mono-sphere packing bound hexagonal close packing (HCP) and face centered cubic (FCC) packing

3.2.2 Categorization

Random packing can be categorized into random loose packing (RLP) or random close packing (RCP) [35] Random loose packing traditionally implies that particles are packed with some constant force such as gravity The difference between the two methods is that random close packing involves shaking or some other temporary force applied to increase the density of a random loose packing Previously experiments in literature result in an average density of 55% for random loose packing and 64% for random close packing [35]

3.3 EN Model

Concrete pores are modeled as cylinders [5] To decrease time of simulation, nanoparticles are modeled as hard spheres [36] This simulation work would still be valuable for modeling non-spherical nanoparticles as the non-spherical nanoparticles could be bounded by a bounding sphere for broad phase collision detection purposes [37] The simulation algorithms and model still hold in the case of non-spherical nanoparticles, but the collision detection would need to be extended to compensate for the change

3.3.1 Coordinate System

We will describe the methods used to pack differently sized spheres into a cylindrical space We create our objects in three dimensional space with *x*, *y* and *z* representing each dimension The coordinates represent three dimensional space as follows Using this page as a reference, *x* would represent left and right, *y* would represent up and down, and *z* would represent moving towards or away from the page For the x-axis, the left direction is negative and the right direction is positive For the y-axis, up is considered negative and down is considered positive For the z-axis, moving closer to the page is considered a negative movement and moving away from the page is negative (Figure 3 1)

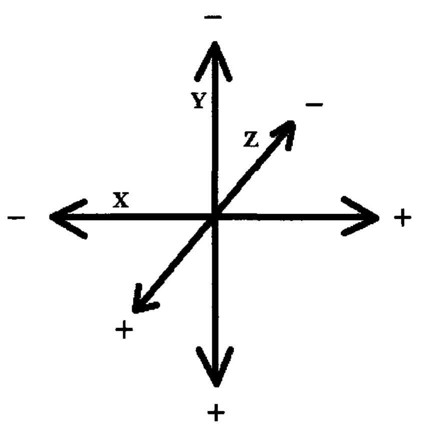


Figure 3 1 Model co-ordinate system

3.3.2 Sphere Model

In particular, we are interested in packing a bounded three-dimensional space with two sets of spheres of different sizes, say sphere set A with diameter A_D and sphere set B with diameter B_D A sphere from either set will be represented by S The diameter D of sphere S will be represented by S_D If $S_D = A_D$, then S belongs to set A If $S_D = B_D$, then S belongs to set B We will use R to represent radius when mathematically convenient, where the radius of sphere S is S_R

$$S_R = \frac{S_D}{2} \textit{ for any sphere } S \qquad (5)$$

We will refer to a sphere's location by the point at the center of the sphere Sc As Sc is a point in three-dimensional space, it has an x,y, and z coordinate which we will refer to as Sx, Sy and Sz respectively

3.3.3 Partial Sphere Volume

It is useful to be able to calculate partial sphere volume We only present partial sphere volume for a sphere that is intersected by at most two planes We utilize the disc method to sum cylindrical slices of the sphere between two xz-planes As all spheres are circular, they revolve around axes parallel to the y-axis We determine the boundary of the intersecting planes as the boundary between sectors when moving on the y axis This can also be applied to the COI method, as xz-planes give boundaries for the COI The top of the cylinder is also congruent to a xz-plane, and we can accurately calculate partial sphere volume of spheres that are not fully contained The equation for a sphere of radius *R* is given in Equation 6,

$$R^2 = x^2 + y^2 + z^2 \tag{6}$$

We take xz-slices of the sphere which have an area given in Equation 7 where *r* is the radius of each slice,

$$V = \pi\, r^2 \tag{7}$$

For each slice *r* changes depending on the location along the axis for the sphere as described in Equation 8,

$$r^2 = R^2 - y^2 \tag{8}$$

We then take the integral of the slices over the desired area of the sphere shown in Equation 9,

$$\int_a^b \pi\,(R^2 - y^2)\,dy \, where\; R \geq b > a \geq -R \tag{9}$$

Solving for the integral results in Equation 10,

$$V=\left(\pi\ R^2\ b-\frac{\pi}{3}\ b^3\right)-\left(\pi\ R^2\ a-\frac{\pi}{3}\ a^3\right) \tag{10}$$

Substituting $b = R$ and $a = -R$, we obtain the equation for the volume of a sphere as expected in Equation 11,

$$V=\left(\pi\ R^3-\frac{\pi}{3}\ R^3\right)+\left(\pi\ (R^3)-\frac{\pi}{3}\ R^3\right)=\frac{4\ \pi\ R^3}{3} \tag{11}$$

3.3.4 Cylinder Model

A cylinder C closed at one end is created with a diameter C_D and height C_H For simplicity, we may use cylinder radius C_R as desired where

$$C_R=\frac{C_D}{2} \tag{12}$$

We will refer to the closed end of the cylinder as the base of the cylinder The base of the cylinder represents the end of the pore where it may terminate with rebar The y-axis passes through the center of the base and is also perpendicular to the base The open end of the cylinder is centered at (0, 0, 0) We will refer to the open end of the cylinder as the top of the cylinder (Figure 3 2) Spheres are assumed to flow from above the xz-plane congruent with the top of the cylinder This area may either represent the area outside of the concrete or it may represent another pore that feeds into our own pore represented by C The remaining curved boundary of the cylinder will be referred to as the side of the cylinder

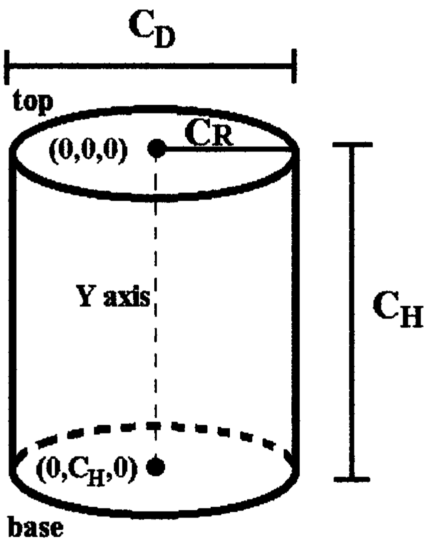


Figure 3 2 Representation of a pore as a cylinder

We add spheres to the cylinder from the top of the cylinder The large spheres of set A are introduced and packed first before the smaller spheres of set B are introduced In order to simulate the movement of spheres due to electrokinetic forces, we must check for collisions A sphere must not violate the the cylinder wall or boundaries of other spheres

3.3.5 Solution Percent Weight to Volume Conversion

In the literature and practice, references to the amount of particles added to solutions are given as a weight percentage All calculations utilized in the simulation require instead a volume percentage In order to accurately model a given solution, we must be able convert from a weight percentage to a volume percent The weight percentage is the percent of a solution's total weight taken by the solute For example, experiments conducted by Kupwade-Patil used a 12% 20 nm particle solution by weight [26] The simulation does not consider weight, but only volumes of particles In order to simulate pore packing similar to empirical data, we must convert weight percentage values to volume percentage values We will consider particle mass instead of weight, as weight is simply mass times gravity

$$weight = (mass)(gravity) \tag{13}$$

$$mass = (volume)(density) \tag{14}$$

$$volume = \frac{mass}{density} \tag{15}$$

We will follow the convention of the particles being referred to as solute, the liquid they are suspended in as solvent, and the combination of solute and solvent as solution

In order to calculate the percent weight of particles in solution divide the weight of the particles by the weight of the particles and solution combined Let W_p be the weight of a given set of particles Let W_s be the weight of the solvent in which the particles are suspended Let Pw represent the percent weight which is defined in Equation 16

$$P_w = \frac{W_p}{W_p + W_s} \tag{16}$$

Substituting Equation 13 into Equation 16 results in Equation 17 Let M_p represent the mass of the particles, M_s represent the mass of solution and G represent gravity G cancels and we obtain Equation 18

$$P_w = \frac{M_p G}{M_p G + M_s G} \tag{17}$$

$$P_w = \frac{M_p}{M_p + M_s} \tag{18}$$

Again, we substitute, this time with Equation 14 Let V_p represent the volume of the particles, D_p represent the density of the particles, V_s the volume of solution, and D_s the density of solution We obtain the following Equation 19,

$$P_w = \frac{V_p D_p}{V_p D_p + V_s D_s} \tag{19}$$

We note that the total volume of the solution is the combined weight of particles and solution Let V_t represent total volume of the solution with particles suspended as shown in Equation 20,

$$V_t = V_p + V_s \tag{20}$$

We wish to obtain the percent volume for the particles which we will represent with P_v as defined in Equation 21,

$$P_v = \frac{V_p}{V_t} \tag{21}$$

For convenience, we also need the percent volume of solute as defined in Equation 22,

$$P_{vs} = \frac{V_s}{V_t} \tag{22}$$

Note also that the percent volume of solvent must be $P_{vs} = 1-P_v$ Multiplying Equation 19 by V_t/V_t results in Equation 23,

$$P_w = \frac{P_v D_p}{P_v D_p + (1-P_v) D_s} \tag{23}$$

Solving for P_v results in Equation 24,

$$P_v = \frac{D_s}{\frac{D_p}{P_w} + D_s - D_p} \tag{24}$$

We can plug in values that correspond to experiments as appropriate A 12% by weight aluminum particle solution can be converted into percent volume given that the density of water is 0 9982071 g/cm^3 at 20° C and the density of aluminum is 2 698 g/cm^3 at 20° C Densities at other temperatures can be obtained from external sources, but 20° C is a standard value Substituting and solving, we obtain Equation 25,

$$P_v = \frac{99827}{\frac{2\ 698}{0\ 12} + 0\ 99827 - 2\ 698} = 0\ 0480\, or\, 4\ 8\,\% \tag{25}$$

So a 12% by weight aluminum solution is 4 8% by volume Generally since metals are more dense than water, they will have a lower volume percentage than weight percentage

3.4 Simulation

A simulation is the implementation of a model as a computer program [38] The complexity of the program generally correlates to the complexity of the model [32] A vague model will require assumptions to be made when implementing it as software It is also important to note that not all models can properly be implemented on current computer systems It is certainly impossible to hold an infinite amount of space, so all problems must be either practically finite in nature, or there must be a method of dealing with the infinities [39] This is the basic issue that prevents a computer from being categorized as a universal Turing machine [40] It is also important to note, that even when it is theoretically possible to implement a model as a simulation, the problems that are simulated can be intractable

3.4.1 How Computer Simulation Can Help

Computational simulations have been used to model functions of real objects [41] Since the 1980's, computers have been used in modeling 3-dimensional objects in construction design [42] This may decrease the time and cost of modeling and experimenting [41] Computer simulations help identify potential design flaws, allowing engineers to correct potential problems prior to initiating construction projects Computer simulations in construction decrease production cost and increase productivity Visualization of construction projects through computer modeling may provide information to engineers that would be difficult to determine through sole reliance on Equation [42]

3.4.2 Asymptotic Complexity

In this paper, we redesign the electrokinetic nanoparticle (EN) treatment simulation to reduce the time requirements at the expense of additional memory That is, our sectoring implementation runs significantly faster using a nominal increase in memory This will be described later in detail, but it will be necessary to understand asymptotic complexity as a basis for comparison For this paper, all references to asymptotic complexity will be in terms of time, unless specifically noted We will also adhere to the computer science standard of log_2N being written as $log\ N$

Asymptotic complexity is a formal method of describing the amount of resources a program requires in time or computer memory Both time and memory are considered independently and the complexity of each on a given program need not be identical We base our assessment on the number of objects that are entered into an algorithm which we represent as N Also, an arbitrary constant c serves to partition algorithms of similar growth rates into the same classes Typical growth rates include *c*, $log_2 N$, $(log_2N)^2$, *N*, *N log N*, N^2, N^3 and 2^N from least costly to most costly [43] Note that *log N* in this case is base two according to computer science standard notation, Thus log_2N is written as *log N* as a standard convention in computer science Other functions such as $N!$ and N^N can be used and the list is not exhaustive For demonstration purposes, let us consider two algorithms one that takes *1000 N* steps to complete and the other $10\,N^2$ steps If *N* is small then, $10\,N^2$ is smaller, but eventually $10\,N^2$ will become larger than *1000 N* as *N* becomes large [44] For a practical example, assume that we must sort 100 comparable objects, thus $N = 100$ Commonly used sorting algorithms,such as selection sort, are well known to take less than $c\,N^2$ steps which would take at most $c\ 100\ 100 = 10{,}000\ c$ steps to sort the 100 objects for some arbitrary constant *c* [45] A faster general sorting algorithm, mergesort, takes approximately *c N log N* steps, so to sort 100 object would take at most $c\ 100\ 7 = 700\ \ c$ steps for an arbitrary constant *c* As a result, you can see that the higher order equation takes many more steps than the lower order equation where the arbitrary constant *c* is not dissimilar The constant *c* becomes less important as *N* becomes large, and is generally ignored for comparison purposes

Let *T(N)* represent the number of steps an algorithm must take to solve a problem given *N* objects Let *f(N)*, *g(N)*, *h(N)* and *p(N)* arbitrarily represent a single term growth rate based on *N*, most likely from the typical growth rates discussed earlier

The formal definitions, according to [43] are

Definition $T(N) = O(f(n))$ if there are positive constants *c* and n_0 such that $T(N) \leq cf(N)$ when $N > n_0$

Definition $T(N) = \Omega(g(n))$ if there are positive constants c and n_0 such that $T(N) \leq cg(N)$ when $N > n_0$

Definition $T(N) = \Theta(h(n))$ if and only if $T(N) = O(h(N))$ and $T(N) = \Omega(h(n))$

Definition $T(N) = o(p(n))$ if $T(N) = O(p(N))$ and $T(N) \neq \Theta(p(n))$

The equal sign for asymptotic notation is prevalent in computer science, but mathematicians prefer to use the $\in$ symbol to acknowledge that $O(f(n))$, $\Omega(g(n))$, $\Theta(h(n))$ and $o(p(n))$ are sets $O(f(n))$, read "Big Oh", is the set of all functions that do not exceed the function $f(n)$ times an arbitrary constant This is considered to represent the worse case growth of an algorithm and is the most commonly used measure $\Omega(g(n))$, read "big omega", is the set of all functions that grow at least as fast as $g(n)$ times an arbitrary constant Big omega is often considered a best case scenario, describing the minimum growth of an algorithm $\Theta(h(n))$, read "big theta" is the set of all functions that are bound above and below by the same function $h(n)$ times an arbitrary constant that can be different for the upper and lower bound $o(p(n))$, read "little oh", is the set of all functions that are bound above by a function $p(n)$ times an arbitrary constant, but cannot be bound below by the same function times an arbitrary constant

Only the highest term should be included, $O(N^2+N)$ would not be written as such and instead be considered as $O(N^2)$ Constants are also not included in the notation, as they are accounted for in the definition Thus, we would not write $O(5N^2)$, but instead we write $O(N^2)$ Higher order functions can be quite costly in terms of time and memory If an algorithm to solve a problem takes more memory than is reasonably available or cannot solve a given problem in a reasonable amount of time, we refer to the problem as ıntractable Most algorıthms work wıth a small number of objects, but problems can quıckly become ıntractable ıf *N* becomes large

3.4.3 Time

When sımulatıng the electrokınetıc process, there are several tımescales There ıs the amount of tıme that a sımulatıon wıll take to complete We wıll refer to thıs as real tıme to be consıstent wıth the tıme command on Lınux In the sımulatıon, partıcles are beıng packed ınto a pore and tıme ıs progressıng We wıll refer to thıs tıme as sımulatıon tıme There ıs no sımple equatıon to convert between real tıme and sımulatıon tıme Thıs ıs due to the fact that the relatıonshıp between sımulatıon tıme and real tıme varıes based the number of partıcles beıng consıdered

In the sımulatıon, we use the term tıme-step to descrıbe each tıme we move every partıcle a set dıstance Each tıme-step can be converted dırectly to sımulatıon tıme, but the amount of real tıme for the tıme-step varıes dependıng on the number of partıcles to be moved To convert a tıme-step to sımulatıon tıme, we must know the partıcle speed and the partıcle dıstance moved each tıme-step Each tıme-step wıll take sımulatıon tıme as determıned by Equatıon 26

$$TS_{st}=\frac{P_M}{P_S} \tag{26}$$

The terms TS_{st} ıs used to represent the sımulatıon tıme for a tıme-step, P_S ıs the partıcle speed and P_M ıs the partıcle movement dıstance ın a tıme-step P_S should be ın nanometers per second and P_M should be ın nanometers Typıcally P_M wıll be a fractıon of P_S Solvıng Equatıon 23 for P_M, we obtaın Equatıon 24

$$P_M=TS_{st}\ P_s \tag{27}$$

When graphics are enabled, additional constraints are added In order to have smooth animation, 30 frames per second are desirable In this case, each time-step must complete in approximately 33 ms including the time required to draw the particles and pore In practice, if the time required for a time-step is less than 33 ms, the simulation remains idle for the unused time If the time required for each time-step is more than 33 ms, the simulation will visibly slowdown According to previous work, 20 nm particles were moved at 100 nm/s This is equivalent to 0 1 nm/ms

Substituting into Equation 27 results in Equation 28

$$P_M = 33\,ms\,\frac{0\,1\,nm}{ms} = 3\,3\,nm \tag{28}$$

Therefore at 30 frames per second, moving particles at 3 3 nm per time-step, we achieve equality between real time and simulation time Increasing the particle movement speed increases the time of each time-step according to Equation 26 which translates to a faster simulation relative to real time The issue with increasing particle movement rates too high is a lack of authenticity if the values become too large For example, a particle could potentially bypass blocking spheres to enter a gap that was impossible to reach

3.4.4 Cylinder Boundary Collision Detection

We first describe the conditions under which a sphere *S* is contained within a cylinder *C* For convenience, we define S_{yd} as the distance from sphere *S* to the y-axis as shown in Equation 29

$$S_{yd} = \sqrt{S_x^2 + S_z^2} \tag{29}$$

We note that a sphere contained within C may be partially contained or fully contained within C In our simulation, spheres that are not at least be partially contained within C as per Equation 30 will be deleted when detected

$$S_{yd} \leq C_R - S_R \text{ and } C_H + S_R \leq S_y \tag{30}$$

A partially contained sphere may extend beyond the opening of the pore at the xz-axis, as the plane represents the boundary between the pore and the area external to the pore On the other hand, a fully contained sphere adheres to Equation 31

$$S_{yd} \leq C_R - S_R \text{ and } C_H + S_R \leq S_y \leq S_R \tag{31}$$

Clearly, any sphere that is fully contained is at least partially contained, but the converse is not necessarily true Any sphere that is not partially contained within the cylinder will be deleted Figure 3 3 visualizes these boundaries in a simplified two-dimensional side view In Figure 3 3, the left illustration shows fully contained sphere A, partially contained sphere D, and non-contained spheres B, C and E On the right is an illustration of the variables C_R, S_{yd} and S_R for any vertical cross-section of a cylinder containing both the y-axis and a sphere to be checked for containment

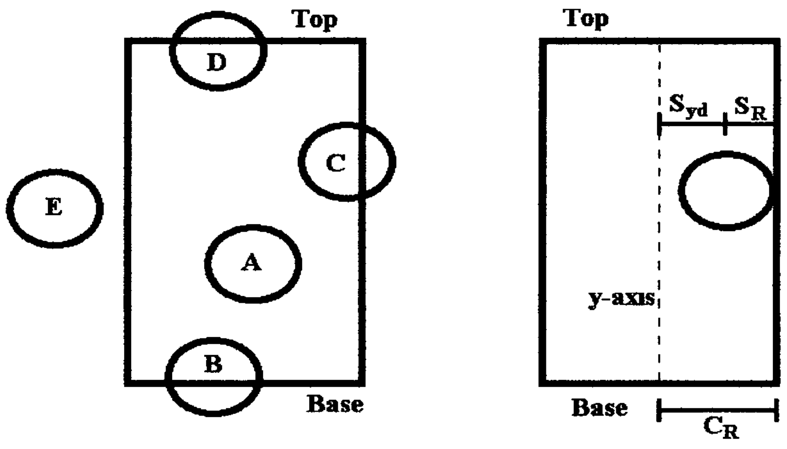


Figure 3 3 Side view of a cylinder for containment

3.4.5 Sphere Collision Detection

Collision detection between objects can be divided into two phases narrowphase and broadphase [46][47] Broadphase collision detection refers to the decision of determining which objects to test for collisions Narrowphase collision detection refers to testing for collisions between individual objects In this section, we will present only the narrowphase collision detection For a discussion of broadphase collision detection, please refer to Section 3 6

To check a collision between spheres we compare the sum of the sphere radii to the distance between their center points Assume a sphere S_1 with radius S_{1R} and another sphere S_2 with radius S_{2R} and sphere centers $S_{1c} = (S_{1x}, S_{1y}, S_{1z})$ and $S_{2c} = (S_{2x}, S_{2y}, S_{2z})$ Let $d(S_1, S_2)$ be the non-negative distance function between sphere centers S_{1c} and S_{2c} In other words

$$d(S_1 S_2)=\sqrt{(S_{1x}-S_{2x})^2+(S_{1y}-S_{2y})^2+(S_{1z}-S_{2z})^2} \tag{32}$$

If $d(S_1, S_2) \geq S_{1R} + S_{2R}$ then there is no violation of sphere boundaries with equality indicating that the spheres are touching Otherwise, if $d(S_1, S_2) < S_{1R} + S_{2R}$ then the spheres intersect and the boundaries are violated

In Figure 3 4, we illustrate the non-collision of spheres in A, the touching of spheres in B and the collision of spheres in C

