

**SOFTWARE TOOLS TO RESOLVE THE UNIQUE CHALLENGES OF
MOLECULAR MEGA-MODELS**

by

Juan C. Lucio-Vega

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Chemical Engineering

Spring 2019

© 2018 Juan C. Lucio-Vega
All Rights Reserved

**SOFTWARE TOOLS TO RESOLVE THE UNIQUE CHALLENGES OF
MOLECULAR MEGA-MODELS**

by

Juan C. Lucio-Vega

Approved: _____
Eric M. Furst, Ph.D.
Chair of the Department of Chemical and Biomolecular Engineering

Approved: _____
Levi T. Thompson, Ph.D.
Dean of the College of Engineering

Approved: _____
Douglas J. Doren, Ph.D.
Interim Vice Provost for Graduate and Professional Education

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Michael T. Klein, Sc.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Prasad S. Dhurjati, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Antony N. Beris, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Raul F. Lobo, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed:

Darin Campbell, Ph.D.
Member of dissertation committee

ACKNOWLEDGMENTS

I would like to begin by thanking my academic role model, one of the nicest, funniest, and smartest people I know, my primary advisor Michael T. Klein. Thank you for guiding me but also allowing me to grow and be creative. We always joke but DMB was your idea, you asked me if I could get rid of Cygwin dependence and I said let me see what I can do. That day marked the beginning of a software tool development that has progressed through my PhD. Thanks to you I was able to work on high performance clusters, use CPUs and GPUs for parallelization, travel the country just based on one simple idea. I am blessed to have you as an advisor. Words can't describe my gratitude.

I would also like to thank my secondary advisor Prasad Dhurjati who always gave me great life guidance and mentorship. I will never forget the trip we had to Dupont and I got yelled at by security when I attempted to move from the back seat to the front seat of your car or the juice bar conversation we had on Main Street on a rainy day.

I would also like to thank my committee members Dr. Antony Beris, Dr. Raul Lobo, and Dr. Darin Campbell. All of you have made an impact on my academic career, whether it is through course work or research. Dr. Antony Beris your numerical analysis class gave me the numerical tools necessary to develop my thesis. Dr. Raul Lobo your great kinetics lectures gave me the strong kinetics background I use daily. Dr. Darin Campbell I had three wonderful months at Aspentech under your

mentorship, you are a very kind and smart individual, who made a cold winter enjoyable.

None of this would have been possible without my rock, my biggest supporter, the love of my life, my beautiful wife, Angela. I love you baby, we did it! You have pushed me up when I have been down. You sat through my endless practice talks. You have unconditionally supported me through my long academic journey. I can't thank you enough for your patience and love. I can't wait for what the future holds for us. This year has a lot in store for us, we are moving to Chicago with Solie and we will be welcoming a new love, Baby Lucio in December 2018!

All my achievements and hard work and determination were instilled by two people, my life role models, my parents. My parents immigrated to the United States with nothing and through hard work and perseverance made it their home. My father endured extreme hardships, being forced to quit school after 6th grade to help provide for his family, and finally moving to the United States at the age of 14 because of the lack of work in Mexico. My mother is one of the strongest women I know. She helped provide for us even when it was hard to make ends meet. I can't imagine enduring the difficulties they faced, thanks to them I am able to achieve the unthinkable in my family, a PhD.

I would also like to thank my brothers Aaron and Abraham; I love you guys so much and you know I am always there for the two of you. I would also like to thank my aunts and uncles who are a supportive and loving group of people, who without I would have never formed into the person I am today. Gracias Tia Lupe, Tio Jaime, Tio Luis, Tia Yadira, Tia Paty, Tio Joel, Tia Chela y Tio Jose.

I would also like to thank my research group mates who include Triveni Billa, Craig Bennet, Zhen Hou, Marguerite Mahoney, Brian Moreno, Scott Horton, and Pratyush Agarwal. Together we made a group that felt like a family. Especially with being so far away from blood family having this great support group made graduate school enjoyable. Even though our research family has changed over the years we still keep in touch and I know this connection will continue through the years.