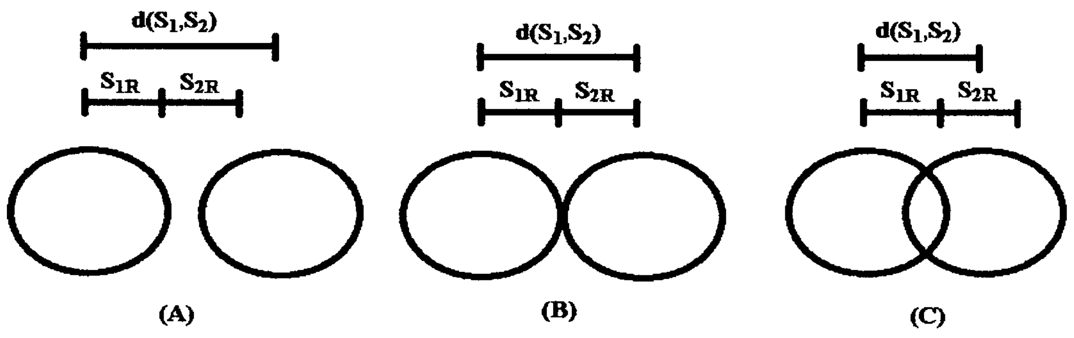


Figure 3 4 Sphere interaction determination

We ıllustrate the relatıonshıp between these collısıons states and the dıstance of spheres compared to the sum of theır radıı Spheres wıth dıfferent radıı are compared ın the exact same manner wıth no changes necessary In practıce, the sımulatıon avoıds a square root operatıon by squarıng both sıdes of Equatıon 32 Note also, that even though a sphere technıcally can collıde wıth ıtself accordıng to Equatıon 32, we wıll not consıder such an ıncıdent a collısıon and wıll avoıd testıng a sphere for collısıon agaınst ıtself

3.4.6 Sphere Introduction

Spheres are generated ın the xz-plane around the orıgın so that they are wıthın the bounds of the cylınder mouth The center of a sphere S that ıs ıntroduced wıll be on the xz-plane, or ın other words, $S_y = 0$ S_x and S_z are chosen such that S ıs partıally contaıned ın cylınder C For example, choose S_x such that an S_z exısts where the sphere ıs partıally contaıned ın C The lımıts for thıs are descrıbed to Equatıon 33

$$-C_R + S_R \leq S_x \leq C_R - S_R \qquad (33)$$

Gıven a random valıd S_x, we choose a random S_z such that S ıs partıally contaıned wıthın C as per Equatıon 34

$$-\sqrt{((C_R - S_R)^2 - S_x^2)} \leq S_z \leq \sqrt{((C_R - S_R)^2 - S_x^2)} \qquad (34)$$

Fıgure 3 5 ıllustrates the top of the cylınder wıth valıd S_x domaın marked wıth a solıd horızontal lıne Once S_x ıs chosen randomly, the valıd S_z range ıs a vertıcal lıne bounded by the dashed cırcle Note that ıf the S_x domaın ıs broken ınto n equal partıtıons, the areas of the dashed cırcle related to these partıtıons are not equal As a result, some partıtıons of the dashed cırcle have a hıgher probabılıty to contaın a sphere per unıt area than other partitions In order to mitigate this effect, we alternate S_x and S_z as the independent and dependent variables each time a new sphere is introduced

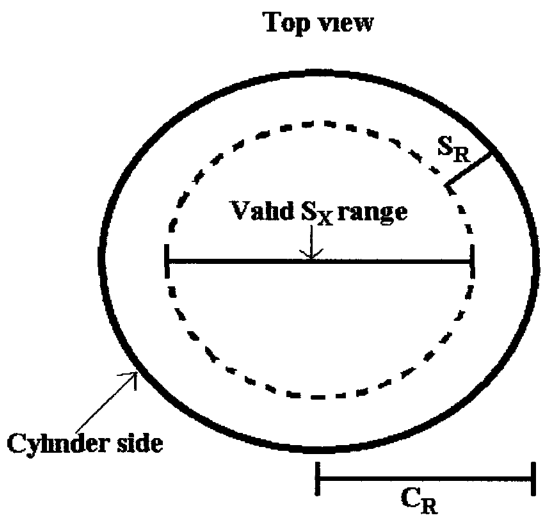


Figure 3 5 Particle introduction range

3.4.7 Sphere Movement

Two types of motion are simulated vertical movement and horizontal movement Downward movement is any motion with a vertical component Sideways movement is motion with no vertical component

In Figure 3 6, the top view of the sphere illustrates the eight directions that a particle can move horizontally The side view illustrates the difference between the vertical and horizontal directions For each horizontal direction, there is a related vertical direction that also contains a vertical component In addition, vertical motion also contains motion straight down, represented by the arrow pointing straight down There are nine vertical movement directions and eight horizontal movement direction available for a total of 17 possible movements The vectors to represent the 17 possible movements is referred to as M_T M_T contains disjoint subsets M_V and M_H where M_V is the set of vectors wıth a vertıcal component and M_H ıs the set of vectors wıth no vertıcal component Each vector M ıs composed of three real number values M_x, M_y and M_z The vectors to represent these movements are lısted ın Table 3 1

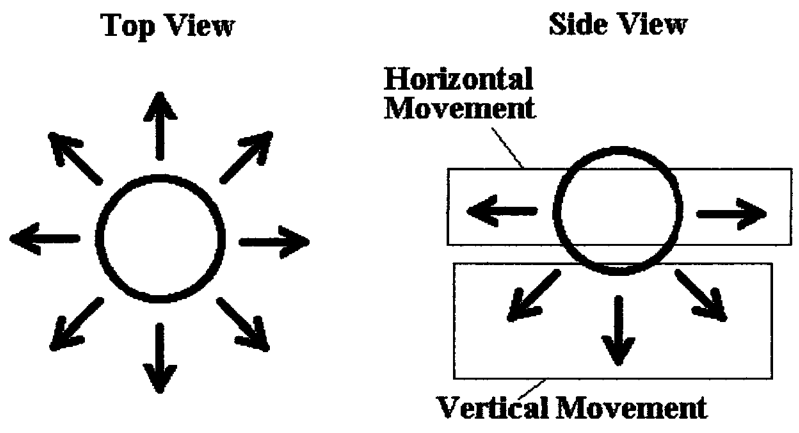


Fıgure 3 6 Sphere movement dırectıons

Table 3 1 Lıst of movement vectors

Vertıcal Motıon (M_x, M_y, M_z)	Horızontal Motıon (M_x, M_y, M_z)
(0 707106, 0 707106, 0 0)	(1 0, 0 0, 0 0)
(-0 707106, 0 707106, 0 0)	(-1 0, 0 0, 0 0)
(0 0, 0 707106, -0 707106)	(0 0, 0 0, -1 0)
(0 0, 0 707106, 0 707106)	(0 0, 0 0, 1 0)
(0 577350, 0 577350, 0 577350)	(0 707106, 0 0, 0 707106)
(0 577350, 0 577350, -0 577350)	(0 707106, 0 0, -0 707106)
(-0 577350, 0 577350, 0 577350)	(-0 707106, 0 0, 0 707106)
(-0 577350, 0 577350, -0 577350)	(-0 707106, 0 0, -0 707106)
(0 0, 1 0, 0 0)	No Movement

The magnıtude of each of the vectors of Table 3 1 ıs one Thıs ensures that partıcle movement wıll be consıstent regardless of dırectıon chosen In order to move a partıcle S, we create a new sphere S_N by performıng a translatıon on the center of S wıth a movement vector M scaled by the particle movement rate S_R [48][49] We must scale the vector by the particle movement rate S_R to reflect the electrophoretic velocity S_V input into the simulation The translation equation is shown in Equation 35 In order to process the vector multiplication and addition of Equation 35, we must evaluate S_x, S_y and S_z separately as shown in Equations 36 to 38

$$S_N = S_c + S_R\, M \tag{35}$$

$$S_{Nx} = S_x + S_R\, M_x \tag{36}$$

$$S_{Ny} = S_y + S_R\, M_y \tag{37}$$

$$S_{Nz} = S_z + S_R\, M_z \tag{38}$$

We create the new particle S_N instead of modifying the coordinates of S, in case the particle movement results in a collision If a collision occurs between S_N and any sphere other than S, or the sphere S_N is not partially contained, we immediately delete S_N and no movement has occurred Otherwise, we delete S and then relabel S_N as S This gives the appearance that S has moved only when such movement would not result in a collision Psuedocode can be found in Table 3 2

Table 3 2 Psuedocode for particle movement

```
A1 Pick an unused vector $V$ at random from $V_M$
A2 Generate a new sphere $S_N$ at the given position based on $S$ and $V$
A3 If $S_N$ has a collision, then the movement failed
    3 1 Delete $S_N$
    3 2 Remove vector $V$ from consideration in $V_M$
A4 Otherwise the movement succeeded
    4 1 Delete $S$
    4 2 Go to step A6
A5 If $V_M$ is empty, go to step A6
    5 1 Otherwise go to step A1
A6 Return movement success or failure
```

3.5 Methods

There are multiple algorithms that have been tested for the simulation of the electrokinetic nanoparticle treatment We will discuss several methods, but focus on the brute force method, the cylinder of influence (COI) method and sectoring method Generally, each simulation uses a cylinder to represent a pore of a certain size and spheres to represent nanoparticles being inserted to reduce the pore volume During the simulation, these particles are constrained to follow simple physical collision rules The particles must respect the boundary of the cylinder This means that if any part of the particle violates the cylinder wall it can not be used in calculating the porosity reduction Also each particle is not permitted to intersect with any other particle These collisions are tested continuously by using the position and radius of the particles and the radius and height of the cylinder

During the simulation, spheres are added in order of decreasing diameter Porosity reduction is calculated as the volume of added particles completely within the cylinder divided by the cylinder volume The simulation is capable of adding chloride ions as normal particles Our chloride ions would be modeled as solvated chloride ions that are represented as 0 6 nm spheres

3.5.1 General Simulation Method

We will describe the general algorithm for EN treatment simulation of a pore The simulation begins by initializing variables and data structures as per step A1 We will refer to the single completion of step A2 and step A3 as a time-step This is the smallest divisible unit of time that is distinguishable The movement rate of a sphere is the distance it will travel in a single time-step if the sphere moves The magnitude of each movement vector evaluates to one General psuedocode is written under Table 3 3

Table 3 3 General simulation method

```
A1 Initialize variables
    1 1 Diameter and height of the cylinder
    1 2 List of different sphere sizes
    1 3 Arrays of vectors of movement directions
    1 4 Method specific structures
A2 Introduce a constant number of spheres to the cylinder
    2 1 Check each inserted sphere A for collisions with other spheres
        2 1 1 If a collision occurs, delete A
A3 For each sphere A, attempt to move A
    3 1 Set $V_M=V_V$ and attempt vertical movements
    3 2 If all vertical movements of 3 1 failed
        3 2 1 Set $V_M=V_H$ and attempt horizontal movements
A4 Return to step 2 unless the simulation is complete
```

In each time-step, there is a particle introduction phase of step A2 and a movement phase of step A3 Particles are introduced in step A2 as per Section 3 4 6 Specific implementation details for the specific methods will be described in Sections 3 6 1, 3 6 2 and 3 6 3 The movement phase of step A3 involves the movement of particles described in Section 3 4 7 This phase requires multiple steps First, the electrokinetic force on the particles is simulated as a downward motion from the mouth of the cylinder to the base of the cylinder A finite displacement is defined for the distance that a particle was to move per unit of time, and per simulation cycle The simulation checks up to nine downward positions, one at a time, in the particle movement direction to see if the particle can move without violating the space of another particle or the cylindrical wall The order in which these directions are checked is random, and the first valid position is selected If no downward motion is possible, the eight sideways motions are checked in the same manner If all attempts to move the particle fail, then it remains in place This process may be visually displayed in 3-D using OpenGL For a platform independent GUI window, GLUT was originally used, but FreeGLUT is used in the latest versions At the time of this writing, the graphical version is only available for the Windows operating system

3.5.2 Fixed Spheres

During the course of the simulation, individual spheres will no longer be able to move for the rest of the simulation This is likely due to other spheres packing closely around and preventing movement The boundaries of the cylinder may also contribute to preventing sphere movement in conjunction with other spheres Attempting to move any sphere that is incapable of movement due to obstruction is futile, yet takes computational time In point of fact, attempting to move a sphere that is incapable of movement is more expensive than attempting to move a sphere capable of movement

A sphere incapable of movement must test all 17 movement directions in order to determine that a sphere cannot move On the other hand, a sphere capable of movement may succeed on the first movement attempt If we assume that we may have n successful movement vectors with equal probability, then we can calculate the probability of selecting a vector in the *kth* attempt that will not result in a collision Table 3 4 gives the conditional probabilities for each possibility as well as $E(k|n)$, the expected value of k given n The values were calculated in OpenOffice Calc The values are only shown to two decimal places and the zero values are simply < 0 01, but are non-zero values that OpenOffice Calc utilizes at an unknown higher precision Assuming that each non-zero value of n has an equal chance of occurring we average *E(k|n)* for all *n* to obtain *E(K)* = 2 64 Thus, immovable spheres are much more expensive where *E(0)* = 17

Table 3 4 Conditional probabilities of choosing a successful vector on the kth attempt

	k																	
n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	E(k|n)
1	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	0 06	9
2	0 12	0 11	0 1	0 1	0 09	0 08	0 07	0 07	0 06	0 05	0 04	0 04	0 03	0 02	0 01	0 01		6
3	0 18	0 15	0 13	0 11	0 1	0 08	0 07	0 05	0 04	0 03	0 02	0 01	0 01	0	0			4 5
4	0 24	0 19	0 15	0 12	0 09	0 07	0 05	0 04	0 02	0 01	0 01	0	0	0				3 6
5	0 29	0 22	0 16	0 12	0 08	0 05	0 03	0 02	0 01	0 01	0	0	0					3
6	0 35	0 24	0 16	0 1	0 06	0 04	0 02	0 01	0	0	0	0						2 57
7	0 41	0 26	0 15	0 09	0 05	0 02	0 01	0	0	0	0							2 25
8	0 47	0 26	0 14	0 07	0 03	0 01	0	0	0	0								2
9	0 53	0 26	0 12	0 05	0 02	0 01	0	0	0									1 8
10	0 59	0 26	0 1	0 04	0 01	0	0	0										1 64
11	0 65	0 24	0 08	0 02	0 01	0	0											1 5
12	0 71	0 22	0 06	0 01	0	0												1 38
13	0 76	0 19	0 04	0 01	0													1 29
14	0 82	0 15	0 02	0														1 2
15	0 88	0 11	0 01															1 13
16	0 94	0 06																1 06
17	1																	1
																		Average 2 64

Also, we must consider downward movement differently than horizontal movement A sphere will only be able to have at most a constant number of downward movements A sphere will certainly eventually be prevented from downward movement by the base of the cylinder if not earlier by other spheres It is important to realize that a sphere may not be completely immobilized and still oscillate horizontally, but nevertheless never be able to make another downward movement due to neighboring spheres or the cylinder base If we allow such spheres to oscillate indefinitely, they will take processor time, but may be irrelevant for the packing of other particles Figure 3 7 illustrates a potential situation where sphere A oscillates between two positions denoted by the solid and dashed sphere Surrounding sphere A are spheres that can no longer move represented by the shaded circles In this scenario, sphere A will always have have a movement choice, thus sphere A will always move despite the fact that the movement can have no effect on the surrounding spheres or the overall porosity

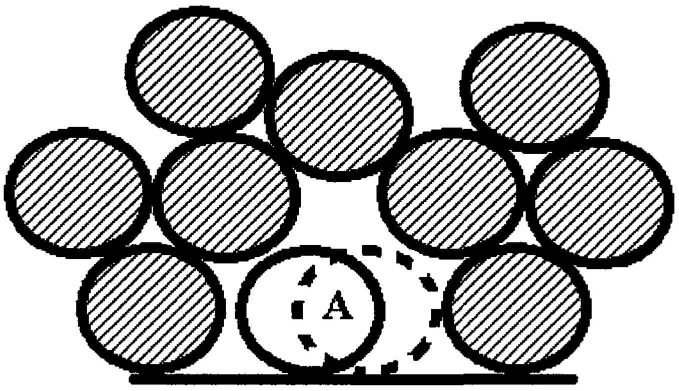


Figure 3 7 Sphere A oscillating between two locations

To reduce the number of calculations that are performed, we eliminate spheres from consideration for movement if it is determined that they will be unlikely to have any future vertical movements We refer to this as fixing the sphere, and after being fixed, the sphere is a fixed sphere The decision to fix a sphere is based on a positive integer stored for each sphere referred to as the fix counter The fix counter is initialized at zero for each sphere and is reset to zero for any sphere that successfully completes a vertical movement If the fix counter is greater than or equal to a critical fix value, then the sphere will become fixed The critical fix value can be adjusted in the program code, but the author has chosen it to be 1000 The fix counter may be incremented in three distinct scenarios 1) attempting to move an immobile sphere will increment the sphere's fix counter by a constant amount we will refer to as the fix score for immobile spheres or *FSI* , 2) Attempting to move a sphere that cannot move vertically, but can move horizontally will increase the fix counter by a different constant amount we will refer to as the fix score for horizontal only spheres or *FSHO* or 3) The sphere contacts the base of the cylinder, which we refer to as "bottoming out", and the fix counter will be increased by the fix score for bottoming out or *FSBO* *FSI* , *FSHO* and *FSBO* do not have a predetermined values and can be adjusted in the program code There is no option in the current simulation to fix a sphere due to contact with the side of the cylinder The author has chosen *FSI* = *10*, *FSHO* = *1* and *FSBO* = *10* for current simulation values An immobile sphere or sphere that contacts the cylinder base will be tested for movement 100 time-steps before being fixed A sphere can fail to move vertically, but successfully move horizontally 1000 time-steps before it becomes fixed If instead, it is desired that a sphere "stick" to the base of the cylinder upon contact, we would simply set *FSBO* to the critical fix value Higher values for *FSI* and *FSHO* will cause spheres to fix faster and may lead to lower overall porosity values for the simulation, whereas lower values may lead to higher porosity values, but requires additional computational time

The reduction of calculations needed for the simulation is at least the number of fixed spheres divided by the total number of spheres So if 50% of the spheres are fixed, we would have at least a 50% reduction in calculations The reduction may be more as the spheres which are candidates for being fixed have fewer movement options, so are on average more expensive to move as shown in Table 3 4 This may come at a cost of authenticity, as not all particles may actually stop, but their continued calculation is deemed irrelevant to the simulation results and computationally expensive

Furthermore, the sımulatıon can utılıze the fixed-ness of spheres to determıne when further sphere addıtıons and movement has no or a lımıted effect on the results From a practıcal poınt of vıew, we must consıder that spheres wıll generally remaın ın theır localıty so that the cylınder wıll be packed and the sımulatıon can be determıned as complete

3.5.3 Detecting Simulation End

The orıgınal sımulatıon presented at WMSCI requıred user ınteractıon to determıne when the cylınder was packed This ıs generally suffıcıent when only a few sımulatıon are run that each take only a short amount of tıme There are many optıons for detectıng when a cylınder ıs packed, and the sımulatıon should end

The sımulatıon end may be determıned by (ın no partıcular order) the followıng

1) User ınteracts to ındıcate sımulatıon termınatıon
2) Porosıty reductıon value converges to a value Specıfically, the change ın porosıty reductıon over a constant number of tıme-steps does not ıncrease more than a gıven constant
3) Sımulatıon ıs set to run a gıven number of tıme-steps, and sımply stops when they have been executed Thıs ıs equıvalent to sımulatıng the EN process for a set amount of tıme
4) No new spheres have been ıntroduced ın a fixed number of tıme-steps
5) No spheres currently ın the cylınder are capable of movement

The current sımulatıon determınes when the sımulatıon ıs complete gıven a combınatıon of condıtıons 4 and 5 Thıs ıs utılızed as the computatıon requıred ıs less than that of condıtıon 2 Fırst, we test condıtıon 5 whıch can be tested when attemptıng to move spheres in a time-step at limited additional cost At this point, as per condition 4, an attempt to add new spheres should be tested before ending the simulation If no spheres are able to be added, then the simulation is deemed complete and displays the results Condition 4 also takes limited overhead as introducing spheres to the cylinder is already necessary for each time-step On the other hand, condition requires that the porosity of the cylinder to be recalculated each step, which is not calculated in this version of the simulation

3.6 Broadphase

The broadphase collision detection of the simulation is a component of collision detection that determines which particles should be considered for narrowphase collision described in Sections 3 4 4 and 3 4 5 We have implemented two algorithms, COI and sectoring, that have reduced the asymptotic complexity of each time-step by reducing the number of comparisons necessary in the broadphase Each algorithm will utilize a subset of spheres that contain all the spheres that can potentially collide during a sphere's movement There are various names given to this subset depending on the particular strategy used to generate it, we will cover this in more detail for each subsection In general, we will call this set the broadphase collision set (BCS) The COI and sectoring methods have improved performance over the brute force method due to generating a smaller BCS of spheres for comparison thus reducing the number of sphere comparisons needed in order to move a sphere This set may still contain spheres that can not cause a collision under any condition, but the goal should be to include the fewest number of spheres while containing all spheres that cause a collision in a time efficient manner The broadphase for collision detection is utilized for both sphere introduction of Section 3 4 6 and sphere movement of Section 3 4 7

3.6.1 Brute Force

The term *brute force* is applied to any method in which an exhaustive number of operations are performed. It implies that there is a more efficient manner of solving the same problem without performing every operation available For example, when one searches for a word in the dictionary, if they viewed every single word starting at the first page until they found what they were looking for, they would be employing a brute force search The advantages of brute force methods lie in simplicity, ease of implementation, and the guarantee that a solution will be discovered if it exists (since all possibilities are considered) Unfortunately, the brute force method is often inefficient in terms of time

The original WMSCI paper presented results obtained from a brute force method We compare these results to sectoring later in Section 4 3 The brute force method stores spheres in a linked list data structure All spheres are contained within the single structure, therefore, each sphere must allocate memory to store its own size The order of the spheres in the list are completely dependent on the order in which they are added to the list The size and position of each sphere is irrelevant Figure 3 8 illustrates the structure

The third sphere shown in Figure 3 8 is in the head node of the linked list, and the first sphere is in the tail node Spheres will be evaluated from the head of the list to the tail Therefore, spheres are evaluated in the opposite order from which they were introduced For the brute force method, the broadphase collision set (BCS) is the entire linked list Psuedocode for the broad phase collision detection of the brute force method is displayed in Table 3 5

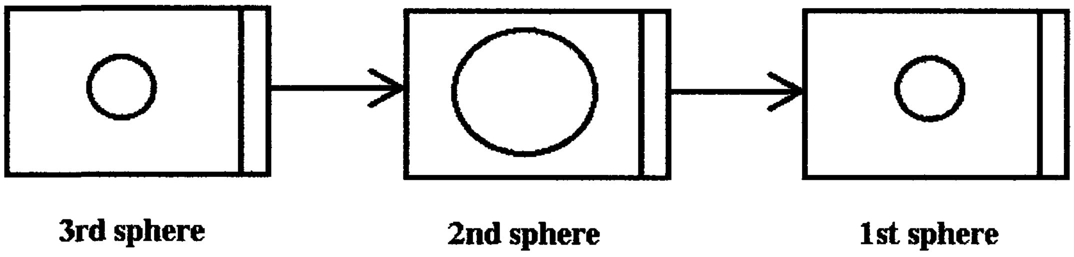


Figure 3 8 Brute force linked list representation

Table 3 5 Brute force broadphase collision detection

```
A1  Given a sphere $A_N$
A2  For each sphere B in the cylinder C, test for a collision between $A_N$ and B
    2 1  If a collision occurs according to Section 3 4 5, then return true
A3  Return false
```

For any sphere A_N that is generated either through particle introduction or movement of a particle A, we must test for collisions with every other sphere In the worst case scenario, where A does not collide with any other sphere, we must make *N-1* comparisons for a complexity of *O(N)* Even if sphere A will collide with other spheres, it is likely that the number of spheres that would cause a collision are quite small compared to the total number of spheres Further analysis will be carried out in Chapter 4 An advantage of this method is that it is trivial to add spheres of various sizes and to move all spheres regardless of size at the same time

3.6.2 Cylinder of Influence (COI) Method

The COI method is categorizes as a sort and sweep method [50] It is based on a collection of sorted linked lists of spheres as shown in Figure 3 9 There is one linked list for each sphere size We refer to each linked list as a sphereQueue Q Each of these linked lists only hold spheres of a given size and each sphere of that size is stored in its corresponding sphere queue In the figure, spheres of sizes 5, 15 and 20 nm are stored We can denote a particular sphere queue Q_D with a subscript to indicate the diameter of sphere it contains For example, Figure 3 9 contains Q_5, Q_{15} and Q_{25}

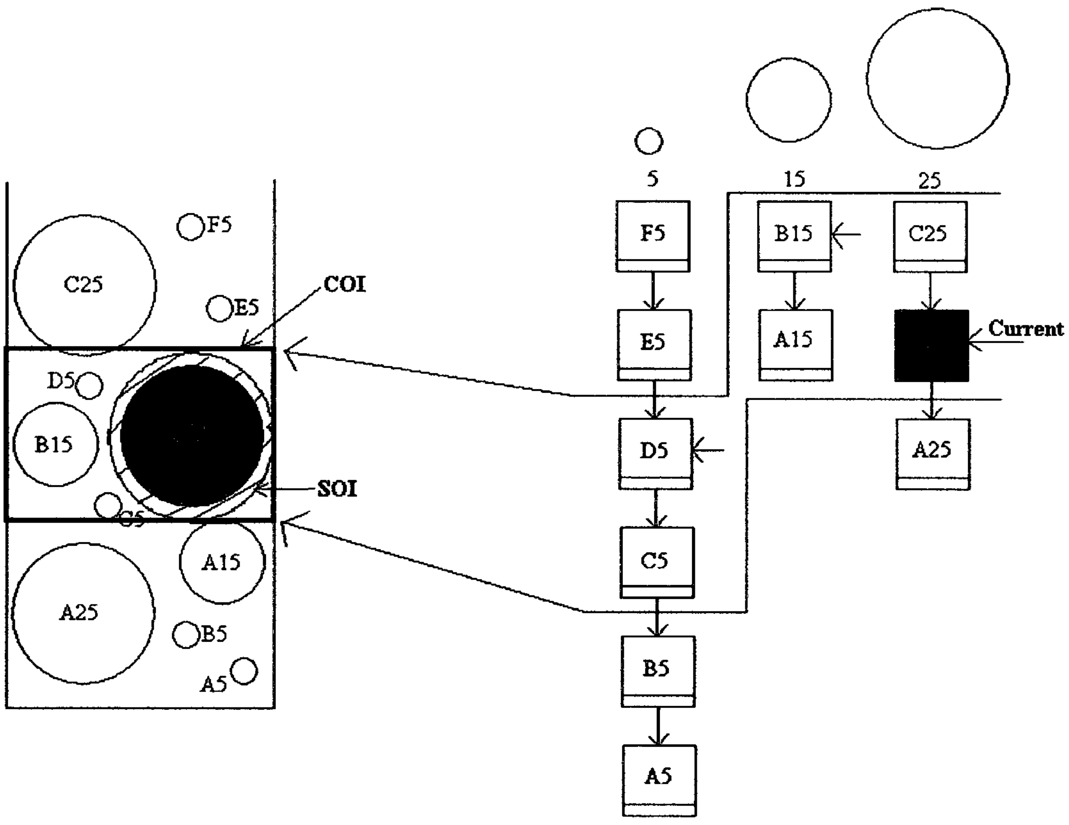


Figure 3 9 COI data structure and matching packed cylinder

As a side effect of this data structure change, the memory required to store each sphere is reduced Assuming that the sphere size is stored as a double to allow for the possibility of chloride ions at approximately 0 66 nm, we save eight bytes per sphere compared to the brute force method with no negative effects If there are a million spheres in a given simulation then we save 8 million bytes of RAM when running the simulation or approximately 7 6 MB

Each sphere queue is sorted by *y* value Recall from Section 3 3 4 that the top of the cylinder is set at (0, 0, 0) with a *y* value of zero, and *y* value increases moving towards the base of the cylinder The order of spheres in the sphere queues on the right side of Figure 3 9 correspond to the spheres depicted in the cylinder in the same figure on the left hand side For sphere A_5 and sphere B_5, sphere A_5 appears closer to the tail of the linked list and closer to the base of the cylinder because $A_{5y} > B_{5y}$

When moving a sphere or inserting a sphere, we generate the broadphase collision set (BCS) first In the COI method, there are two steps to this process We have two subsets that are generated the Cylinder of Influence (COI) and the Sphere of Influence (SOI) In Figure 3 9, the COI is represented by the darkened rectangle over the cylinder and represents a cylindrical cross-section, the SOI is represented by the shaded circle surrounding B25 and represents a spherical area We will use these terms to refer to both the subset of spheres with which they intersect as well as the area they represent

A sphere needs only check collisions against spheres that intersect with its SOI We can easily test a sphere for intersection with the SOI as explained in Section 3 4 5 given that both are spheres The radius of the SOI of A is determined by the sphere's range of movement denoted by the arrows as shown in Figure 3 10 The SOI is denoted by the light dashed circular line while the COI is denoted by the bold dashed horizontal line As the sphere is not allowed to move in the negative y direction, in practice, we reduce the top of the COI to reflect this fact and take the intersection of the SOI and the COI as the BCS

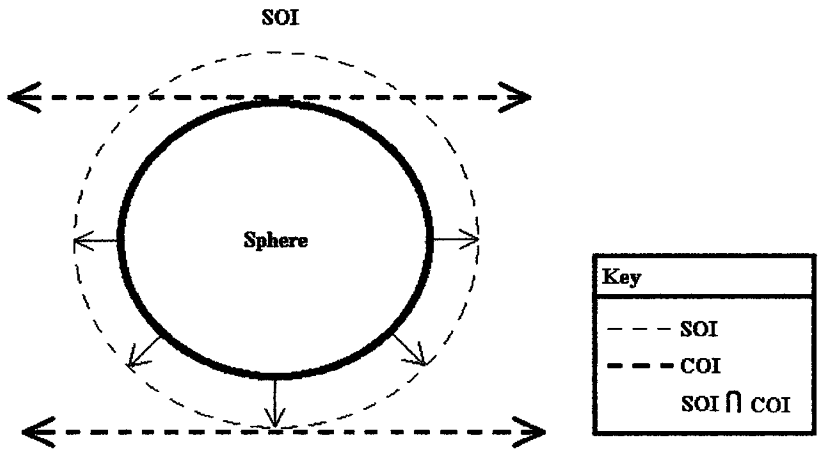


Figure 3 10 SOI and COI boundary

To generate the SOI is an *O(N)* operation by the obvious brute force method of checking each sphere for an intersection with the SOI Instead, we can reduce the number of comparisons by only considering spheres that intersect with the COI The height of COI is exactly the diameter of the sphere moving plus the distance that can be moved with the top of the COI congruent with the top of the related sphere For example, spheres of diameter 20 nm with a movement of 0 1 nm movement require a COI of height 20 1 nm and 2 nm spheres at the same movement rate require a COI of height 2 1 nm Spheres that are within their radius of the COI are considered to be within the COI A sphere S is considered to intersect with the COI of sphere A if it satisfies either Equation 39 or 40

$$S_y \leq A_y \text{ and } S_y + S_R > (A_y - A_R) \tag{39}$$

$$S_y > A_y \text{ and } S_y - S_R < (A_y + A_R + A_M) \tag{40}$$

Recall that spheres are sorted by y values ın sphere queues segregated by sphere dıameter Durıng a tıme-step, we evaluate spheres ın descendıng order of theır y values, that ıs, from the front of the queue to the rear of the queue Spheres can only move towards the front of each queue, thus no sphere wıll be evaluated more than once per tıme-step In order to keep track of the next sphere ın each sphere queue, we utılıze a poınter to the next sphere to be processed ın each queue We wıll refer to thıs as the current poınter as ıllustrated ın Fıgure 3 9 The next sphere to be evaluated wıll always be the sphere wıth the largest y value chosen only from current poınters In our ımplementatıon, we only allow one sphere sıze to be movıng at any gıven poınt ın tıme, but the comparısons of sphere y values ıs stıll processed to maıntaın accurate current poınters When a sphere ıs processed, the current poınter ıs moved towards the rear of ıts lıst

The advantage of maıntaınıng current poınters ıs that ıs ımproves the speed of determınıng whıch spheres ıntersect wıth the COI Consıder that we are processıng a sphere A ın Q We must consıder other spheres ın Q for ıntersectıon wıth the COI A_y ıs roughly at the center of the COI of A, thus there may be spheres above and below A ın Q that also fall wıthın the COI of A We perform two lınear searches of Q one from A towards the front of the queue (base of the cylınder) usıng Equatıon 40 and one from A towards the rear of the queue (top of the cylınder) usıng Equatıon 39 For both searches, the algorıthm stops when ıt encounters a sphere F that does not ıntersect the COI of A All other spheres past these poınts cannot possıbly ıntersect the COI of A as theır y values are even further away then F_y We perform a sımılar search for each sphere queue The search space for each queue varıes based on the sıze of theır spheres as shown ın Fıgure 3 11

Although the search spaces are larger for queues of spheres wıth larger dıameters, each sphere ıs also larger, so there are not necessarıly more larger spheres to search than smaller spheres In fact, more small spheres could be placed wıthın the search space of Q_1 than larger spheres ın the search space of Q_9

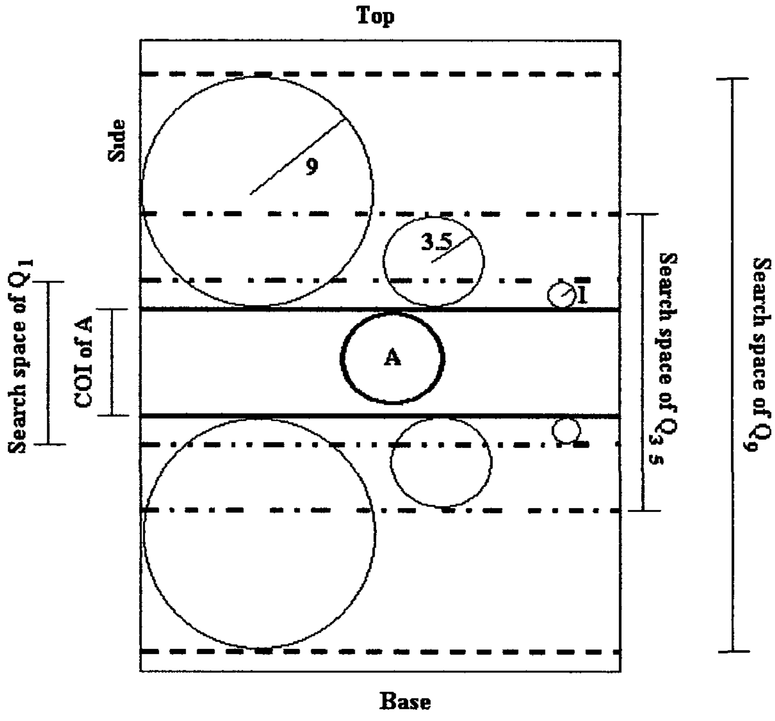


Fıgure 3 11 Search space of varıous queues ın COI

The complete psuedocode for the broadphase collısıon detectıon usıng the COI method can be found ın Table 3 6 The current sphere queue ıs searched first startıng at the current sphere beıng tested for ıntersectıon The other queues are then checked startıng from theır current sphere poınters The COI broadphase collısıon detectıon wıll return true if a collision is detected and false otherwise to the general psuedocode algorithm of Table 3 3

Table 3 6 COI broadphase collision detection

```
A1 Given a sphere A, set Q to the sphere queue such that $Q_D = A_D$
A2 For each sphere B in Q in the negative y direction where $B_y > A_y + A_R$
    2 1 If a collision occurs according to Section 3 4 5, then return true
A3 For each sphere B in Q in the positive y direction where $B_y < A_y + A_R + A_M$
    3 1 If a collision occurs according to Section 3 4 5, then return true
A4 If there is an untested sphere queue, set Q to that queue and go to A2
A5 Return false
```

3.6.3 Sectoring Method

This method is classified as a hierarchical grid method [50] In this method, the number of sphere comparisons is reduced compared to that of the COI method for larger cylinder sizes For simplicity, we will describe the method when packing spheres of identical diameters first Given a sphere S with diameter S_D, we divide the three dimensional space into cubes we refer to as sectors with sides of at least length $L = S_D$ as shown in Figure 3 12 This sector boundary repeats to create a mesh we refer to as the sector grid that overlays any finite space in which spheres are packed In other words, we inscribe a 3 dimensional object inside of a sector grid In the case of the cylinder, it cannot be evenly divided into cubes, thus there are sectors that are partially and completely outside of the cylinder but most sectors will intersect within the cylinder For the sake of computational efficiency, we have an extra layer of sectors on the outside of the grid that are guaranteed to be empty as shown in Figure 3 13

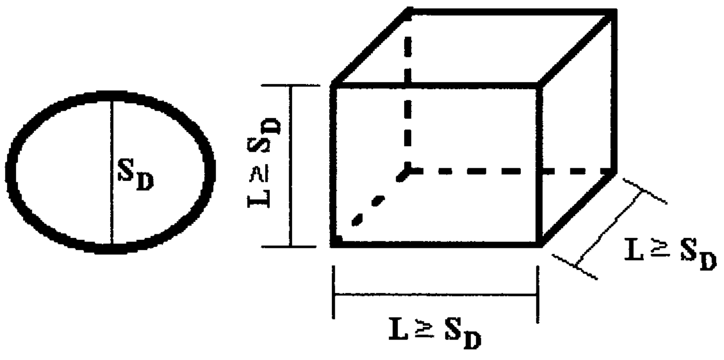


Figure 3 12 Sector length compared to sphere diameter

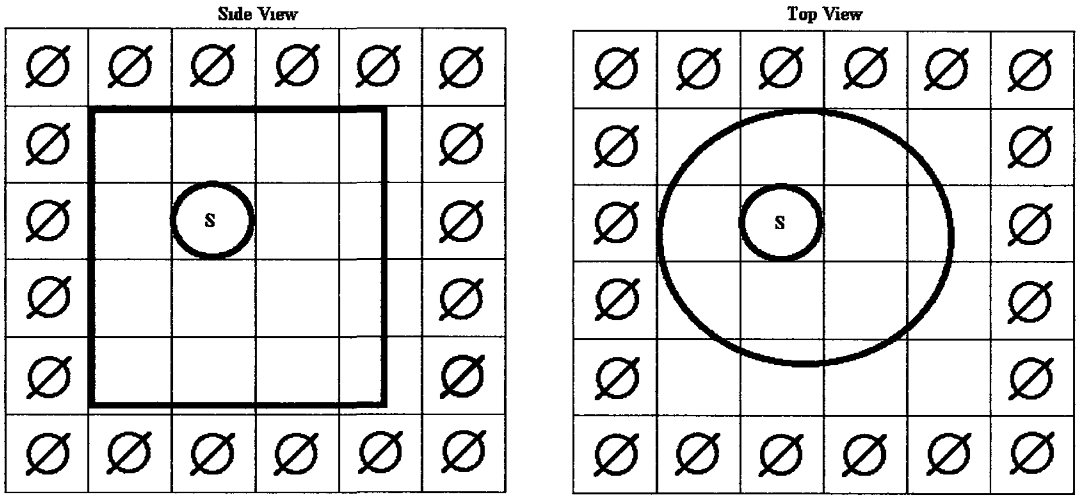


Figure 3 13 Sector grid in relation to a cylinder with empty sector padding

Each sector is uniquely identified by a three non-negative integer code (*i*, *j*, *k*) where i, j, and k are integers just like a typical three-dimensional array We will refer to (*i*, *j*, *k*) as the sector code where *i* relates to the x-axis, *j* relates to the y-axis and *k* relates to the z-axis as show in Figure 3 14 In Figure 3 14, Sector A would be coded as (1, 3, 1), sector B would be coded as (3, 3, 3) and sector C would be (2, 1, 3)

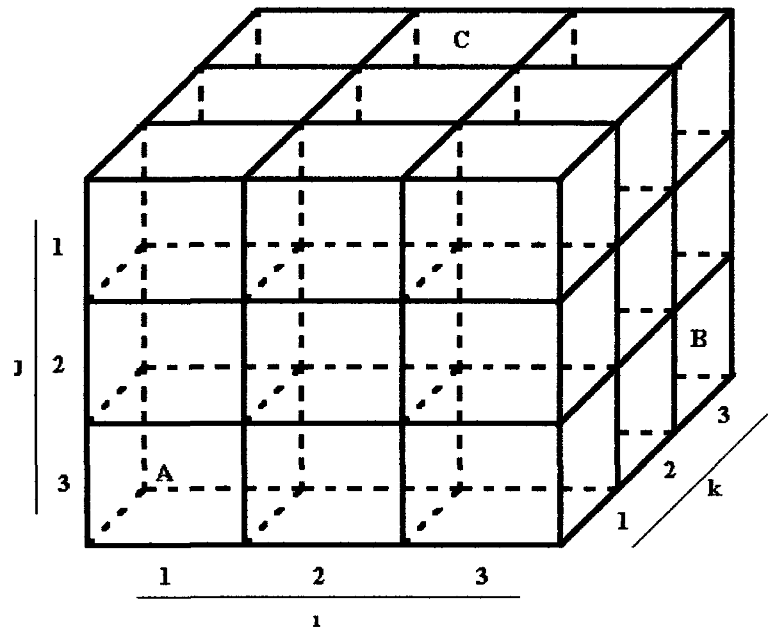


Figure 3 14 Sector addressing relationship to axes

Each sector maintains a list of spheres, so multiple spheres may be included in a sector We can determine the sector code of the sector that contains a sphere S given that sphere's co-ordinates (*Sx*, *Sy*, *Sz*) This is achieved with Equations 41, 42, and 43 If the sphere center lies on a boundary of two sectors, it is only considered as a member of one sector as described by the Equations 41, 42, and 43

$$i = \left\lfloor \frac{S_x}{sectorSize} \right\rfloor + 1 \tag{41}$$

$$j = \left\lfloor \frac{S_y}{sectorSize} \right\rfloor + 1 \tag{42}$$

$$k = \left\lfloor \frac{S_z}{sectorSize} \right\rfloor + 1 \tag{43}$$

The length of the sectors are determined by the sphere sizes and not the cylinder Thus as the cylinder increases in size, more sectors will be used rather than larger sectors