Finally, I would like thank the grad students that helped mold ideas during my thesis. Brian Moreno or BMM, my first senior grad student, the one who helped me get my initial grasp on the group software tools and grad student life. You were a great friend and group mate. Scott Horton, my second senior group mate, whom I had many discussions with about coding. You were certainly a driving force for me to achieve the things I have through computer programming. Lastly, I would like to thank the guy that I am senior grad student to, my gym buddy, my great friend and an awesome person Pratyush Agarwal. You are one of the smartest and kindest people I know, thanks for all the discussions we had about research, board games, and life in general.

TABLE OF CONTENTS

LIST OF TABLES	xiii
LIST OF FIGURES	xv
ABSTRACT	xx

Chapter

1	INTRODUCTION	1
1.1	Feedstock Conversions to Valuable Products	1
1.2	Advantages of Kinetic Models	2
1.3	Feedstock Concentration	4
1.4	Advantages of Molecular-level Models	5
1.5	Modeling Large Systems	8
1.6	Challenging Regions in Molecular-level Modeling	9
1.7	Research Objectives	13
1.8	Thesis Scope	13
2	KINETIC MODELING TOOLKIT	16
2.1	Kinetic Modeler's Toolkit (KMT) Software Ecosystem	16
2.2	Interactive Network Generator (INGen)	17
2.2.1	Computational Representation of Molecules	18
2.2.2	INGen Reaction Network Development	19
2.3	Property Database (PD)	21
2.4	Initial Condition Generator (ICG)	21
2.5	Dynamic Model Builder (DMB)	22
3	DYNAMIC MODEL BUILDER: A C++ REACTION ENGINEERING MODELING FRAMEWORK	23
3.1	Introduction	23
3.2	Installation	28
3.2.1	Non-Developer	28
3.2.2	Developer	29

3.3	Chemical Engineering Principles in the DMB framework.....	31
3.3.1	Mole Balances	32
3.3.2	Stoichiometry	34
3.3.3	Rate Laws	36
3.3.4	Linear Free Energy Relationship (LFER) Concepts	39
3.3.5	Thermodynamics	41
3.3.6	Energy Balance.....	44
3.3.7	Numerical Solutions to Ordinary Differential Equations (ODEs).....	46
3.4	Dynamic Model Builder Program	50
3.5	Program Workflow	54
3.5.1	Parsing and Organizing Model Data	55
3.5.2	Pre-calculations	58
3.5.3	Solving Ordinary Differential Equations.....	59
3.6	DMB Model Independent/Dependent Framework.....	67
3.7	Performance Differences from the Previous Framework	70
3.8	Multi-Phase Framework	74
3.9	Parallel Computing.....	76
3.10	Model Analysis.....	86
3.11	Directory and File Configurations.....	92
3.11.1	Model Format	92
3.11.2	Model Directories	94
3.11.2.1	Mode Directory	94
3.11.2.2	Chemistry Directory	98
3.11.2.3	Reactions Directory	103
3.11.2.4	Kinetic Parameters Directory	105
3.11.2.5	Datasets Directory	108
3.11.2.5.1	Initial Condition Directory	112
3.11.2.5.2	Reactor Information Directory	114
3.11.2.5.3	Observed Directory	114
3.11.2.6	Optimization Directory.....	117
3.11.2.6.1	ASA Tuning Switches File.....	118
3.11.2.6.2	ASA Options File	120
3.11.2.7	Properties Directory.....	125