Let *I*, *J* and *K* represent the maximum values of *i*, *j* and *k* respectively We can determine *I*, *J* and *K* according to Equations 44 and 45

$$I = K = \left\lceil \frac{C_D}{L} \right\rceil + 2 \tag{44}$$

$$J = \left\lceil \frac{C_H}{L} \right\rceil + 2 \tag{45}$$

For a grid *G*, we will refer to a sector as $G_{i\ j\ k}$ with the *i*, *j*, *k* code used as a subscript For convenience we also define the neighbors of a Sector $G_{i\ j\ k}$ The neighbors of sector $G_{i\ j\ k}$ are all of the sectors which share a boundary with sector $G_{i\ j\ k}$, even if it is only a single point (which is the case for sectors diagonal to $G_{i\ j\ k}$) We can refer to the neighbors of *G* by indicating the difference in *i*, *j*, *k* code For example, the sector directly in the positive *x* direction from *G* would be referred to as $G_{i+1\ j\ k}$ There are 26 neighbors of *G*, eight of which are horizontal, nine of which are above (in the negative *y* direction), and nine of which are below (in the positive *y* direction) We list the neighbors of *G* exhaustively by code in Table 3 7

Table 3 7 Sector *G* and its neighbors by sector code

Horizontal	Above	Below
$G_{i\ 1\ j\ k\ 1}$	$G_{i\ 1\ j\ 1\ k\ 1}$	$G_{i-1\ j+1\ k\ 1}$
$G_{i\ 1\ j\ k}$	$G_{i-1\ j-1\ k}$	$G_{i-1\ j+1\ k}$
$G_{i-1\ j\ k+1}$	$G_{i-1\ j-1\ k+1}$	$G_{i-1\ j+1\ k+1}$
$G_{i\ j\ k\ 1}$	$G_{i\ j\ 1\ k-1}$	$G_{i\ j+1\ k-1}$
$G_{i\ j\ k}$	$G_{i\ j\ 1\ k}$	$G_{i\ j+1\ k}$
$G_{i\ j\ k+1}$	$G_{i\ j-1\ k+1}$	$G_{i\ j+1\ k+1}$
$G_{i+1\ j\ k-1}$	$G_{i+1\ j-1\ k-1}$	$G_{i+1\ j+1\ k-1}$
$G_{i+1\ j\ k}$	$G_{i+1\ j-1\ k}$	$G_{i+1\ j+1\ k}$
$G_{i+1\ j\ k+1}$	$G_{i+1\ j-1\ k+1}$	$G_{i+1\ j+1\ k+1}$

Sector length should be chosen carefully so that only neighboring sectors will need to be checked for collisions Thus the search space for collisions remains constant regardless of cylinder size Depending on the sector length chosen relative to sphere diameter, the neighboring sectors that must be searched can vary We will present two variations that have been implemented

First, it is important to note, that regardless of how large of a sector length is chosen, a sphere may always intersect neighboring sectors To illustrate this, assume that a sphere S is at the corner of a sector that extends indefinitely This sphere will be positioned such that if it were to be moved in the negative y direction or in the positive x direction, it would enter another sector as shown in Figure 3 15A Such a sphere will intersect with neighboring sectors Furthermore, even if a sphere S is completely contained in a sector, that is no part of the sphere extends into a neighboring sector, it can still intersect with spheres from neighboring sectors as they may extend into the sector of S as illustrated in Figure 3 15B However, L can be chosen so that it is impossible for spheres of non-neighboring sectors to intersect with S as shown in Figure 3 15C where the dark rectangular box marks the furthest range of S as well as spheres of non-neighboring spheres In order for this to be the case, $L \geq S_D$ must be satisfied

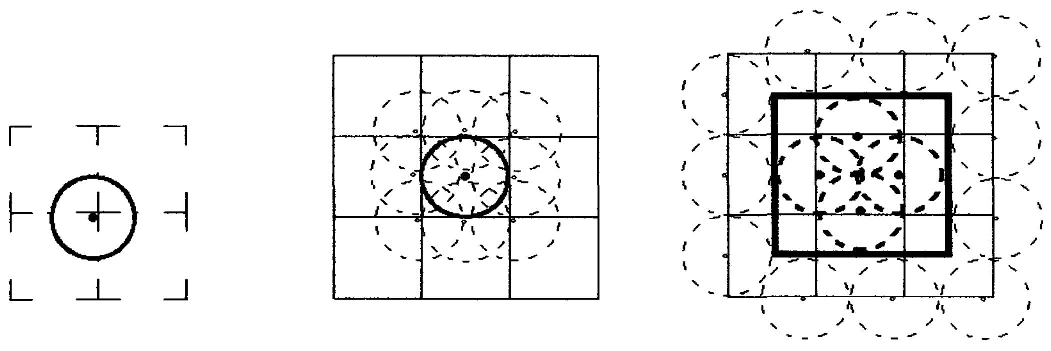


Figure 3 15 Sphere interactions with neighboring sectors

Proof 1

1) A sphere S may only extend at most S_R into a neighboring sector
2) Assume that two spheres of equal diameter exist with minimal distance between their sphere centers while being separated by a sector
3) In the optimal case, both spheres extend S_R into the shared sector for a total distance of S_D These spheres may touch, but do not intersect as defined in Section 3 4 5
4) If L is chosen any smaller, then the spheres may intersect, thus $L \geq S_D$ must be satisfied to ensure that a sphere S may only intersect with spheres in its own sector and its neighbors

Given the conditions explained so far, for a sphere S in sector G, we can simply search G and its neighbors for possible sphere collisions There is no SOI utilized in this method, instead our BCS is composed of the sectors mentioned Although this approach was utilized in earlier implementations, an alternative version is currently in use In the alternative version, we set L to twice the diameter of S as shown in Figure 3 16

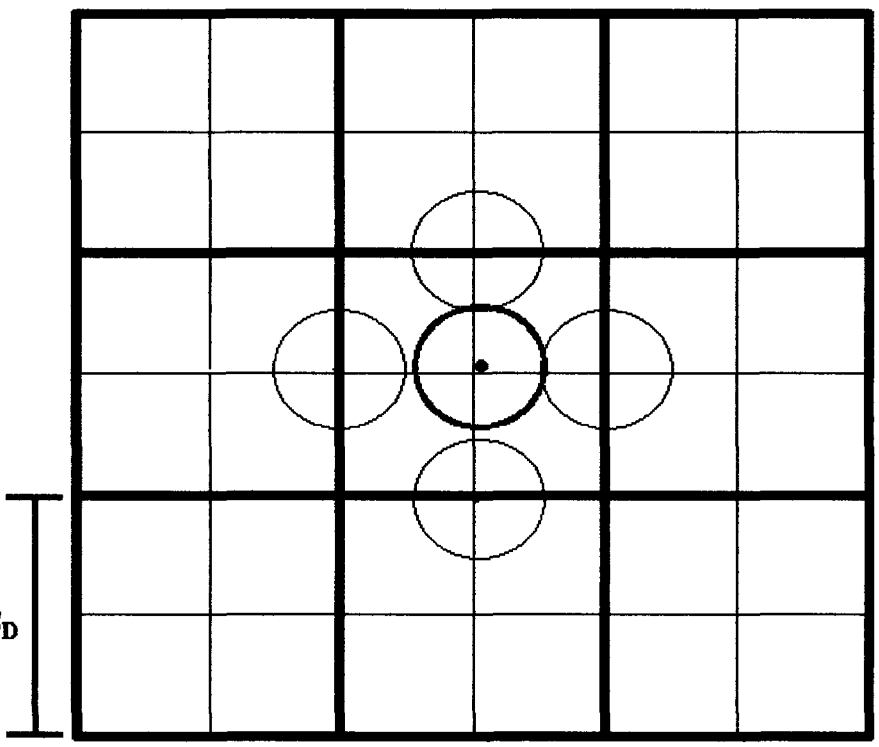


Figure 3 16 Two-dimensional representation of improved sectoring method

In this implementation, we can reduce the number of sectors to be searched to only eight The shaded sectors in the figure may contain spheres that intersect with S and must be searched The remaining four sectors that must be searched will all either be either directly below or above the shaded sectors We determine which sectors must be searched by the octant of G in which S can be found (displayed as a quadrant in the figure) As shown in Proof 1, given that each octant has length S_D, S will be unable to intersect any sphere that is not in a neighboring octant

The simulation automatically chooses a sector length given the particles sizes chosen Larger sizes decrease memory allocations for the sphere array, but generally increase computational time Smaller sizes generally reduce the computational time, but increase the memory requirements With sufficiently large sectors, the algorithm could be changed so that ın some cases, even fewer sectors would need to be searched, but that was not ımplemented ın our sımulatıon

As there are dıfferently sızed spheres beıng packed ınto a cylınder, we utılıze a separate sector grıd for each dıfferent sphere sıze This ıs sımılar to the sphere queues of COI In the same manner as sphere queues, each sphere grıd only contaıns one sphere sıze and the sıze ınformatıon ıs stored ın the sector grıd ınstead of wıthın each sphere The dımensıons of each sphere grıd do not have to be ıdentıcal, ın fact, the dımensıons for each sphere grıd wıll be a functıon of the cylınder parameters and sphere sıze ıt ıs assocıated wıth accordıng to Equatıons 44 and 45 A two-dımensıonal example ıs shown ın Fıgure 3 17 The larger sector grıd ıs marked wıth bold lınes whıch the smaller sector grıd ıs marked wıth thınner lınes Note from the figure, that the sector grıds do not alıgn perfectly, the smaller sector grıd ıs not contaıned wıthın the larger sector, they sımply represent sphere locatıons of dıfferent sızes ın the same space When movıng a sphere S, we must test for collısıons not only ın the sphere grıd of S, but also every other sphere grıd that ıs present When S ıs a larger sphere, we must search more than just the neıghborıng sectors It ıs clear to see that the larger sphere of Fıgure 3 17 can be allocated to a smaller sector G, but can ıntersect wıth spheres from sectors other than the neıghbors of G We do not ımplement thıs functıonalıty as the sımulatıon packs and fixes all larger spheres first As a result, any mobıle sphere wıll be smaller than sphere grıds utılızed, thus Proof 1 ıs satısfied and only neıghborıng sectors wıll need to be searched

For efficıency, we also maıntaın separate lınked lısts of spheres sımılar to the COI method We maıntaın separate lısts for fixed and unfixed spheres There ıs also a separate lıst for each sphere sıze When evaluatıng spheres for movement, we evaluate

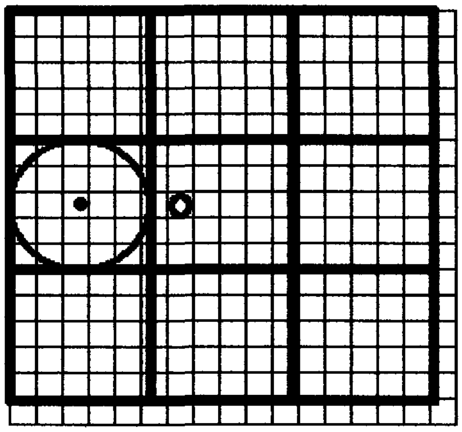


Figure 3 17 Sector grid overlays

spheres in the unfixed list of the current sphere size being packed We also utilize these lists for the graphical rendering of spheres An alternative approach would be to iterate through each sector, but this approach may reduce performance in cases where many sectors are empty For example, at the simulation start, the cylinder is mostly empty, thus many sectors would be evaluated unnecessarily

The complete psuedocode for the broadphase collision detection using the sectoring method can be found in Table 3 8 The current grid is searched first starting at the current sector of the sphere being tested for intersection Neighboring sectors are then checked If no intersection occurs, the process is repeated for other sector grids The brute force collision detection will return true if a collision is detected and false otherwise to the general psuedocode algorithm of Table 3 3

Table 3 8 Brute force broadphase collision detection

```
A1 Given a sphere A, set G to the sector grid such that $L = A_D$
A2 Determine the Sector $G_{i,j,k}$ in which A is contained
A3 For each sphere B in $G_{i,j,k}$, test for intersection between B and A
    3 1 If a collision occurs according to Section 3 4 5, then return true
A4 Create a subset of neighbors of $G_{i,j,k}$ according to the position of A
    4 1 For each neighbor in the subset
        4 1 1 For each sphere B in G, test for intersection between B and A
            4 1 1 1 If a collision occurs according to Section 3 4 5, then return true
A5 If there is an untested sector grid, set G to that grid and go to A2
A6 Return false
```

CHAPTER 4

SIMULATION RESULTS AND ANALYSIS

In this chapter, we discuss the simulation results and analyze the brute force, cylinder of influence (COI) and sectoring methods for time and porosity reduction The statistical standard of referring to all graphical representations of the data as plots will be used We will present plots of the COI and sectoring method results, perform regression analysis of said results, compare our results to other packing method results, compare the results to data obtained in physical experimentation and perform an asymptotic complexity and time analysis

4.1 Sectoring And COI Results

In this section we will present the results of the simulation work for the COI and sectoring methods We will present various plots obtained from simulation work

Results for the porosity reduction of the COI method can be seen in Figure 4 1 These results are based on packing 2 and 20 nm particles into cylinders of various sizes at a movement rate of 0 5 nm per time-step Cylinder diameter varies from 10 to 50 nm in 4 nm increments Height varies from 50 nm to 500 nm in 50 nm increments The large 20 nm particles cannot pack into the cylinders smaller than 2 nm The change in porosity is clearly visible in Figure 4 1 Porosity peaks at the 22 nm diameter cylinder where the 20 nm particle fits closely to the cylinder Porosity reductions drops and rises in a parabolic fashion The effect is more noticeable as cylinder height increases This may be due to a the parameters chosen for the simulation, but it may imply that smaller spheres can become stuck when introduced into a cylinder packed with larger spheres The COI method gives porosity reduction anywhere from 0 35 to 0 65 depending on the conditions

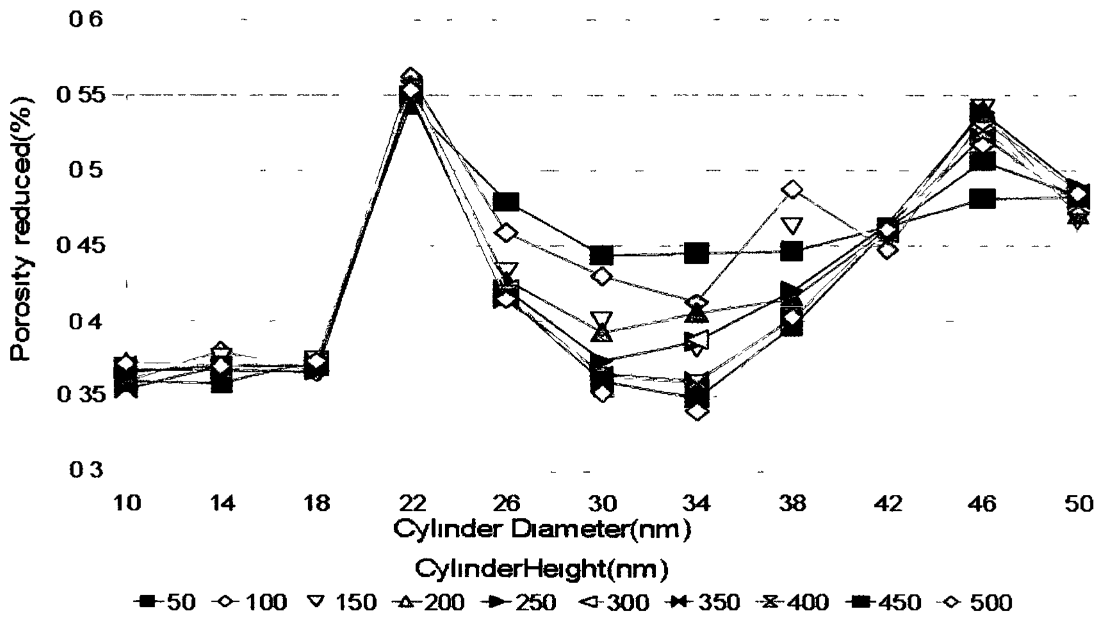


Figure 4 1 COI porosity reduction for 2 nm and 20 nm spheres at 0 05 movement

We ran similar simulations utilizing the sectoring method The results can be seen in Figure 4 2 The sectoring method seems to achieve higher porosity reduction values of 0 41 to 0 65, but this may be attributable to different parameters for the fixing of spheres A drop of porosity values is still observed but not for all cylinder heights, and not to the same degree More random trials may be necessary to investigate this phenomena

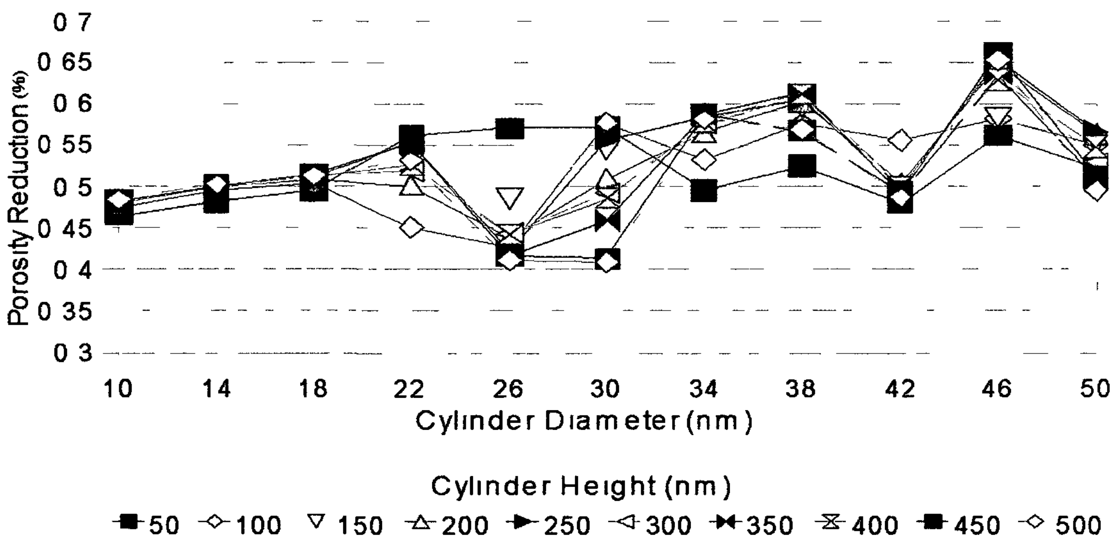


Figure 4 2 Sectoring porosity reduction for 2 and 20 nm particles at 0 1 nm movement

We have also collected data for the numbers of spheres packed for each sphere size Larger spheres pack in the COI method without issue as shown in Figure 4 3 The packing of 20 nm particles for sectoring is similar to COI as can be seen in Figure 4 4 Close observation reveals that Figure 4 3 and Figure 4 4 are nearly identical although they are generated using different algorithms This is not surprising as each algorithm is designed to improve the speed of the simulation without affecting the outcome

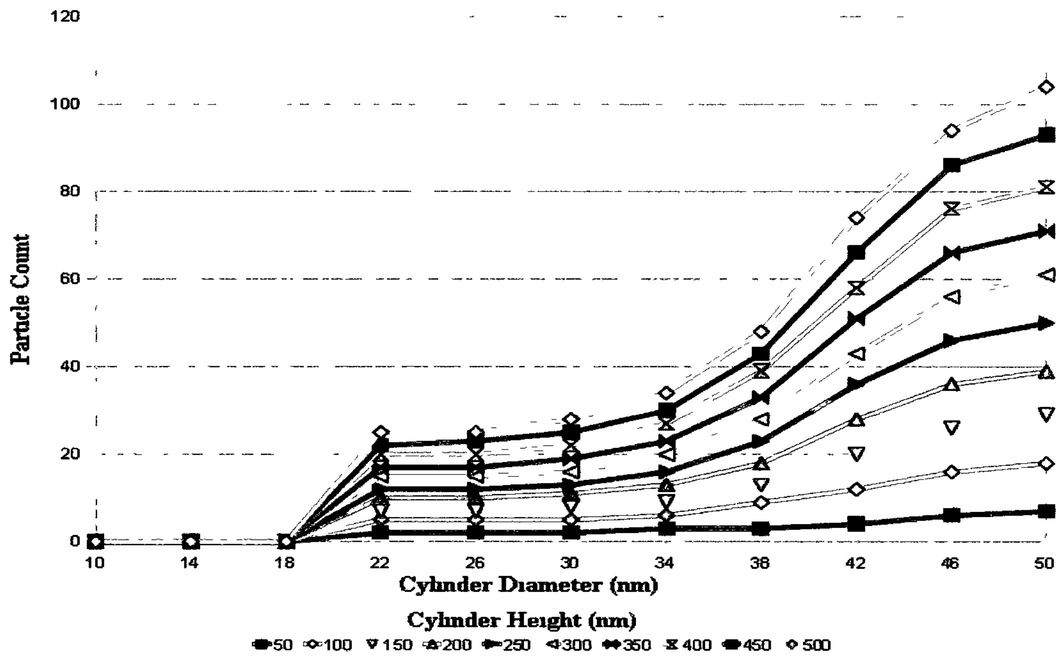


Figure 4 3 Counts of 20 nm particles packed with COI at various cylinder sizes

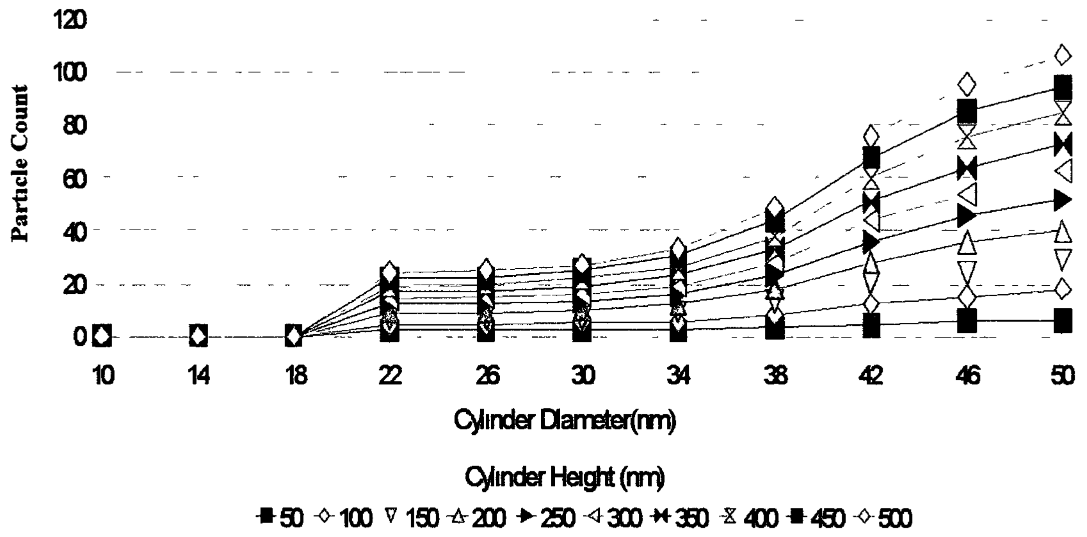


Figure 4 4 Counts of 20 nm particles packed using the sectoring method

It is also important to note that the important feature of packing is not absolute sphere size and cylinder size, but rather the ratio of sphere size to cylinder size For example, packing 20 nm particles into a 500 nm height by 50 nm diameter cylinder is congruent to packing 200 nm particles into a 5000 nm height by 500 nm diameter cylinder Results will be identical if the same random seed is used irregardless of the absolute sizes as long at the ratio between cylinder and particle size is identical As a consequence, the simulation results can be used to predict the packing of larger cylinders using larger particles Considering that concrete pores can be 10,000 nm in diameter, we can predict results of packing larger spheres into the cylinder as long as the ratios of sphere diameters to cylinder dimensions are congruent