	3.11.2.8 Solutions Directory.....	127
	3.12 Conclusion.....	129
4	APPROACHES AND SOFTWARE TOOLS TO AID THE DEVELOPMENT OF PATHWAYS BASED MOLECULAR-LEVEL KINETIC MODELING OF LIGNIN PYROLYSIS	130
	4.1 Abstract.....	130
	4.2 Introduction	130
	4.3 Lignin Molecular Description	132
	4.4 Computational Methods	136
	4.4.1 Reaction Network Construction	137
	4.4.1.1 Primary Reactions: Ether cleavage and Alcohol Dehydration Pathways.....	137
	4.4.1.2 Secondary Reactions: Decarbonylation, Enol-Keto Tautomerization, and Styrene Degradation Pathways	141
	4.4.1.3 Hydrogen Incorporation	141
	4.4.2 Reaction Network Merging Approach	142
	4.4.3 Initial Conditions and Reactor Conditions	145
	4.4.4 Model Equations and Kinetics.....	145
	4.4.5 Kinetic Modeling Environment.....	146
	4.5 Results and Discussion	147
	4.6 Summary and Conclusions	152
5	MOLECULAR-LEVEL SIMULATION OF THERMOGRAVIMETRIC ANALYSIS (TGA): A CELLULOSE PYROLYSIS EXAMPLE	154
	5.1 Abstract.....	154
	5.2 Introduction	155
	5.3 Computational Methods	157
	5.3.1 Cellulose Molecular Representation.....	157
	5.3.2 Reaction Network Construction	158
	5.3.2.1 Cellotriose Thermal Degradation Pathways.....	159
	5.3.2.2 Cellobiose and Cellobiosan Thermal Degradation Pathways.....	159
	5.3.2.3 Cellobiose and Cellobiosan Char Formation Pathways.....	162

5.3.2.4	Glucose Thermal Degradation Pathways	165
5.3.2.5	Small Molecule Char Formation Pathways	167
5.3.3	Model Equations and Kinetics.....	169
5.3.4	Thermogravimetric Analysis (TGA) Simulation Approach.....	170
5.3.5	Kinetic Modeling Environment.....	173
5.4	Results and Discussion	174
5.5	Summary and Conclusions	179
6	DEVELOPMENT OF PARALLEL KINETIC PARAMETER ESTIMATION STRATEGIES FOR COMPLEX MOLECULAR-LEVEL SIMULATIONS	180
6.1	Abstract.....	180
6.2	Introduction	180
6.3	Computational Methods	182
6.3.1	Reaction Network Construction	182
6.3.2	Molecule Property Database.....	189
6.3.3	Initial Condition Generation.....	190
6.3.4	Model Equations and Kinetics.....	190
6.3.5	Kinetic Modeling Environment and High Performance Computing	193
6.4	Results and Discussion	194
6.5	Summary and Conclusions	197
7	APPLICATION DEVELOPMENT	199
7.1	Creating One Framework	199
7.2	Dynamic Model Builder Editor GUI.....	200
7.3	Summary.....	204
8	KINETIC PARAMETER ESTIMATION METHODOLOGY	205
8.1	Motivation	205
8.2	Kinetic Parameter Estimation Methodologies.....	205
8.2.1	Step 1: Determine Initial Conditions and Bounds.....	206
8.2.2	Step 2: Determine Issues through Visualization	207
8.2.3	Summary.....	207
9	DISSERTATION SUMMARY AND CONCLUSIONS	208

9.1	Summary and Conclusions	208
9.2	Recommendations for Future Work	211
9.2.1	Continuation of the Lignin Pyrolysis Work	212
9.2.2	Addition of a Differential Algebraic Equation Solver	213
9.2.3	Parallelization of Kinetic Parameter Estimation	214
9.2.4	Large Model Explicit Ordinary Differential Equations.....	217
9.3	Closing Remarks	218
REFERENCES		220
Appendix		
A	ADJACENCY LIST, DMB OPTIONS, AND LIGNIN MOLECULES	225
A.1	Adjacency List Example.....	225
A.2	DMB Framework.....	225
A.3	Modified Lignin Structures	235

LIST OF TABLES

Table 1: DMB Framework Design and Requirements	29
Table 2: Stoichiometry Matrix for $aA + bB \rightarrow cC + dD$	35
Table 3: Stoichiometry Matrix including $cC + dD \rightarrow fF$	35
Table 4: Example Reaction Network	56
Table 5: Extracted Reaction Level Data.....	56
Table 6: DMB Model Directory Hierarchy	93
Table 7: Control File Option Information	94
Table 8: LHHW Directory File Hierarchy	99
Table 9: Hydrogen Dependence File Format	99
Table 10: Adsorption Constant File Example	101
Table 11: Adsorption Group Exponents (Adapted (Froment, Bischoff, & de Wilde, 2011))	102
Table 12: Reaction Site Mapping File Format Example	103
Table 13: Example Kinetic Parameters File.....	106
Table 14: Datasets Directory File Information.....	109
Table 15: Dataset Loading Example	110
Table 16: Product Fractionation Example.....	111
Table 17: Datasets Directory Hierarchy	112
Table 18: Initial Condition File Format Example	112
Table 19: Differentiation Length File.....	113

Table 20: Reactor Info File Format	114
Table 21: Observed Directory File Description	115
Table 22: ObsSpeciesFrac0 and MixPropObsFrac0 Inputs.....	116
Table 23: ASA Control Files	117
Table 24: LFER Tuning Switches	119
Table 25: Kads Tuning Switches File Format	119
Table 26: ASA Options and Default Values	120
Table 27: Example 1: Initial Value, Lower and Upper Bounds File.....	121
Table 28: Example 1: LFER Tuning Switches File.....	122
Table 29: Example 2: LFER Tuning Switches File.....	122
Table 30: Example 2: Initial Value, Lower, and Upper Bounds	122
Table 31: Example 3: Adsorption Constant Tuning Switches File	123
Table 32: Example 3: Kads Tuning Switches File Update.....	124
Table 33: Example 3: Initial Value, Lower, and Upper Bounds	124
Table 34: Current Solution Files	127
Table 35: Adapted Freudenbergs Reaction Networks	147
Table 36: Lignin Pyrolysis Rate Constant Values.....	150
Table 37: Cellulose Pyrolysis Kinetic Parameters (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014)).....	177
Table 38: List of DMB Defined Mixture Properties	225
Table 39: Species Properties in Property Database	233

LIST OF FIGURES

Figure 1: Large Molecule Representation (Freudenberg, 1959)	11
Figure 2: Kinetic Modeling Toolkit	17
Figure 3: Bond Electron Matrix Isohexane Example	19
Figure 4: Isohexane Type C Beta Scission Example (Includes Reactant Matrix, Reaction Matrix and Product Matrix).....	20
Figure 5: Dynamic Model Builder Model Building Process.....	25
Figure 6: Adapted Pillars Chemical Reaction Engineering (Fogler, 2006).....	31
Figure 7: Simplified Model Class.....	51
Figure 8: DMB Framework Design.....	53
Figure 9: Explicit ODE Function for system with 4 species with 2 reactions (Rxn1: Irreversible, Rxn2: Reversible)	60
Figure 10: Rate Law Portions.....	62
Figure 11: 2D Jagged Array	64
Figure 12: Jagged Array Stoichiometry Implementation with Paired Data	66
Figure 13: Implicit ODE Function	67
Figure 14: Explicitly Written Rate Laws Created by DMB Framework.....	69
Figure 15: Explicitly Written Mole Balances Created by the DMB Framework.....	69
Figure 16: DMB Multi-Phase Approach	75
Figure 17: Program Processing on the CPU.....	77
Figure 18: CPU-GPU Program Processing	79
Figure 19: 35 Years of Microprocessor Trend Data (Adapted from (Harrod, 2012))..	80