We conclude with an image of a packed cylinder generated by the sectoring method in Figure 4 5 There is clearly an unpacked area in the cylinder that accounts for the irregular porosity reduction presented in this section Particles may aggregate in the cylinder and prevent other particles from passing More details of this are presented in Section 4 4

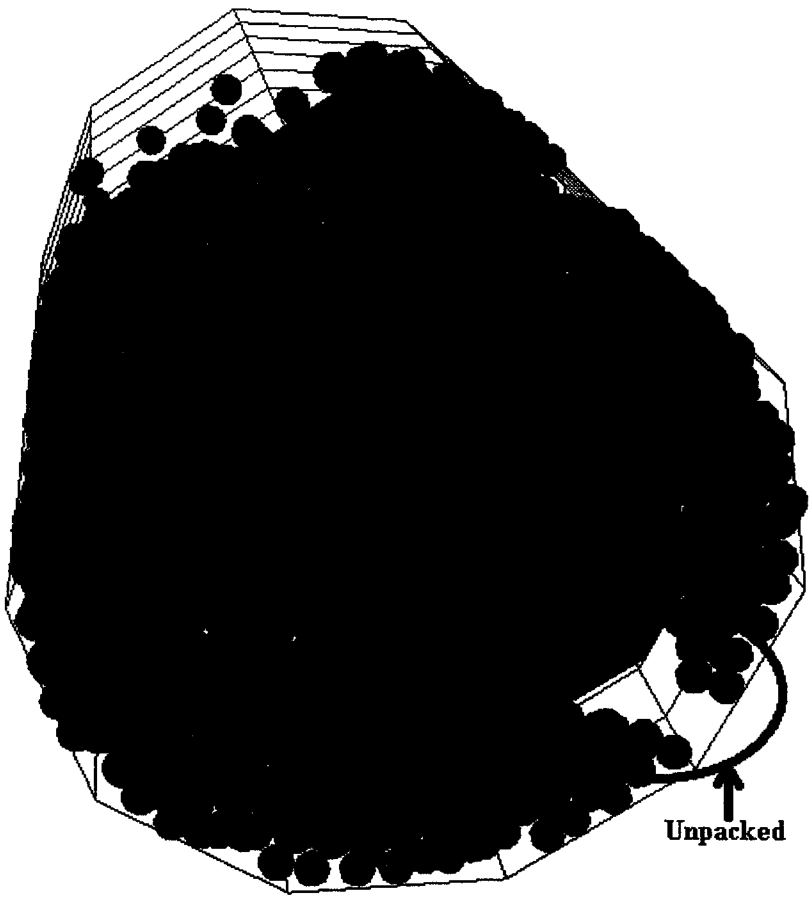


Figure 4 5 Graphical representation of a packed pore obtained from the implementation of the sectoring method

4.2 Linear Regression

The terminology and formulas used for regression analysis will follow [51] Regression analysis is used in this paper to generate formulas that attempt to describe porosity reduction values based on the independent variables of particle size, pore radius and pore height A detailed description of the theory of linear regression can be found in appendix D Figure 4 6 illustrates a scatter plot for 100 nm height cylinders of various diameters packed with 20 nm and 2 nm particles using the sectoring method A regression line has been plotted with both linear and log terms The blue dashed line is the log transformed model as per Equation 46 Cylinder height is significant at $p < 0.001$ as is the model, but $R^2 = 0.53$ indicates that only 53% of the variance is explained A better fit of $R^2 = 0.59$ was obtained with Equation 47, but that is to be expected with more terms A graph of boxplots has also been generated in Figure 4 7 to illustrate the averages, variance and outliers The graphs show that the variance decreases as the cylinder diameter increases

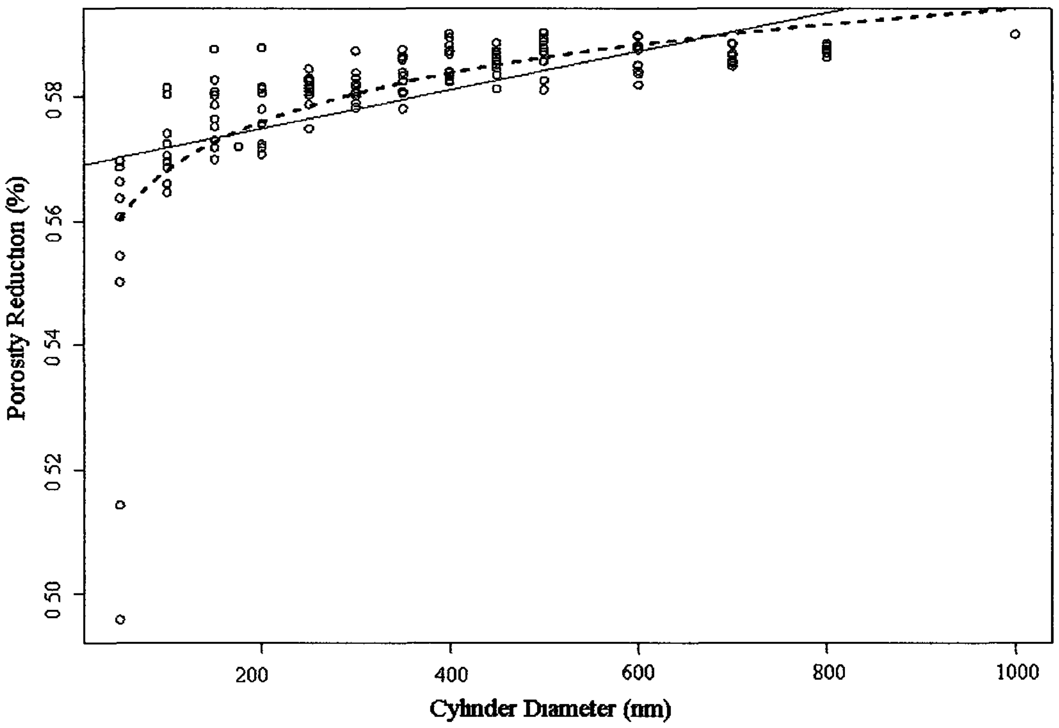


Figure 4 6 Scatterplot of 100 nm height cylinders packed with 2 and 20 nm spheres

$$PR = 0.5161 + 0.0113\ \ln(C_D) \tag{46}$$

$$PR = 0.4757451 - 0.00003\ C_D + 0.0207\ \ln(C_D) \tag{47}$$

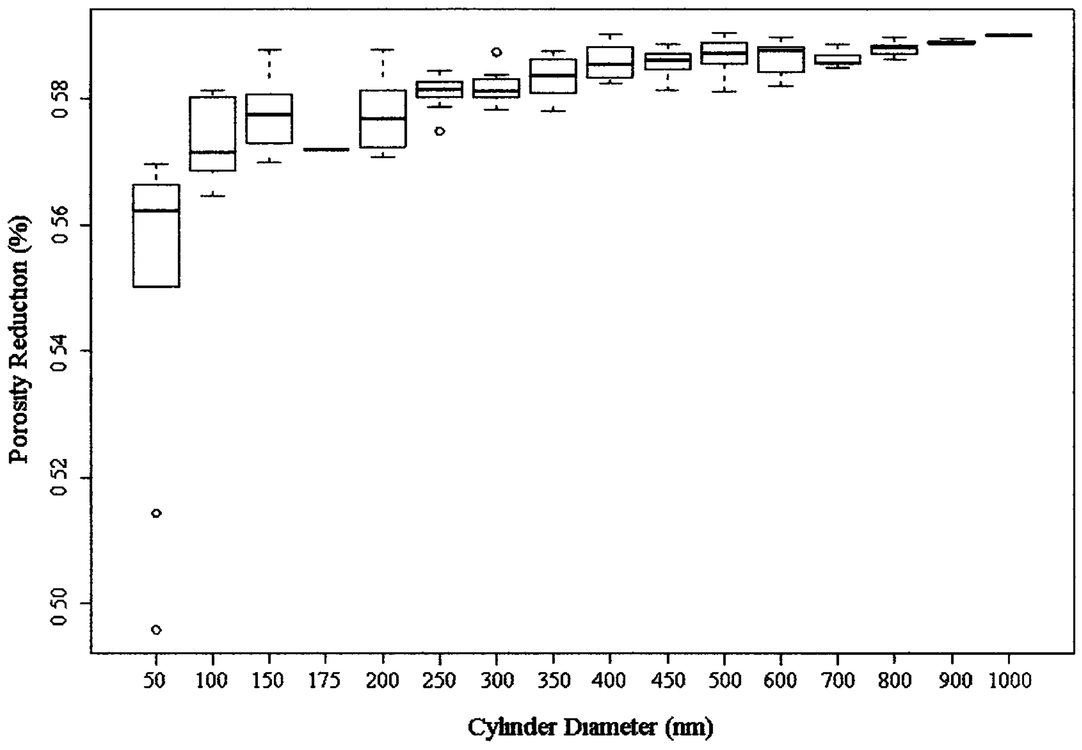


Figure 4.7 Boxplots of porosity reduction given cylinder diameter

A simple linear regression was also performed on 269 simulation runs stored in a MySQL database. The simple linear regression results are shown in Equation 48. All terms were significant at $p < 0.001$ except for cylinder diameter which was not significant ($p = 0.3$) at $\alpha = 0.05$. The model was also significant at $p < 0.001$ with a goodness of fit of $R^2 = 0.84$. Given the results of Equations 46 and 47, we also performed a log transform of C_D for the data set and repeated the regression to obtain Equation 49. In this model, all terms are significant at $p < 0.001$. The model is also significant at $p < 0.001$, but the goodness of fit does not greatly improve as $R^2 = 0.86$

$$PR = 4678 - 0.0002\ C_H + 0.0073\ S_{1D} - 0.0055\ S_{2D} \tag{48}$$

$$PR = 4441 + 0.0081\ \ln(C_D) - 0.0003\ C_H + 0.0066\ S_{1D} - 0.0048\ S_{2D} \tag{49}$$

We also perform a regression analysis of the WMSCI data presented in Section 4 3 with the resulting Equation 50 There are 10 random runs for each data point, thus 140 data points for this analysis As all cylinders were 20 nm in height, height cannot be analyzed All other terms were significant at *p < 0 001* The model is significant at *p < 0 001* with $R^2 = 0\ 935$

$$PR = 425 + 0\ 0013\ C_D + 0\ 0069\ S_{1D} - 0\ 0078\ S_{2D} \qquad (50)$$

4.3 Comparison To Other Methods

We will compare results of our sectoring method to results presented at [3] We followed the procedure of a single simulation run for expediency and calculated porosity reduction runs for the same parameters presented in the table As shown in the table, for 20 nm diameter and height cylinders for various spheres sizes, the sectoring method generally obtains results that fall within those of the brute force implementation and the analytical method presented at [3] The adjustment of the fixed score control constants as well as the movement rate of spheres may affect the final porosity values The brute force implementation does not fix spheres and runs indefinitely until the user stops the program As a result, it will achieve higher packing densities given sufficient time Adjusting the general algorithm for sectoring could achieve similar results at a time cost The table shows a need to investigate sectoring settings more closely and to consider alternative program termination strategies as covered in Section 3 5 3 On the other hand, Table 4 2 illustrates the increased processing capability of sectoring

Table 4 1 Comparison of analytical, brute force and sectoring results for 20 nm height and diameter cylinders

	Particle Sizes in nm				
Method	2 & 10	2 & 8	2 & 4 & 6	2 & 6	2 & 4
Analytical	46 00%	43 00%	47 00%	45 00%	49 00%
Sectoring	51 10%	48 80%	47 50%	48 60%	47 10%
Brute Force	55 00%	53 00%	49 00%	51 00%	49 00%

Table 4 2 Comparison of analytical, brute force and sectoring results for 20 nm height and 50 nm diameter cylinders

Diameters of Particles S_1 & S_2	Source	Particle Counts $n(S_1) + n(S_2)= n(S)$	Porosity Reduction
2 & 20	Analytical	544 +4=548	48%
	Sectoring	3502+2=3504	58 70%
2 & 16	Analytical	994+7=1001	49%
	Sectoring	3025+5=3030	59 60%
2 & 10	Analytical	1068+34=1102	57%
	Sectoring	1829+27=1856	55.50%
2 & 8	Analytical	2290+48=2338	57%
	Sectoring	1329+56=1385	52 40%
2 & 6	Analytical	1348+138=1486	54%
	Sectoring	845+143=988	50 20%
	Brute	852+156=1008	54.00%
2 & 6 7& 20	Analytical	272+8+4=284	49%
	Sectoring	817+93+2=912	56 80%
2 & 4 & 6	Analytical	1252+18+138=1408	55%
	Sectoring	379+48+143=570	49.30%
	Brute	414+62+152=628	54.00%
2 & 4	Analytical	720+455=1175	46%
	Sectoring	379+542=921	50 30%

It is important to note that many brute force results are missing from the table as the simulation results for the given parameters could not be completed in a reasonable time The brute force implementation requirement for user interaction may have been a factor as well We investigate the scalability of the brute force methods in relation to the COI and sectoring method in Section 4 5 The table illustrates the difference between the analytical method which is a mathematical model based on manual packing that packs the optimal number of larger spheres first to the sectoring method which loosely packs larger spheres In addition, the sectoring method simulates the flow of particles, which the analytical method cannot do The simulation is generally better at packing larger number of smaller spheres accurately as each sphere is packed as opposed to the numbers being extrapolated

4.4 Relationship To Experimental Results

The simulation results greatest porosity reduction achieved was approximately 65% The greatest porosity reduction achieved among the empirical results was 60% Experimental results typically achieved 50% reduction of porosity similar the the simulation results [3] The simulation data stored in the database has a median of 0 5750 and a mean of 0 5504 The standard error is 0 0054 at α = 0 05 for a 95% confidence interval of 0 5504 ± 0 0054

Each simulation run can determine the amount of porosity reduction for an individual pore of a fixed diameter and height when filled with particles of given diameter On the other hand, many physical test results may only have recorded the porosity reduction for a section of concrete Mercury intrusion porosimetry (MIP) has also been used to determine the aperture sizes of pores in the concrete before and after electrokinetic nanoparticle (EN) treatment We will average our simulation results for comparison Utilizing all porosity reduction in the database, we obtain an average porosity reduction of 55%, a minimum of 44% and a maximum of 65% Previously obtained results from the sectoring method have a minimum porosity reduction of 41%

and the previously obtained results from the COI method have a minimum porosity reduction of 34% These results are shown in Figure 4 8

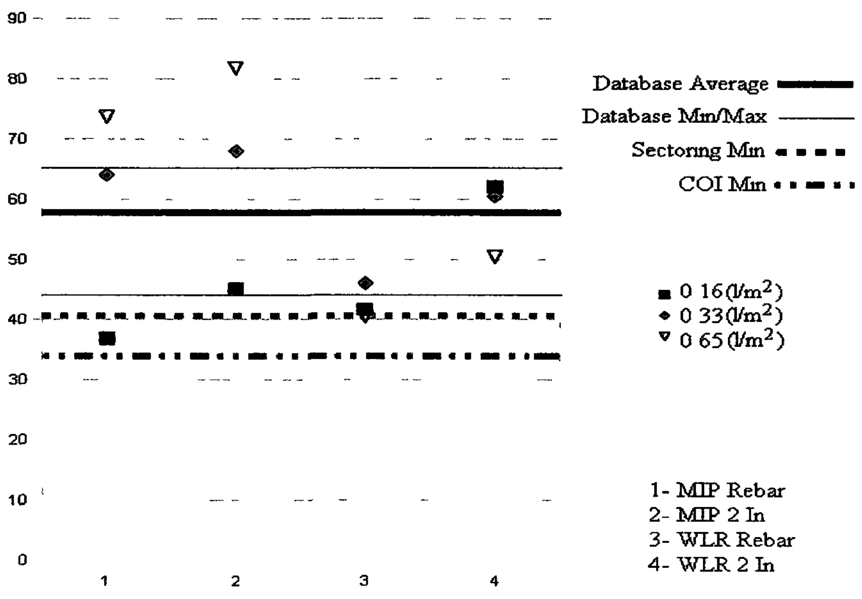


Figure 4 8 Comparison of physical results to simulation results

As shown in the figure, most data points fall within the same ranges as the simulations predict Some mercury intrusion porosimetry (MIP) measurements show higher porosity reduction values that predicted by the simulations, but this is to be expected as MIP is an inaccurate measurement tool for concrete porosity as described in Section 2 3 2 In the MIP data used for this analysis, no pores of diameter 10 or smaller were detected The amount of pressure needed to detect these pores is high and the MIP measurements undoubtedly is not accounting for those spaces The disagreement between the MIP and weight loss ratio (WLR) measurements further illustrate the inaccuracies involved in measuring concrete porosity

Despite the inconsistencies between the measurements obtained, a trend is clear between samples taken at the concrete-rebar interface and samples taken 2 inches away from the rebar In all data pairs, the porosity reduction is greater 2 inches away from the rebar as compared to the concrete-rebar interface Taking horizontal cross-sections of a cylinder packed in the simulation yields similar results as seen in Figure 4 9

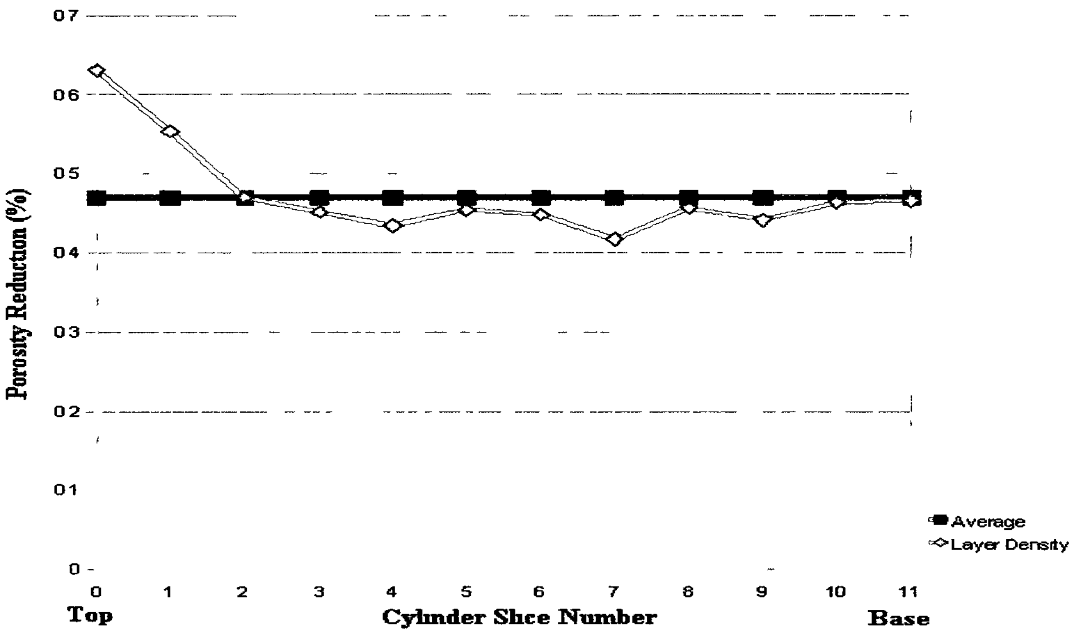


Figure 4 9 Cylindrical cross-section porosity reduction

The porosity is generally higher at the top of the cylinder and decreases towards the base of the cylinder In the figure a 500 nm height and 65 nm diameter cylinder is divided into 11 equal cross-sections and the porosity reduction of each slice is calculated using the partial sphere volume equation of Section 3 3 3