Figure 20: ASA Sequential Objective Function Calculation	81
Figure 21: Parallel ASA Objective Function Calculation	83
Figure 22: LU Decomposition Performance Comparison.....	85
Figure 23: Comprehension vs Scientific Rigor Paradox	86
Figure 24: Reaction Rate Law Flux 3D Map (Reaction 0-20).....	89
Figure 25: Reaction Rate Law Flux 3D Surface Map Initial (Reactions 20-44).....	91
Figure 26: Reaction Rate Law Flux 3D Surface Map Final (Reactions 20-44).....	91
Figure 27: Example DMB Reaction File.....	105
Figure 28: van't Hoff Plot Manipulation through Modifiers	108
Figure 29: DMB Mixture Property Definition Example	126
Figure 30: Example Simulation Profile Plot for a Large System.....	129
Figure 31: Coniferyl Alcohol Radicals.....	133
Figure 32: β -ether linkages typical bond energy [kJ/mol] atoms numbered for use, (Adapted) (Faravellia, Frassoldatia, Migliavacca, & Ranzia, 2010).....	134
Figure 33: Adapted Freudenberg Lignin Molecule (Freudenberg, 1959)	136
Figure 34: Klein β -O-4 Linkage Cleavage Pathway and Reaction Matrix	138
Figure 35: McDermott Ketone Forming β -O-4 Primary Cleavage Pathway and Reaction Matrix	139
Figure 36: McDermott β -O-4 Linkage Alcohol Dehydration Pathway and Reaction Matrix	139
Figure 37: McDermott β -O-4 Linkage Secondary Cleavage Pathway and Reaction Matrix	140
Figure 38: α -O-4 Ether Linkage Cleavage Pathway and Reaction Matrix.....	140
Figure 39: Styrene Degradation Pathways and Reaction Matrix	141
Figure 40: Dehydration Reaction Pathway and Reaction Matrix.....	142

Figure 41: INGen Network Merging Approach	143
Figure 42: Butane and Isobutane String Code Example	144
Figure 43: String Code Reaction Description	145
Figure 44: Modified Lignin Structure	148
Figure 45: Experiment vs Model Important Species Results	150
Figure 46: Lignin Simulation Approach.....	152
Figure 47: Diebold Global Cellulose Pyrolysis Model (Diebold, 1994).....	156
Figure 48: Cellotriose Molecule	158
Figure 49: Cellotriose Degradation Pathways: 1. Thermohydrolysis 2. Thermal degradation	159
Figure 50: Cellobiose and Cellobiosan Fragmentation Pathway to form Formic acid	160
Figure 51: Cellobiose and Cellobiosan Ring Contraction to form HMF precursor ...	161
Figure 52: Cellobiose and Cellobiosan Depolymerization Pathway	162
Figure 53: Cellobiose and Cellobiosan Ring Fragmentation Pathway to form Formic Acid	164
Figure 54: Char Precursor Formation from Cellobiose and Cellobiosan	165
Figure 55: Glucose Degradation Pathways (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014))	166
Figure 56: D-Glucose Degradation Pathways Part 1 (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014))	167
Figure 57: D-Glucose Pathways Part 2 (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014)).....	167
Figure 58: Toluene Self-Alkylation Char Pathway	169
Figure 59: Thermogravimetric Analysis (TGA) Simulation Approach	173
Figure 60: Thermogravimetric Analysis (TGA) Simulation Result.....	175

Figure 61: Vapor Fraction vs Temperature Result	176
Figure 62: Melt Dynamics as a Function of Temperature.....	179
Figure 63: Paraffin Isomerization and Reaction Matrix.....	184
Figure 64: Type A β -scission Pathway and Reaction Matrix.....	185
Figure 65: Type B β -scission Pathway and Reaction Matrix	185
Figure 66: Type C β -scission Pathway and Reaction Matrix	185
Figure 67: Saturation Pathway and 4H Saturation Reaction Matrix	186
Figure 68: Ring Isomerization Pathway and Reaction Matrix	187
Figure 69: Ring Opening Pathway and Reaction Matrix	187
Figure 70: Dealkylation Pathway Normal Paraffin Product.....	188
Figure 71: Dealkylation Pathway Isoparaffin Product	188
Figure 72: Side Chain Cracking Type C Pathway and Reaction Matrix.....	189
Figure 73: Side Chain Cracking Type B Pathway and Reaction Matrix.....	189
Figure 74: normal-Paraffin Model Prediction based on Carbon Number Distribution	195
Figure 75: Isoparaffin Model Prediction based on Carbon Number Distribution.....	195
Figure 76: Mono-Naphthenic Model Prediction based on Carbon Number Distribution	196
Figure 77: Mono-Aromatic Model Prediction based on Carbon Number Distribution	196
Figure 78: Simulation Time in the Objective Function as Function of Number of Datasets.....	197
Figure 79: DMB GUI Reactions.....	203
Figure 80: DMB GUI Model Information.....	204
Figure 81: Lignin Feed Tuning Approach.....	213

Figure 82: Parallel Kinetic Parameter Estimation	216
Figure 83: Explicit ODE File Separation	218
Figure 84: 2-methylhexane Adjacency List	225
Figure 85: Lignin dimer β -O-4 linkage	235
Figure 86: Lignin trimer, 2 β -O-4 linkage.....	235
Figure 87: Lignin trimer, 1 β -O-4 linkage & 1 α -O-4 linkage	235
Figure 88: Lignin tetramer, 2 β -O-4 linkage & 1 α -O-4 linkage.....	235
Figure 89: Lignin tetramer, 1 β -O-4 linkage & 1 α -O-4 linkage.....	236
Figure 90: Lignin pentalignol, 3 β -O-4 linkage & 1 α -O-4 linkage	236
Figure 91: Lignin Octalignol, 4 β -O-4 linkage & 3 α -O-4 linkage	237
Figure 92: Lignin Nonalignol, 5 β -O-4 linkage & 4 α -O-4 linkage	238

ABSTRACT

The yield optimization of high value molecular species is a highly pursued objective in the chemical engineering discipline. This optimization has been labeled as “molecular management” and its application to chemical processes is seen as way to increase technology profitability and viability (Speeches, 2018). In theory, molecular management is ideal but in practice it is difficult due the complexity entailed in real-world chemical systems. Actual chemical systems involve large numbers of molecules and or molecular structures impeding the application of molecular management. The structural size of the molecules reacting and the number of reactive sites on the molecules create a combinatorial problem. This problem is further exasperated as increases in molecular numbers are inevitable in chemical reactors. Large numbers of molecules and/or reactions abstract associations with kinetic and process parameters due to sheer numbers involved. Molecular-level modeling can help navigate through these barriers due its fundamental basis and molecular traceability. Molecular-level models can decipher and optimize chemical processes at molecular resolutions. As eluded, molecular-level models of real complex systems can quickly reach large sizes, sometimes reaching mega levels in terms of reactions and/or species. Model development at these scales becomes time consuming and cumbersome affecting all model design phases: building, solving, kinetic parameter estimation, and editing. Advances in computer science allow for the development of new modeling techniques and approaches that decrease the time spent in molecular-level model development of large systems. The Dynamic Model Builder (DMB) is a C++ object oriented modeling

framework that accounts for varying model sizes while functioning independently of program compilation. The DMB framework creates, stores, and simulates molecular models from computer system memory. The DMB framework is further enhanced to deal with large systems through parallelization of both the ordinary differential equations (ODE) solver and the kinetic parameter estimation routines. DMB's implicit ODE solver lower-upper (LU) decomposition routine was parallelized on a CPU-GPU hybrid system using compute unified device architecture (CUDA) based MAGMA GPU libraries. The kinetic parameter estimation objective function simulations were multi-threaded on the CPU using Open Multi-Processing (OpenMP). These parallelization approaches decreased kinetic parameter estimation time for large molecular systems. To illustrate the robustness of this modeling framework three reactor models were investigated: lignin pyrolysis, cellulose pyrolysis and diesel hydrocracking.

A molecular-level kinetic model by means of an adapted Freudenberg lignin structure was developed for lignin pyrolysis at the temperature of 600 °C. A detailed reaction network was established taking into account primary and secondary lignin pyrolysis chemistries from literature. An exhaustive method was produced to handle the large structure reactivity by merging reaction networks. The adapted Freudenberg large lignin structure was modified based on a comparison of the reaction network and experimentally observed products. The evaluation of the kinetic model consisted of validation with molecular species measured in experimental work. The model output showed agreement with experimental results from Zhang et al. (Zhang, Resende, & Moutsoglou, 2014).