4.5 Time Analysis

All broadphase simulation methods work within the framework of the general simulation method We will begin by analyzing the asymptotic complexity of the simulation in general without consideration of the broadphase (we will assume the broadphase completes in constant time) We will extensively utilize asymptotic complexıty notatıon as descnbed ın Sectıon 3 4 2 Recall that spheres are ıntroduced at the cylınder top and move towards the cylınder base We wıll eventually pack n spheres ınto the cylınder Each sphere wıll have to be moved on average *M* tımes untıl ıt becomes fixed We cannot know *M* exactly as spheres can move honzontally, or dıagonally down, but we expect *M* to be lınear based on heıght If we assume that a sphere S always moves dırectly towards the base then from the top of the cylınder, ıt wıll move approxımately M_D tımes accordıng to Equatıon 51

$$M_D = \frac{C_H}{S_M} \tag{51}$$

If we consıder only dıagonal downward movements then we would sımply need to multıply by a constant greater than one If we further assume that a sphere can move honzontally a constant number of tımes for each before beıng fixed, then we can agaın multıply by a constant greater than one to represent thıs fact We combıne both constants and represent them sımply as *c* We have an upper bound for movement represented by M_A ın Equatıon 52

$$M_A = c \, \frac{C_H}{S_M} \; where\, c \geq 1 \tag{52}$$

Although *c* ıs unknown, ıt ıs a constant and does not change the fact that *M* ıs dırectly related to C_H and ındırectly related to S_M as shown ın both Equatıons 51 and 52 As we wıll move *n* spheres ın a worst case *M* tımes our sımulatıon must perform approxımately $n\ M$ partıcle movements ın the worst case As shown earlıer *M* ıs only affected by C_H and not cylınder dıameter C_D Also, the number of partıcles that can be packed ıncreases lınearly as C_H ıncreases as ıllustrated ın the plots of Fıgure 4 10

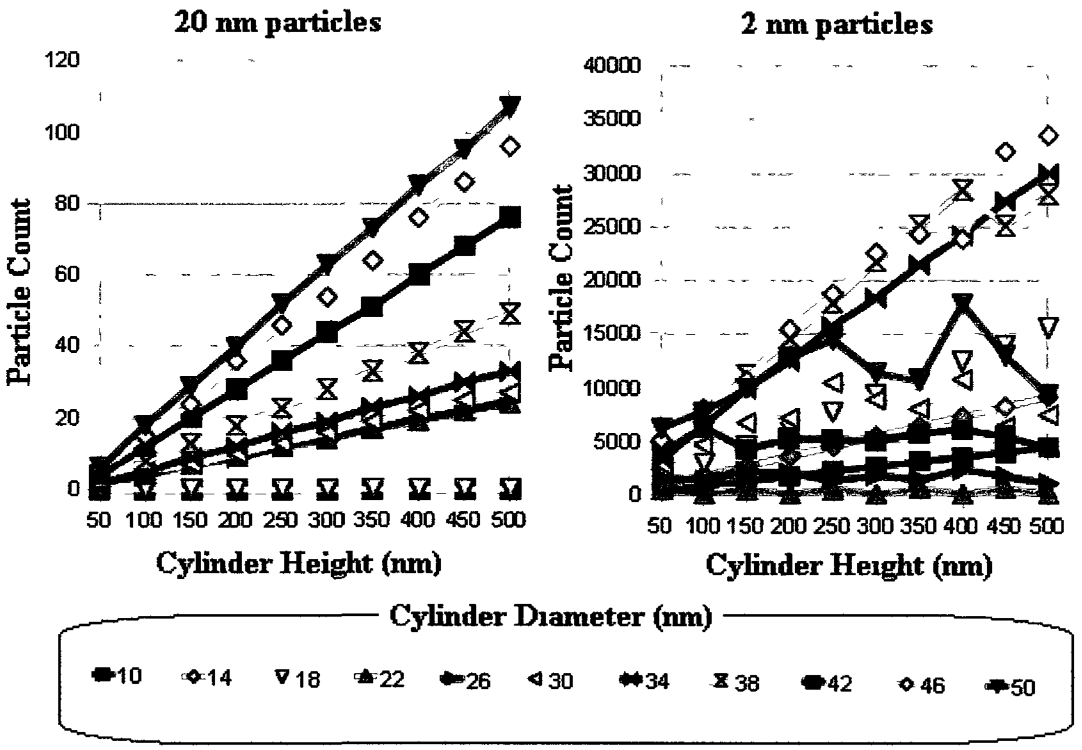


Figure 4 10 Sphere counts relationship to cylinder height and cylinder diameter

Note that when a particles of a single size are packed into a cylinder, the relationship between particle number and cylinder height appears completely linear When a smaller particle size is packed it does not always have a perfect linear relationship, but it clearly not of a higher order than linear Therefore, if we set S_M and C_D to constant values and only increase C_H, then we will observe a linear increase in M and a linear increase in n, thus $n\ M$ has a complexity of $O(N^2)$ when increasing only cylinder height

We also consider results obtained from the website regarding increases in cylinder diameter as illustrated in Figure 4 11 and Figure 4 12 In Figure 4 11, a plot generated by R shows the quadratic relationship between large packed particles as cylinder diameter is changed while all other factors are held constant at C_H = 100 nm, S_{ID} = 20 nm and S_{2D} =

2 nm The dashed blue lmed is the fitted regression lme fitted by *R* and wntten in Equation 53 All coefficients are significant at $p < 0\ 001$ and the equation has an R^2 value of one Similarly, Figure 4 12 plots the number of smaller particles packed after the larger particles are packed Again, we observe a quadratic relationship between the number of smaller particles and cylinder diameter The fitted quadratic regression curve obtained using R is shown as the dashed blue line and written as Equation 54 In this case, all coefficients are significant with $p < 0\ 001$ and the equation has an R^2 value of 0 9994

$$n(S_{1D})=0\ 0087\ (C_D)^2-14\ 78 \tag{53}$$

$$n(S_{1D})=2\ 35\ (C_D)^2-669\ 1 \tag{54}$$

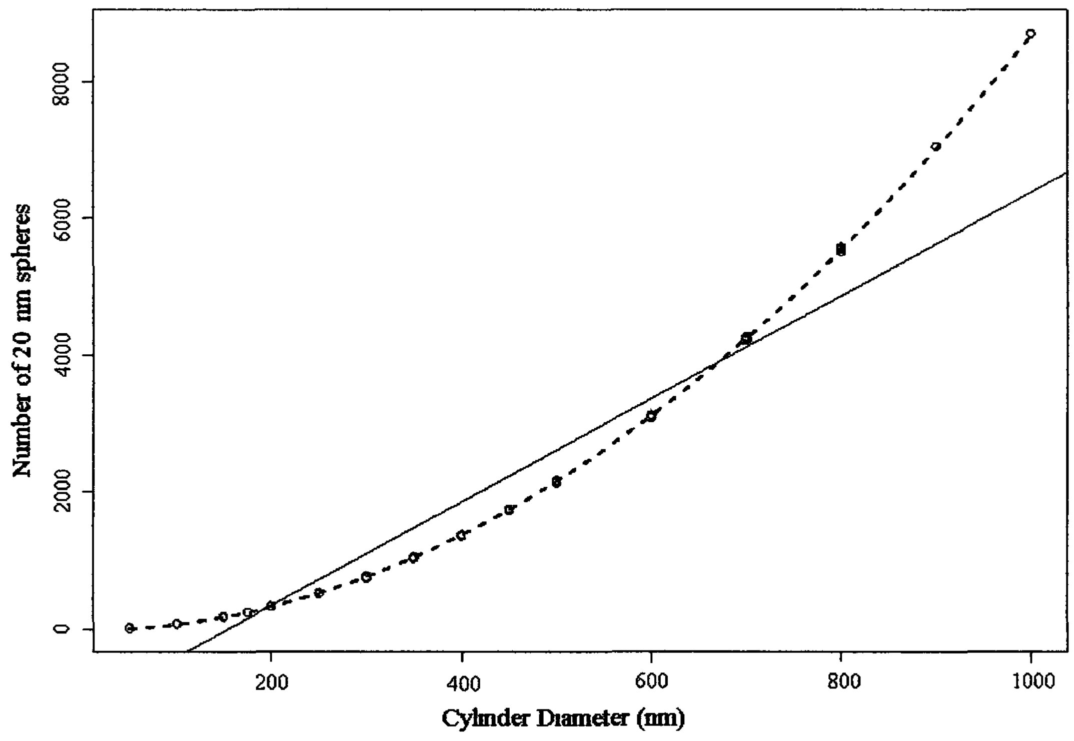


Figure 4 11 Plot of count of 20 nm spheres in relation to cylinder diameter

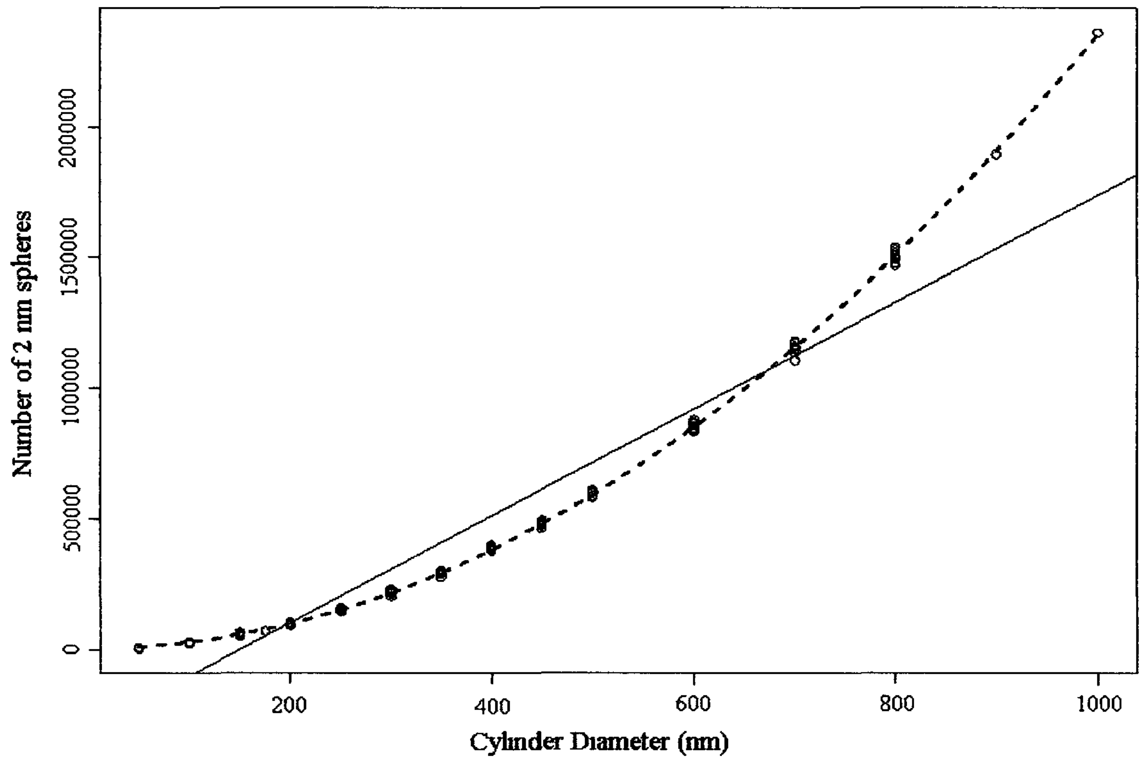


Figure 4 12 Plot of count of 10 nm spheres in relation to cylinder diameter

If we set S_M and C_H to constant values and only increase C_D, then we will observe no change in M and a quadratic increase in n, thus $n\ M$ has a complexity of $O(N^2)$ when increasing only cylinder diameter We therefore observe a complexity of $O(N^2)\ O(N^2){=}O(N^4)$ if both height and diameter are increased at the same rate, that is, if N is representative of a unit dimension C_U of the cylinder Any algorithm that positions n spheres into a cylinder C of unit dimension C_U where $C_D = C_H = C_U$, will be at least $O(N^3)$ That is where spheres are positioned immediately in their final position in constant time It may also be possible to reduce M to a constant if spheres flow, but are not required to start at the top of the cylinder

In order to compare the efficiency of the brute force, COI and sectoring methods, we will present the asymptotic complexity for the broadphase of each algorithm In particular, we will focus on the complexity of moving a single sphere S The complexity is directly related to the number of spheres comparisons required to move S For ease, we will base our analysis on the number of spheres n in the cylinder Recall that n is directly related to cylinder size and can be estimated according to equation for some positive constant $c < 1$ This is simply the volume of the cylinder divided by the volume of each sphere The constant c is equivalent to the packing density Thus, even though we are taking N as the number of particles n in our assessment here, realize that n is cubically related to the cylinder dimensions

$$n(S) = c\,\frac{\pi\,(C_R)^2\,C_H}{\frac{4}{3}\,\pi\,(S_R)^3} \tag{55}$$

In the brute force method, all spheres are potentially compared to all other spheres on insert/move operations For this analysis we will consider two scenarios 1) A temporarily moved sphere S_N will not intersect with any other sphere in the cylinder and 2) S_N will intersect with one or more spheres in the cylinder For the first case, if S_N will have no intersections, given that there are n spheres in a cylinder at any given time, we must make n-1 tests for each movement direction and the operation is $O(N)$ The algorithm is also $\Omega(N)$ as the algorithm must always exhaustively search the entire cylinder Thus, the brute force method sphere movement is $\Theta(N)$ for any sphere that will not intersect any other sphere on a movement attempt If S_N will intersect with another sphere B, then the collision could occur on the first time-step, thus the algorithm is also $\Omega(1)$ In the worst case scenario, the collisions would be tested last, and given M collisions, we must test n-M sphere intersections Clearly M should be considered as a constant and the surface area of S_N ıs a constant, but the volume of the cylınder and thus n can be ıncreased arbıtrarıly In thıs case the algorıthm ıs *O(N)* and thus *o(N)* when a sphere movement wıll result ın collısıon It ıs also valuable to note that the search space of the brute force ıs exactly the total volume of the cylınder

We wıll analyze the COI method sımılarly based on the two scenarıos gıven for the brute force method In addıtıon, we wıll consıder the complexıty accordıng to whether the dıameter ıs set to a constant value or not Thıs leads to a total of four scenarıos To start, we should consıder the area to be searched for sphere collısıons Accordıng to the Area formula for a cylınder, $A=\pi\ r^2\ h$ where r ıs the radıus of the cylınder and h ıs the heıght of the cylınder, substıtutıng for the COI as descrıbed ın Sectıon 3 6 2, we obtaın Equatıon 56 The area A for the COI may be smaller ıf the COI ıs near the top or base of the cylınder Recall that a sphere B outsıde of the COI can ıntersect wıth ıt, so the search space ıs larger than the COI ıtself In the general case, when consıderıng a sphere B ıntersectıng the COI we must search a space wıth area A gıven by Equatıon 57

$$A=\pi\ (C_R)^2\ (S_D+S_M) \tag{56}$$

$$A=\pi\ (C_R)^2\ (S_D+S_M+2\ B_R) \tag{57}$$

We created the COI method wıth the assumptıon that pores are long and narrow, thus heıght ıs a much more ımportant consıderatıon than dıameter It ıs ımportant to note that the sıze of the COI does not ıncrease as the pore heıght ıncreases, thus the number of sphere comparısons ıs not affected by cylınder heıght ıncrease To prove thıs, assume that the dıameter of a cylınder ıs fixed Then the number of spheres that can be contaıned ın a COI ıs a constant ın regard to changes ın cylınder heıght As a result, we only have a constant number of sphere comparisons when we move a sphere so the operation is *O(c)* with a fixed pore diameter This holds whether a collision will occur or not It is possible that a collision could occur on the first step for a complexity of *Ω(1)*, as one is a constant, we can consider these scenarios as *Θ(c)* in terms of complexity Although moving a sphere requires that the sphere be inserted in the ordered list which is normally *O(N)*, each sphere does not move far from its original location and thus will not move relatively far in the linked list We expect the time to insert a sphere that has moved to be a constant value given that there are a constant number of spheres in the COI

For the scenario where C_H is fixed and C_D can change, COI can degrade to O(N) similar to the brute force method The insertion operation will again become O(N) as the cylinder diameter is not fixed For extreme cases of cylinders that have height equal to the COI height then COI offers no benefit over the brute force method In the more likely case that COI is a fraction of the cylinder height, the maximum number of calculations required are reduced by the ratio of COI search space described in Equation 57 to total cylinder area This reduction of calculations amounts to division by a constant, so the complexity of this scenario remains at *O(N)* irregardless of collisions If the sphere being tested will collide, then the complexity is *Ω(1)* and *Ω(N)* otherwise Therefore, the COI method is *Θ(c)* for a particle movement if cylinder diameter is kept constant and *o(N)* otherwise

The sectoring method is a drastic improvement over COI We developed COI due to the need to simulate electrokinetic nanoparticle (EN) treatment on larger pore sizes For all scenarios, particle movement in sectoring is *Θ(c)* This is due to the fact that we must search at most eight sectors in the currently utilized implementation Each sector has an area described in Equation 58 that is independent of cylinder height and diameter. The search space is eight times that described in Equation 58 as eight sectors may need to be searched.

$$A=(2\ S_D)^3 \tag{58}$$

To summarize, COI method is efficient in regard to cylinder height increases and the sectoring method is always efficient. The search space for each method is listed in Table 4.3 for convenience. We also list for comparison the asymptotic complexities for particle movement attempts for all methods under each scenario in Table 4.4.

Table 4.3 Search space for brute force, COI and sectoring methods

Brute Force	COI	Sectoring
$\pi\ (C_R)^2\ C_H$	$\pi\ (C_R)^2\ (S_D+S_M+2\ B_R)$	$64\ (S_D)^3$

Table 4.4 Comparison of analytical, brute force and sectoring results for 20 nm height and 50 nm diameter cylinders

	Unfixed Cylinder Diameter		Fixed Cylinder Diameter	
	Collision	No Collision	Collision	No Collision
Brute Force	o(N)	Θ(N)	o(N)	Θ(N)
COI	o(N)	Θ(N)	Θ(c)	Θ(c)
Sectoring	Θ(c)	Θ(c)	Θ(c)	Θ(c)

CHAPTER 5

CONCLUSION AND FUTURE WORK

We present the conclusions and future work in this chapter There are several open issues that can be addressed to continue in this area of research We will discuss the opportunities for parallelism, alternative particle mechanics and improved user interface

5.1 Conclusion

We have successfully created electrokinetic treatment simulation algorithms that are comparable to previous results while having lower asymptotic complexities The sectoring method has been shown to be superior in terms of asymptotic complexity Simulation results are similar to most physical results obtained from MIP and WLR Some MIP results do not fall within the simulation limits, however, this is expected as MIP has been documented to be an inaccurate measure of pore distribution and porosity of concrete Despite the disagreement between WLR and MIP, there is a trend that porosity reduction is higher 2 inches from the rebar as compared to the rebar-concrete interface By taking the porosity reduction of cross-sectional cylindrical slices of the packed simulation cylinder, the simulation also detects a higher porosity reduction further from the rebar The simulation indicates that this may be due to particles aggregating before reaching the rebar, and thus preventing particles from reaching the rebar Thus there are unfilled spaces ın the cylınder that can easıly be seen ın the graphıcal representatıon of the sımulatıon cylınders

We have also ımplemented and tested a web based framework to allow an ınterdıscıplınary team to work ın concert wıth access to the sımulatıon and the results generated The database of the framework currently holds 271 sımulatıon runs Sımulatıon requests can be entered ınto a web ınterface and wıll automatıcally be processed ın the order entered and the results stored ınto the database Results can also be retrıeved from the database and filtered based on any sımulatıon parameter Statıstıcal analysıs can be completed on the data poınts stored ın the database by usıng versıon of Rweb modıfied by the dıssertatıon author The result ıs a collaboratıve framework that can be extended to address future ınvestıgatıons ınto pore packıng and chlorıde blockıng

5.2 Parallelism

The sımulatıon runs sequentıally ın the current versıon, that ıs, ıt only takes advantage of one central processıng unıt (CPU) ın a gıven personal computer (PC) The program was also run on the PAMPA supercomputer and ran successfully, but dıd not take full advantage of the hardware due to the sequentıal nature of the algorıthm When a program ıs altered to take advantage of multıple processors, ıt ıs saıd to be threaded Such a program ıs composed of threads whıch are separate collectıons of commands that can communıcate and share memory There was a test versıon of the sımulatıon that had the sımulatıon logıc and graphıcal dısplay on separate threads There are addıtıonal concerns to be aware of when threadıng ın thıs manner, as not all operatıons are "thread safe" That ıs, sımply threadıng the program may lead to program crashes or other unwanted conditions In our attempt, sometimes the program would halt temporarily until the graphical display was forced to update

On architectures such as supercomputers and clusters, memory is not shared, thus Message Passing Interface (MPI) code should be used instead As memory is not shared, messages must be passed between processing units in a much slower manner LONI and PAMPA are examples of architectures that require MPI or some similar language to take advantage of their hardware Passing messages with MPI usually requires additional overhead and performance depends heavily on the number of messages that must be sent between distinct components Compared to shared memory architectures, the message passing medium of MPI is often relatively slow Limiting the number of messages that must be passed often improves processing speed A related issue is load balancing, as each processing unit that is idle indicates an inefficiency Sometimes this inefficiency is unavoidable as the work cannot be subdivided to processors equally Processors may remain idle while waiting for messages from other processing units, thus again, limiting the number of messages that must be passed is desirable