Thermogravimetric analysis (TGA) of cellulose pyrolysis was simulated using a molecular-level kinetic model. A temperature ramp between 373-1073 K was imposed in order to achieve the TGA simulation. A recursive optimization method was utilized to calculate the degree of polymerization of the starting cellulose molecule from literature bulk properties. The model's primary reaction pathways for cellulose decomposition allowed for degradation of the active cellulose molecule by hydrolysis and thermal degradation to create cellobiose, cellobiosan, and glucose. The mechanisms revealed by Agarwal et al. were used to degrade cellobiose (Agarwal, Dauenhauer, Huber, & Auerbach, 2012). An extension of the cellobiose mechanisms were applied to cellobiosan. The reactivity of glucose was captured by the reaction networks proposed in the work of Zhou et al. (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014). Monomer and dimer degradation pathways produced volatiles, furans, aldehydes, ketones, and char through a series of complex reactions. Linear free-energy relationships were applied to minimize the number of model kinetic parameters. Executing the cellulose pyrolysis model in TGA mode provided a prediction of the mass loss. The TGA simulation showed agreement with the experimental trends.

A molecular-level kinetic model has been developed for gas phase diesel hydrocracking on a bifunctional metal acid catalyst. The diesel feed was modeled as a complex mixture of paraffins, multi-branched isoparaffins, naphthenics, and aromatics. Linear free energy relationships in the Bell-Evans-Polanyi form were applied to minimize the number of model kinetic parameters (Bell, 1936) (Evans & Polanyi, 1935). The model contained 86 independent parameters for control of hydroisomerization and hydrocracking reactions. The model was utilized to develop

robust kinetic parameter estimation approaches involving parallel tuning methodologies.

Chapter 1

INTRODUCTION

1.1 Feedstock Conversions to Valuable Products

Material transformations into valuable products is a fundamental area of study in the chemical engineering discipline. It is exemplified by various biological, chemical, petrochemical, and energy engineering systems where product output is key and dictates real world day-to-day business and technology operations. These transformations typically involve complex molecular interactions that are driven by various process parameters. Understanding the relationship between molecular dynamics and process parameters allows engineers to modify and/or retrofit systems to produce a desired output. This work will focus on various systems involved in the energy and petrochemical landscape. It has been observed that the yield composition of these processes varies significantly based on alterations to several parameters: temperature, pressure, catalyst type, mixing, space velocity, and feed composition. Modifications of these process variables may have a combinatorial affect making it difficult to discern relationships based solely on experimentation. Therefore, kinetic models are valuable tools that provide a platform for process prediction, process design and development (PDD), and the elucidation of molecular mathematical associations.

1.2 Advantages of Kinetic Models

The widely utilized alternative to kinetic models are empirical models. Empirical models don't necessarily have a basis on theory, they rely heavily on the data used to train them. A characteristic empirical model employs simple mathematical expressions such as linear, polynomial, and logarithmic equations to describe experimental data. In the case of a polynomial empirical model, several parameters in the polynomial can be altered to achieve a better match between the model output and experimental quantities. If a reasonable fit is achieved between the observed experimental data and the empirical model, performance can be considered satisfactory. In terms of interpolation empirical models inherently perform well between data they are trained on but when attempting to extrapolate out of the range of the training data performance may be crippled due to uncertainty. The lack of a fundamental basis in empirical models for chemical processes produces unreliable and unpredictable results when data points are extrapolated. Interpolation and extrapolation are key aspects in the modeling of chemical processes, highlighting the need for both empirical and kinetic modeling approaches. These methods both have their separate strengths and weaknesses during application in chemical reaction engineering. Empirical models are simple and fast but lack extrapolation abilities, while kinetic models are complex, difficult to develop, and are not as fast as empirical models. The capability to perform interpolation and extrapolation in the same platform is a great advantage for kinetic models and is the reason why they continue to be highly sought after.