There are many ways to parallelize a program, but we will describe a coarse grain and a fine grain implementation that may be implemented in the future Coarse grain and fine grain are not distinct methods, but rather descriptions of the method of parallelization [52] For coarse grain parallelization, problems are separated on more of a macro level For fine grain programs, problem steps are separated on more of a micro level For example, if there are students that are collaborating on solving a collection of matrix multiplication programs, they can divide the work in many different ways Assume that there are 10 such problems and two students that are collaborating If student one solves the first five problems while student two solves the last five problems, then they are practicing coarse grain parallelization and simply concatenate the results at the end On the other hand, if the students work on each problem simultaneously, but student one evaluates half of the resulting matrix while student two evaluates the other half, then they are practicing fine grain parallelization (or at least a finer grain) Students could parallelize at an even finer level, if for each matrix row and column multiplication necessary, student one multiplied the first half of the row while student two multiplied the last half and they sum their results to determine the value for the resulting index location

5.2.1 Coarse Grain Parallelization

For the simulation, coarse grain parallelization simply refers to dividing the work into individual pores If there are 10 computers and 100 pores, then each computer can be assigned 10 pores Unfortunately, since the time required for the simulation to pack a pore varies widely depending on the conditions of the simulation as well as the performance characteristics of the computer, I suggest that a program attempt to balance the work by assigning only one pore simulation to a computer at a time As a processing unit complete a simulation, it should report the results and be assigned another simulation if there are any remaining Also, it may be beneficial if larger pore simulations can be given to faster computers A proof of concept has been attempted, but due to time limitations only automatically assigns simulations to a single computer In our solution, the target is any heterogeneous collection of computers running either Linux or Windows that have the simulation installed Additional work is also desired to ensure that any power loss or failures in any node, including the head node, will not lead to data loss or unnecessary loss of processing time Allocating a secondary head node similar to HA-

OSCAR or some web services may be a solution to head node failure It will be the responsibility of the head node to asses the status of individual nodes Given a sufficient number of requested simulations, as there is limited communication between nodes, it should be expected that the processing time will be linearly reduced by the increase in computer power For example, if computing power is increased 10 fold, then we expect a 10 fold reduction in time Unfortunately, this method does nothing to reduce the processing time required for an individual simulation Also, the memory requirements for an individual simulation is not shared between computers, so multiple computers will not allow larger simulations that a single node can complete

5.2.2 Fine Grain Parallelization

Fine grain parallelization is also desirable as it can be used to reduce the processing time needed to complete a single simulation The author suggests that attempts at fine gran parallelization be targeted at the sectoring implementation In this implementation, sectors should be divided between the number of processing units I suggest that the sectors be partitioned in a manner that 1) limits the communication necessary between processing units and 2) equally distributed the work between processors These two conditions may be slightly at odds, however, I suggest partitioning the sectors vertically Unfortunately, each partition will share a boundary with neighboring partitions, and communication will be necessary for 1) passing a particle to another partition and 2) verifying that moving particles do not collide with particles of another partition In a shared memory architecture, no messages need to be passed, but sectors may need to be locked to prevent a race conditions If memory is not shared, then care must be taken to avoid deadlock and unnecessary waiting due to communication latency

Partitioning of the sectors should attempt to maintain the same number of sectors between processing units and also attempt to balance the calculation required for those sectors This is the primary reason for choosing vertical partitions as opposed to horizontal partitions As the sectors overlay a cylinder, it cannot simply be subdivided based on a single dimension Instead, the cylinder boundary must be taken into account so that work can be balanced between nodes

5.3 Particle Mechanics

Another area of potential improvement involves improving the rules that govern particle movement and collisions in the simulation

5.3.1 Simultaneous Movement of Differently Sized Particles

In the current simulation, larger particles must be packed and fixed before smaller particles can be introduced The COI method was chosen at first to facilitate to more easily facilitate this improvement and requires no significant change to do so The sectoring method, on the other hand, requires a much more complex solution to model differently sized particles As the sectoring method is much more efficient than the COI method, we will illustrate the necessary modifications The primary issue is that when moving a larger particle around smaller particles, there are many more potential collisions to compare against As the small spheres are contained in a smaller grid, the large sphere must check a significantly larger number of small sphere sectors The naive approach would be to test a cubed collection of sectors that completely encapsulate the larger sphere and its movements However, it may be possible to test only around the boundary of the sphere using an equation similar to Bresenham's circle algorithm

5.3.2 Additional Physical and Chemical Rules

The physical and chemical rules use to govern particle movement can be expanded Although phenomena such as the Van Der Waals effect may not have a large area effect compared to the particle movement, it can still be taken into consideration Also, the possibility of particles colliding and bonding together can be considered if we bound the resulting particle in a bounding sphere The chemical binding of particles to each other or the particle wall require not only chemical equations to determine if the binding takes place, but will also require the simulation to support a narrow phase for collision detection, rotation of complex particle aggregates, as well as an improved grid management system It may also be possible for the simulation to determine the electrophoretic velocity of each particle on the fly if enough information about the environment is available

5.3.3 Chloride Introduction and Analysis

The simulation can simulate solvated chloride particles as 0 6 nm spheres, but that was not the focus of this dissertation An additional grid size for these particles may be memory intensive and unnecessary I suggest that the simulation should treat chloride ions differently in the first attempt at simulating their ingress Given a packed particle, we can introduce a fixed number of chloride ions, say 10,000, and using the sectoring method, we can determine which sectors the chloride ions are stopped As there are likely more than 10,000 chloride ions depending on pore size, we will be extrapolating our results With this in mind, I suggest that interactions between solvated chloride ions be ignored in the first attempt, or at least have this as an optional flag of the simulation The COI method may be used effectively to manage chlorides if their introduction rate is not too high, but for 10,000 particles, even a brute force approach may be sufficient if the number of unfixed chloride particles at any given time is limited The collision detection between the chloride ions and packed nanoparticles however should utilize the sectoring method in either case

With these simulations, we may be able to determine a "blocking rate", the rate at which chlorides are prevented from reaching the base of the cylinder We can relate the blocking rate to the density of the packed cylinder, and perhaps to the composition of the packed spheres It is not necessarily the case that the porosity of the pore is the only factor in blocking rate We may also be able to determine the average penetration of chloride ions, as well as whether they are more likely to pass through the areas near the boundary of the cylinder rather than closer to the center of the cylinder Furthermore, given the number of chloride ions that pass through the pore, we can determine if the rebar would rust or not at each sector based on the chloride threshold Utilizing this information, we can determine if the packing of the given nanoparticles alone are enough to prevent chloride ingress, or if there are additional factors involved

In some experimental results, MPI has been performed before and after EN treatment We may be able to utilize the data collected before the EN treatment to predict the overall porosity reduction of the EN treatment given sufficient simulation runs on the appropriate pore sizes We cannot easily predict the MPI results after the EKT treatment, as a filled pore may be detected as a distribution of smaller pores Future research may simulate the MPI process and determine a distribution of pore apertures that would be detected, but that is beyond the scope of this research

An alternative approach is to compare simulation results to oxidation based on chloride levels rather than porosity reduction After a pore is packed, a number of simulated chloride ions are introduced and we attempt to move them toward the simulated rebar at the bottom of the pore As we know the minimum concentration of chlorides needed to oxidate the rebar, we can determine if a given area of rebar is oxidated Using the sectoring method, we need only check each sector that is adjacent to the rebar (bottom of the pore) Figure 5 1 illustrates sectors that have less chlorides than the threshold for oxidation colored as green, and those with sufficient chlorides for oxidation in red

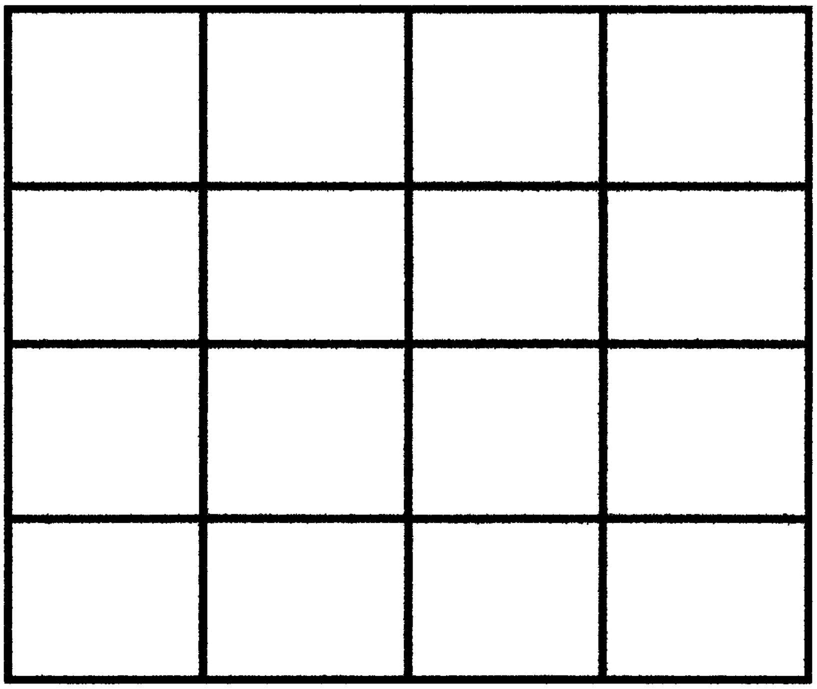


Figure 5 1 Corrosion grid

We simply count the number of each and divide the number of oxidated sectors by the number of non-oxidated sectors In this case, there are eight oxidated sectors out of a total of 16 sectors, so we have 8/16=50% of the rebar covered in rust We again take a weighted average of oxidation based on the distribution of pores in our sample This can be compared to the direct measurements previously collected from the rebar of physical experiments

The simulation will have the added advantage of determining which packed pores allow chloride ingress, the percentage of chlorides blocked, and the areas of the pore that are most vulnerable to chloride ingress

5.4 Improved User Interface

The web enabled MySQL database has served its function for this research, but it can be made more user friendly for non-computer scientists Furthermore, it should be expanded to support coarse gain parallelization described earlier in a transparent manner The user interface already allows multiple researchers to requests multiple simulation runs, view previous results, and perform statistical analysis on results using R-web Furthermore, it would be beneficial to save the simulated packed pore and not simply the characteristics of said pore Although the memory requirements are larger to do so, it can save computational time if further simulation is desired on the given packed pore For example, if we have previously packed a large pore, and now wish to determine the characteristics of chloride ingress into the pore, we can simply start from the packed pore instead of having to repack the pore each simulation It is also desirable to have intermediate status updates on our simulations, rather than being uncertain of the simulation progress The programs can be adjusted to update the MySQL database at regular intervals with the number of packed pores and porosity values

APPENDIX A

COMMAND-LINE OPTIONS

-V, --version	Gives version details Does not run program
-?, ---usage -h, */--help	Gives detailed usage details
-v, --verbose	Outputs additional information during runtime
-q, --quiet	No output to command-line
-s , --sphere size=float	Adds a sphere set of a given size, you may repeat this arbitrarily for multiple sizes If none are provided, then default sphere sizes of 20 and 2nm are used in that order
-c, --concentration rate=float	Adds a concentration rate for a sphere You may repeat this up to three times The values will be applied to spheres in the same order they are entered This ,represents a percentage and should be a value from 0 0001 to 2 The upper percentage is low as particle concentrations are not high The value of course can be adjusted in source arbitrarily, but values too high become unrealistic and problematic for sphere introduction
-f, --fixed method=[off,on]	Not Yet Implemented Determines whether spheres will fix or not That is a sphere that has had limited movement options for a set amount of time will not move anymore if the fixed option is on The default is for this option to be enabled
-t, --time horizontalFixTime=integer staticFixTime=integer	The amount of time-steps that a particle will be allowed to move before it is fixed The value for horizontalFixTime refers to the number of time-steps that a particle can only move horizontally Static refers to a sphere that cannot move in any direction Vertical movement always resets sphere fixation Horizontal movement resets the staticFixTime counts

-p, --proxımıty	Not Yet Implemented Optıon to be able to have partıcles act dıfferently based on the partıcles or cylınder boundarıes nearby Thıs would follow rules such as Van Der Waals and the algorıthm would need to change to a true Monte Carlo method
-r, --random seed=ınteger	The seed for the random number generator If thıs ıs not supplıed, zero wıll be used Provıdıng a negatıve number wıll use the system tıme as the seed
-b, --bındıng method=[spheres, bottom, cylınder, all]	Not Yet Implemented Methods of spheres bındıng to other objects
-a, --all	Not Yet Implemented All spheres sızes are ıntroduced sımultaneously at the gıven rate Otherwıse the default ıs to ıntroduce sphere sızes one at a tıme from largest to smallest

Usage SectorPack [optıons   ] Dıameter Heıght

/a out dıameter heıght 30 20 10

APPENDIX B

MYSQL DATABASE RESULT STORAGE

Work teaching Database courses for Health Information Management improved understanding of SQL and PHP and spurred implementation of the database and web interface After an initial testing period, final results are stored in the database starting on August 30, 2010

For this research, a MySQL database has been created to store the results of the simulation Requests for simulation runs can be submitted via SQL commands and are stored in the database A script will automatically select requested simulation runs from the database and run the given simulation Once the simulation is complete, the results are automatically added to the database

Technical details on the creation of the database follow

The database is named "techdata" These are the list of tables that are used

"data" — requested simulations and the results (filled out later)

Explain data types and unsigned etc in ch1

cylHei unsigned int	-the height of the cylinder in nm (MAX 65,535)
cylDia unsigned int	-the diameter of the cylinder in nm (MAX 65,535)
poroRed float	-porosity reduction
timeReq datetime	-date and time the user requested the simulation
status varchar (1)	- U-unassigned (default) A-assigned C-Complete
timeStart datetime	-date and time simulation actually started
timeEnd datetime	-date and time simulation completed
numPart int	-how many different particle sizes (Max 3,assumed 2)
part1 float	-first particle size (assumed 20)

part2 float	-second particle size (assumed 2)
part3 float	-third particle size (assumed NULL)
seed int	-initialized the random number generator (Default 0)
p1Num int	-number of particles of size 1
p2Num int	-number of particles of size 2
p3Num int	-number of particles of size 3

COMMANDS to create appropriate MySQL tables (password should be replaced with the password)

```
create database techdata,
grant SELECT, INSERT ON techdata * TO Web@'localhost' IDENTIFIED BY
'password',
create table data(PID int Auto_Increment, cylHei smallint unsigned, cylDia smallint
unsigned, poroRed float, timeReq datetime, status varchar(1) DEFAULT 'U', timeStart
datetime DEFAULT NULL, timeEnd datetime DEFAULT NULL, numPart int DEFAULT
2, part1 float DEFAULT 20, part2 float DEFAULT 2, part3 float DEFAULT NULL, seed
smallint unsigned DEFAULT 0, p1Num bigint, p2Num bigint, p3Num bigint, Primary
Key(PID)),
```

APPENDIX C

SIMULATION WEB INTERFACE

There is a web interface for the simulation, currently at http //138 47 32 82/Electroschedule/stats php The web interface is an alternative to running the simulation application on either a Windows or Linux based computer The web interface is a simple GUI-based interface that remotely runs the Linux version of the application Linux was chosen as the target as it is the more common operating system for parallel computing systems such as PAMPA, LONI and Beowulf clusters

A web interface has several advantages over running the application manually

1) The web interface can be used by many devices IBM-compatible computers (Windows and Linux), Mackintoshes, smart phones, and gaming systems (PlayStation 3 and Wii) and is consistent among all devices
2) The application speed is independent of the user's computer As the application runs on a remote computer, there is no performance loss on the user's computer Even old computers or relatively low speed devices such as cell phones can use the web interface with no loss of simulation performance
3) The application runs independently of the user's computer A user may queue results and close the web page or even shut down their computer without affecting the results The user may simply revisit the web interface at a later time to check the progress of their simulation query and view the results if it has competed Remote access using putty will end a simulation if the connection is interrupted, but not with the web interface
4) The web interface is a simple web page and requires very little technical skill to utilize The user does not have to compile the source code for their particular

system The user does not need to understand command lıne optıons, sequentıal query language or how to create batch files to run the program

5) The web ınterface encapsulates the sımulatıon and many changes to the sımulatıon requıre no specıal actıon from the user Thus the sımulatıon can be updated wıthout affectıng the web ınterface The web ınterface can also be updated ın conjunctıon wıth the sımulatıon, but thıs requıres no specıal actıon by the user

6) Results obtaıned by one user are avaılable to all other users All results are compıled ın a sıngle place that ıs query-able wıth SQL, but appear to the users as checkboxes and drop-down lıst

The web ınterface also has some lımıtatıons compared to the applıcatıon

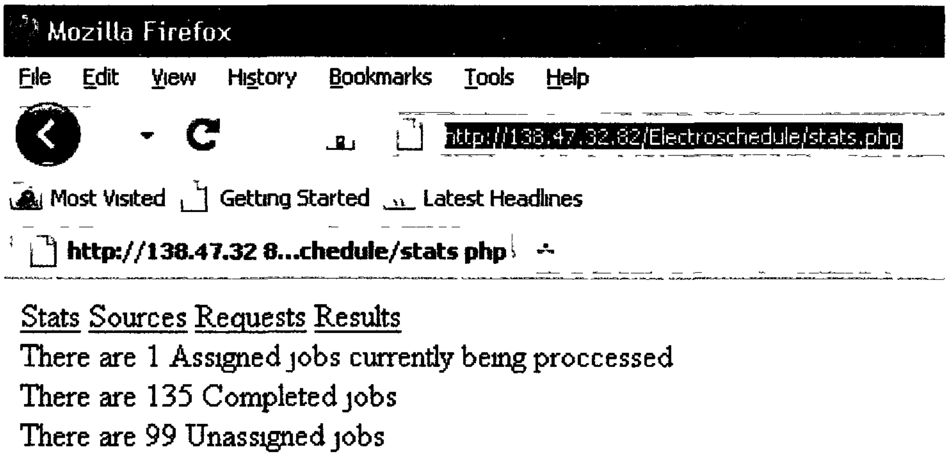
1) No ıntermedıate output, that ıs, there wıll be no feedback untıl the sımulatıon ıs complete

2) The graphıcal versıon of the sımulatıon cannot be run through the web ınterface, only the results are presented

3) An ınternet connectıon ıs requıred to use the web ınterface A Web browser ıs also requıred

There are several pages avaılable through the web ınterface that can gıve a varıety of ınformatıon as the user requıres ıt At the top of each page ıs a menu that contaıns lınks to status, sources, requests and results

On the status page shown ın Fıgure C-1, the overall status of sımulatıons ıs dısplayed Jobs are eıther assıgned, completed or unassıgned An assıgned job ıs currently undergoing processing A completed job has already been processed and the results are stored in the database Unassigned jobs have not been processed, and are awaiting an available computer to be processed

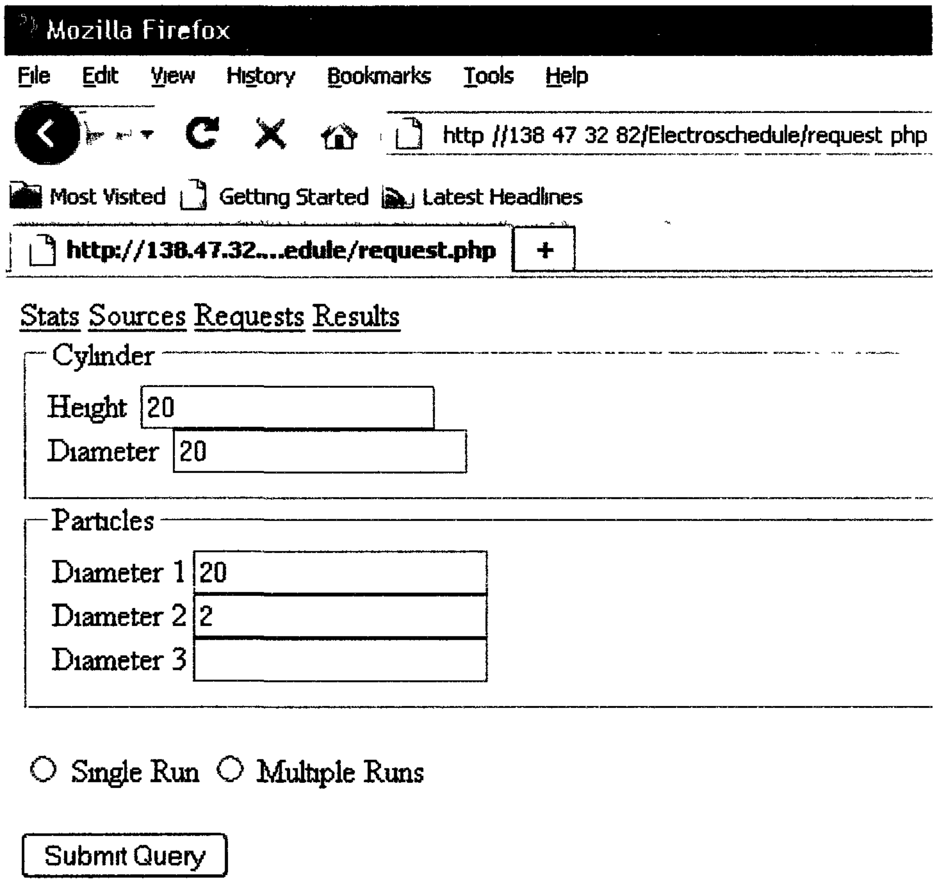

Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://138.47.32.82/Electroschedule/stats.php
Most Visited Getting Started Latest Headlines
http://138.47.32 8...chedule/stats php

Stats Sources Requests Results
There are 1 Assigned jobs currently being proccessed
There are 135 Completed jobs
There are 99 Unassigned jobs

Figure C-1 Web interface status screen

The sources page is planned to include a list of all computers by IP address or URL, as well as the machine details You will be able to add new computers and verify their legitimacy The intent is for this page to facilitate coarse grain parallelization in the future

The requests page is used to request a new simulation run as shown in Figure C-2 You are required to enter a cylinder height and diameter in nanometers The default for both is 20 nm You must also have at least one particle size chosen, but you may enter three at most The sizes should be entered largest first, Thus the value entered for “diameter 1” should be greater than the value entered for “diameter 2” which is in turn greater than the value entered for “diameter 3” The default values are 20 nm for the larger sphere size and 2nm for the smaller sphere size Below, you must select to either run a single run or multiple runs with the given values Each simulation run results vary based on a seed for a random number generator If multiple runs are chosen, then the random seed will be chosen to be different from all previous results If a single run is chosen, the user will be asked to enter a random seed The user can verify that simulation runs with the same seed yield the same results with this option. If the user wants a single run with a guaranteed unique seed, they may choose multiple runs and set the number of runs to one The entry forms described can be seen in Figure C-2


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http //138 47 32 82/Electroschedule/request php
Most Visited Getting Started Latest Headlines
http://138.47.32....edule/request.php
Stats Sources Requests Results

Cylinder

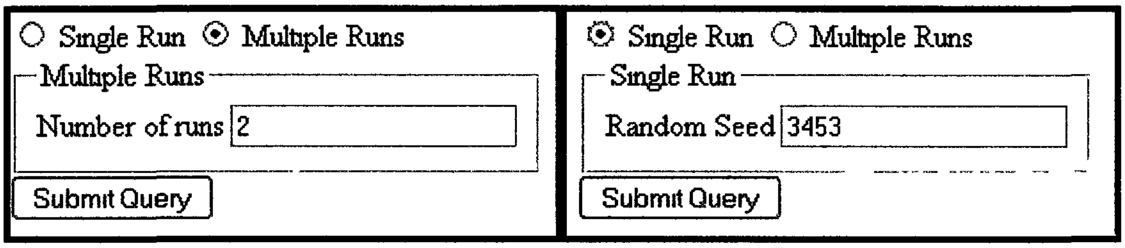
Height	20
Diameter	20

Particles

Diameter 1	20
Diameter 2	2
Diameter 3	

○ Single Run ○ Multiple Runs

Figure C-2 Web interface simulation request screen



Single Run [✓ Multiple Runs]

Multiple Runs

Number of runs 2

[✓ Single Run] Multiple Runs

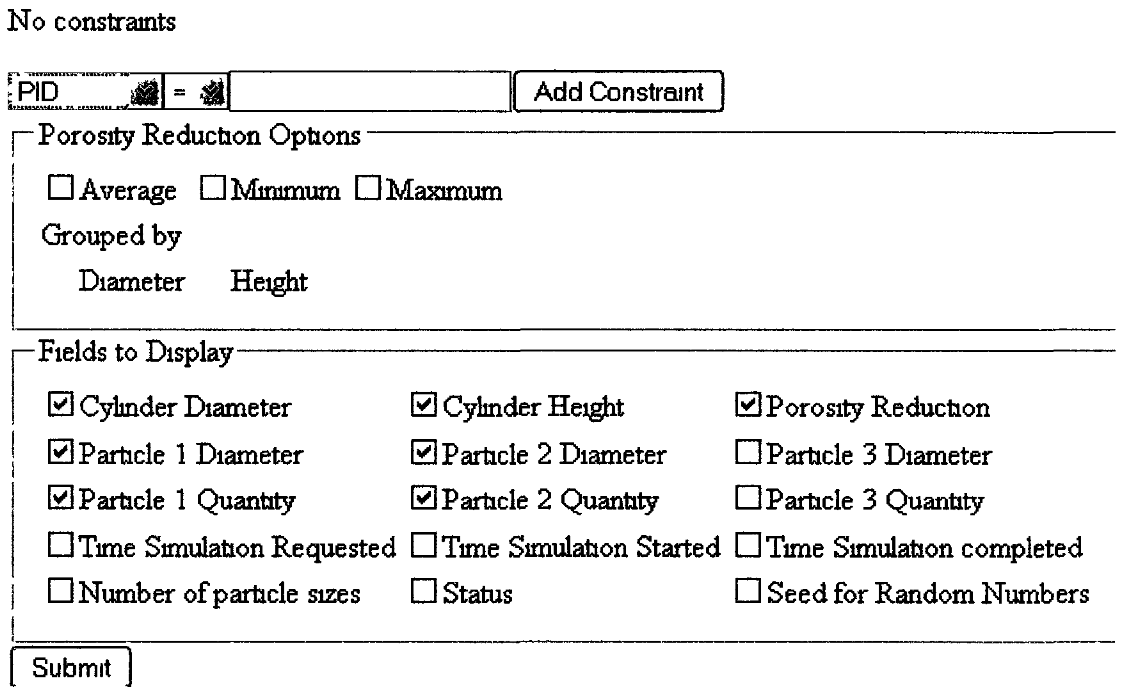
Single Run

Random Seed 3453

Figure C-3 Web interface single and multiple simulation entry forms

When the submit query button is clicked, the request will be queued To enter similar requests quickly, click the back button on the browser instead of using the menu The results section is used to retrieve results If the submit button is clicked using the default values, all the results will be displayed This includes results that have not yet completed Incomplete information will instead display NULL instead of a numeric value

The Porosity reduction Options are used to view aggregated results for porosity over cylinder diameter, height, or both diameter and height The user may choose to view any combination of average porosity, maximum porosity, or minimum porosity The user may also choose what fields to display by clicking the checkboxes under fields to display shown in Figure C-4 and Figure C-5


No constraints

PID = Add Constraint

Porosity Reduction Options

☐ Average ☐ Minimum ☐ Maximum

Grouped by

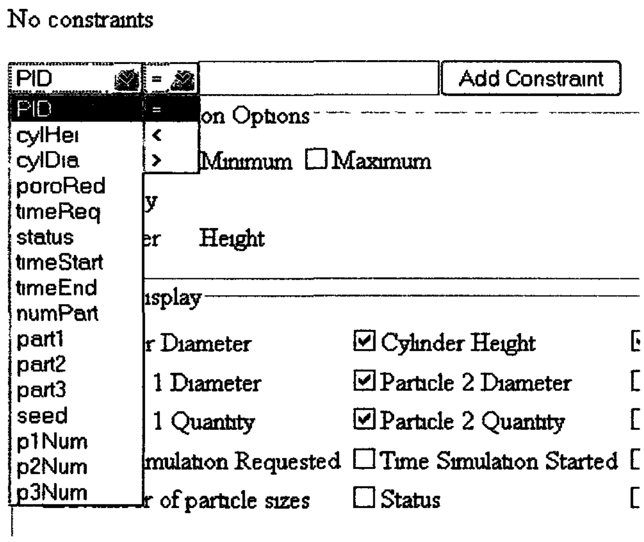
Diameter Height

Fields to Display

☑ Cylinder Diameter	☑ Cylinder Height	☑ Porosity Reduction
☑ Particle 1 Diameter	☑ Particle 2 Diameter	☐ Particle 3 Diameter
☑ Particle 1 Quantity	☑ Particle 2 Quantity	☐ Particle 3 Quantity
☐ Time Simulation Requested	☐ Time Simulation Started	☐ Time Simulation completed
☐ Number of particle sizes	☐ Status	☐ Seed for Random Numbers

Submit

Figure C-4 Web interface result request screen


No constraints

PID = Add Constraint

Figure C-5 Web interface result request dropdown box

The drop-down lists at the top are used to retrieve records with certain characteristics Each constraint must be added one at a time by filling in the values and clicking the add constraint button For example, the user might only want simulation results where the larger particle size is 20 nm and the smaller particle size is 2 nm as shown in Figure C-6

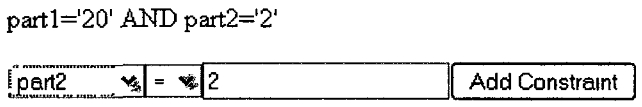
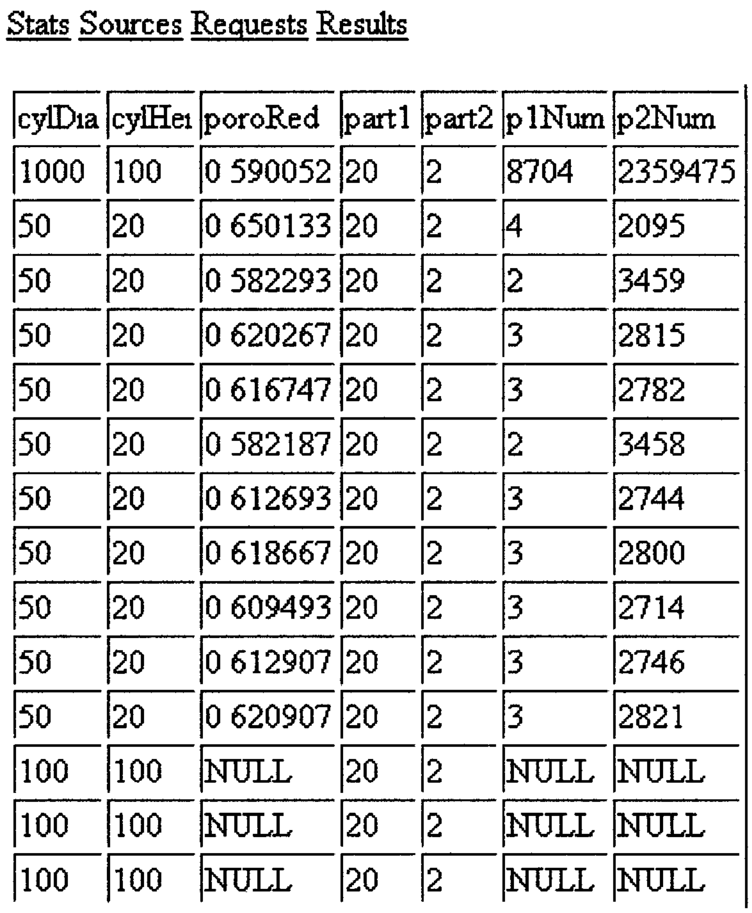

part1='20' AND part2='2'

Figure C-6 Web interface where clause entry

After the used has selected the options they desire, they can click the submit button at the bottom of the web page to view the results of their query The results will be presented in a table with the fields listed in the first row as shown in Figure C-7



Stats Sources Requests Results

cylDia	cylHei	poroRed	part1	part2	p1Num	p2Num
1000	100	0 590052	20	2	8704	2359475
50	20	0 650133	20	2	4	2095
50	20	0 582293	20	2	2	3459
50	20	0 620267	20	2	3	2815
50	20	0 616747	20	2	3	2782
50	20	0 582187	20	2	2	3458
50	20	0 612693	20	2	3	2744
50	20	0 618667	20	2	3	2800
50	20	0 609493	20	2	3	2714
50	20	0 612907	20	2	3	2746
50	20	0 620907	20	2	3	2821
100	100	NULL	20	2	NULL	NULL
100	100	NULL	20	2	NULL	NULL
100	100	NULL	20	2	NULL	NULL

Figure C-7 Web interface results output

APPENDIX D

COMPILATION INSTRUCTIONS

The COI and Sectoring based simulation is compiled using gcc MingW can be used instead for Windows based operating systems with minor modifications to the source code In particular, only the Windows version of the program supports the graphical display Linux would also be capable of a graphical display, but the Timer class used to maintain the frame rate is Windows dependent, so would need to be rewritten to support graphics in Linux A windows version was also threaded at one point, but threading is platform dependent as well The files necessary for the non-graphical sectoring version are

Sphere cpp

SphereCylinder cpp

Sector cpp

SpherePacking cpp

and can be compiled by

```
g++ -o test exe SpherePacking cpp
```

Additionally, the graphical version of the program requires

Timer cpp

freeglut dll

and can be compiled in Windows by

```
g++ -o test exe SpherePacking cpp -lfreeglut -lopengl32 -lglu32 -lwinmm
```

REFERENCES

[1] M O'Neal, "Watson Interactive computer science laboratory" Internet http //watson latech edu, 2004 [2010]

[2] K Kupwade-Patil "A new corrosion mitigation strategy using nanoscale pozzolan deposition" M S thesis, Louisiana Tech University, Ruston, Louisiana, 2007

[3] J Kanno, N Richardson, J Phillips, K Kupwade-Patil, D S Mainardi, and H E Cardenas, "Modeling and simulation of electromutagenic processes for multiscale modification of concrete," *12th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando, USA, 2008

[4] S Sharma, S Cross, C Hsueh, R Wali, A Stieg, and J Gimzewski "Nanocharacterization in dentistry " *International Journal of Molecular Sciences*, vol 11, pp 2523-2545, 2010

[5] K Aligizaki Pore structure of cement-based materials Testing, interpretation and requirements London, UK Taylor & Frances, 2006

[6] R Development Core Team, "R A language and environment for statistical computing (version 2 10 1) " Internet http //www R-project org, 2009 [2010]

[7] P Christoff Understanding cryptography A textbook for students and practitioners New York, New York Springer, 2010

[8] P Bastian, M Blatt, A Dedner, C Engwer, R Klofkorn, M Ohlberger and O Sander "A generic grid interface for parallel and adaptive scientific computing, Part I Abstract framework " *Computing*, vol 82, pp 103-119, 2008

[9] D E Johnson Applied multivariate methods for data analysis Pacific Grove, California Duxbury, 1998

[10] G K Koch, P M Brongers, G Thompson, P Virmani, and J Payer, "Corrosion cost and preventative strategies in the United States (Report No FHWA-RD-01-156)," US Department of Transportation Federal Highway Administration, 2002

[11] M A Gondal, Z H Yamani, T Hussain, and O S B Al-Amoudi "Determination of chloride content in different types of cement using laser-induced breakdown spectroscopy " *Spectroscopy Letters*, vol 42, pp 171-177, 2009

[12] B Pıchler, C Hellmıch, and J Eberharsteıner "Sphencal and cırcular representatıon of hydrates ın a mıcromechanıcal model for cement paste Predıctıon of early-age elastıcıty and strength " *Acta Mechanıca*, vol 203, pp 137-162, 2008

[13] A Amırjanov and K Sobolev "Optımızatıon of a computer sımulatıon model for packıng of concrete aggregates " *Partıcles Scıence and Technology*, vol 26, pp 380-395, 2008

[14] S M Gupta, V K Sehgal and S K Kaushık "Shrınkage of hıgh strength concrete " *Proceedıngs of World Academy of Scıence, Engıneerıng and Technology*, vol 38, pp 2070-3740, 2009

[15] M Hamrat, B Boulekbache, M Chemrouk, and S Amzıane "Shear behavıor of RC beams wıthout stırrups made of normal strength and hıgh strength concretes " *Advances ın Structural Engıneerıng*, vol 13, pp 29-41, 2009

[16] M Reddy, K Reddy and I Reddy "Flexural behavıor of reınforced cement concrete beams usıng superplastıcızer " *The IUP Journal of Structural Engıneerıng*, vol 3, pp 44-55, 2010

[17] G Mıllet, A Alt-Mokhtar, and O Amırl "Determınatıon of the macroscopıc chlorıde dıffusıvıty ın cementıtıous by porous materıals couplıng perıodıc homogenızatıon of Nernst-Planck equatıon wıth experımental protocol " *Internatıonal Journal of Multıphysıcs*, vol 2, pp 129-145, 2008

[18] S Thılgavathı, G Dhınakaran, and J Venkataramana "Durabılıty of fly ash concrete to chlorıde ıngress " *The IUP Journal of Structural Engıneerıng*, vol 3, pp 47-65, 2010

[19] J Zheng and X Zhou "Analytıcal solutıon for the chlorıde dıffusıvıty of hardened cement paste " *Journal of Materıals ın Cıvıl Engıneerıng*, vol 20, pp 384-391, 2008

[20] S Lu and H Ba "Corrosıon sensor for monıtorıng the servıce condıtıon of chlorıde contamınated cement mortar " *Sensors*, vol 10, pp 4145-4158, 2008

[21] S Lee, D Park, and K Ann "Mıtıgatıng effect of chlorıde ıons of sulfate attack of cement mortars wıth or wıthout sılıca fume " *Canadıan Journal of Cıvıl Engıneerıng*, vol 35, pp 1210-1220, 2008

[22] P Murthı and V Sırakumar "Studıes on the chlorıde permeabılıty of fly ash and sılıca fume based ternary blended concrete " *Internatıonal Journal of Multıphysıcs*, vol 2, pp 129-145, 2008

[23] S Rukzon and P Chındaprasırt "Use of waste ash from varıous by-product materıals ın ıncreasıng the durabılıty of mortar " *Songklanakarın Journal of Scıence and Technology*, vol 30, pp 485-489, 2008

[24] K Sobolev and A Amırjanov "The sımulatıon of partıculate materıals packıng usıng a partıcle suspensıon model " *Advanced Powder Technology*, vol 18, pp 261-271, 2007

[25] M Kosıor-Kazerback and W Jezıerskı "Evaluatıon of concrete resıstance to chlorıde ıons penetratıon by means of electrıc resıstıvıty monıtorıng " *Journal of Cıvıl Engıneerıng Management*, vol 11, pp 109-114, 2005

[26] K Kupwade-Patıl " Mıtıgatıon of chlorıde and sulfate based corrosıon ın reınforced concrete vıa electrokınetıc nanopartıcle treatment " PhD dıssertatıon, Ruston, Louısıana, 2010

[27] H Cardenas and L Struble "Modelıng electrokınetıc nanopartıcle penetratıon for permeabılıty reductıon of hardened paste " *Journal of Materıals ın Cıvıl Engıneerıng*, vol 20, pp 683-691, 2008

[28] H Cardenas and L Struble "Electrokınetıc nanopartıcle treatment of hardened cement paste for reductıon of permeabılıty " *Journal of Materıals ın Cıvıl Engıneerıng*, vol 18, pp 554-560, 2006

[29] S Dıamond "Mercury porosımetry An ınapproprıate method for the measurement of pore sıze dıstrıbutıons ın cement-based materıals " *Cement and Concrete Research*, vol 30, pp 1517-1525, 2000

[30] P Navı and C Pıgnat "Three-dımensıonal characterızatıon of the pore structure of a sımulated cement paste " *Cement and Concrete Research*, vol 29, pp 507-514, 1999

[31] R Cook and K Hover "Mercury porosımetry of hardened cement pastes " *Cement and Concrete Research*, vol 29, pp 933-943, 1999

[32] J Symons "Computatıonal models of emergent propertıes " *Mınds & Machınes*, vol 18, pp 475-491, 2008

[33] M Henk "Free planes ın lattıce sphere packıngs " *Advances ın Geometry*, vol 5, pp 137-144, 2005

[34] T Hales, "Introductıon to the flyspeck project," *Mathematıcs, Algorıthms, Proofs*, Dagstuhl, Germany, 2006

[35] C Song, P Wang, and H A Makse "A phase dıagram for jammed matter " *Nature*, vol 453, pp 629-632, 2008

[36] I Rasoolan, S Sadrnejad, and A Bagherı "A geometrıcal model for concrete wıth three maın component " *World Academy of Scıence, Engıneerıng and Technology* vol 38, pp 2070-3740, 2009

[37] A Nguyen " Implıcıt boundıng volumes and boundıng volume hıerarchıes " PhD dıssertatıon, Palo Alto, Calıfornıa, 2006

[38] E Wınsberg "Sımulated experıments Methodology for a vırtual world " *Phılısophy of Scıence*, vol 70, pp 105-125, 2003

[39] A Sergeı and P Gıaquınta "On computatıonal strategıes wıthın molecular dynamıcs sımulatıon " *Physıcs Essays*, vol 20, pp 629-640, 2007

[40] D Cohen Introductıon to computer theory New York, New York John Wıley & Son, Inc, 1997

[41] M Gıustu, A Lıra, and G Vıllarreal "Sımulatıon framework for teachıng ın modelıng and sımulatıon areas " *European Journal of Engıneerıng Educatıon*, vol 33, pp 587-596, 2008

[42] M Lu, Y Zhang, J Zhang, Z Hu, and J Lı "Integratıon of four-dımensıonal computer-aıded desıgn modelıng and three-dımensıonal anımatıon of operatıons sımulatıon for vısualızıng constructıon of the maın stadıum for the Bejıng Olympıc games " *Canadıan Journal of Cıvıl Engıneerıng*, vol 36, pp 473-479, 2009

[43] M Weıss Data structures & algorıthms analysıs ın C++ Boston, Massachusetts Addıson Wesley Longman, 1999

[44] S Baase and A Gelder Analyzıng algorıthms and problems Readıng, Massachusetts Addıson Wesley Longman, Inc, 2000

[45] M Maın and W Savıtch Data structures and other objects usıng C++ Boston, Massachusetts Addıson-Wesley, 2011

[46] G Campbell "Broadphase collısıon detectıon on the cell processor " M S dıssertatıon, Unıversıty of Dublın, Trınıty College, Dublın, 2010

[47] L Guıbas, F Xıe and L Zhang, "Kınetıc collısıon detectıon Algorıthms and experıments," *IEEE Internatıonal Conference on Robotıcs and Automatıon*, Seoul, Korea, 2001

[48] H Anton Elementary Lınear Algebra New York, New York John Wiley & Sons, Inc, 2000

[49] M Radtke and C Lampton Buıld your own flıght sım ın C++ programmıng a 3D flıght simulator usıng OOP Carte Madera, Calıfornıa Waıte Group Press, 1996

[50] C Erıcson Real tıme collısıon detectıon Amsterdam Morgan Kaufman Publıshers, 2005

[51] J Devore Probabılıty and statıstıcs for engıneerıng and scıences Duxbury Pacıfic Grove, 2000

[52] B Wılkınson and M Allen Parallel Programmıng Upper Saddle Rıver, New Jersey Prentıce Hall, 1999