For millennia's humanity has attempted to predict or forecast outcomes whether it be one's future or the stock market. If predictions were easy everyone would be millionaires. Suffice to say predicting the output of a chemical process is

considered the gold standard in chemical reaction engineering. Kinetic models offer the ability to extrapolate and make real-time informed decisions with little to no expense. Kinetic models have a basis in the fundamental chemistry so they tend to have some forecasting qualities. Through kinetic models chemical reaction engineers can aid in the decision making process involving reactor units, chemical plants, and the overall business. Reactor units have various parameters which affect the reactor output understanding them is crucial when controlling the unit. The various parameters which affect the reactor unit are temperature, pressure, catalyst type, mixing, space velocity, and feed composition. A prime example of parameter variability is illustrated by the petroleum industry, where the refinery feed varies daily. A refinery's feed, crude oil, fluctuates due to availability and price. The composition of the crude oil affects chemical reactor units' output as well as the catalyst life. Predicting the feed effects on reactor units is critical for reactor operators to make decisions that can help avoid large issues or unit shut downs. From a business perspective, predictions can help make guided decisions that lead to higher profitability and or investments in new technologies. From a process design and development standpoint kinetic models can infer a reactor unit's incorporation in a plant wide flow. This is especially true as state of the art chemical plants have variabilities upstream or downstream which affect other processes. The strength of platform for prediction is clear but kinetic models also offer advantages during process design and development.

The addition of a kinetic model to the design and development cycle of a chemical process provides a connection between the chemical information and process parameters. A kinetic model can be used for conceptual reactor design due to its fundamental basis. The optimal process parameter trajectories of a chemical process

like temperature, pressure and catalyst type can be understood through a kinetic model. The optimal trajectories can be utilized to obtain a desired output. Process retrofitting, revamping and catalyst selection can also be supported by a kinetic model. Models can also provide assistance with process scale-up. Many of the advantages delivered by a kinetic model in the design and development cycle have been discussed. A chemical basis is necessary to produce the valuable information that supports the process design and development cycle.

Kinetic models utilize a combination of chemical and mathematical principles to describe the chemical kinetics of a process. This creates a link to the underlying science and a platform for both chemical and mathematical information to be displayed. A chemical basis allows for the translation of chemical information such as reaction chemistry and rate expressions to be represented mathematically. This link allows quantitative measurements of parameters that might be abstracted during experimentation. In the case of a hydrocracking, a quantitative understanding of adsorption constants and hydrogen catalytic effects can be obtained through a kinetic model even if they can't be directly measured experimentally. Through kinetic models the dynamics at various conversions can be calculated. The advantages that kinetic models provide are immense and are not unnoticed by industry and academics alike.

1.3 Feedstock Concentration

This thesis focuses on the transformations of energy and chemical feedstocks to valuable products at molecular resolutions. Specifically, the feedstocks of interest are biomass and petroleum. Each of these feedstocks can be thermally and catalytically processed to yield higher value molecular species. As mentioned earlier, several reactor and molecular parameters control the product composition of chemical

processes. Most biomass and petroleum processes are kinetically controlled and can lead to variable product distributions with slight reactor parameter variations. Therefore, it is necessary to understand the fundamental chemical kinetics involved to manage the propagation of molecular products.

1.4 Advantages of Molecular-level Models

This dissertation utilizes molecular-level kinetic models to understand the underlying molecular mechanisms of chemical processes. Molecular-level kinetic models improve upon all the advantages of lumped kinetic models. A long standing set of counter arguments still remain, why model systems at molecular resolutions? How do molecular-level kinetic models improve our understanding? The answers to these questions are:

- Advancement of analytical techniques
- Molecular management

The advancement of analytical techniques has been propelled forward by a desire to separate, identify, and quantify complex mixtures by molecular components (Phillips, 1991). Compound identification has been employed for over a century while quantification of each component has been a recent find. The transfer from solely qualitative information to qualitative-quantitative information has been crucial for the progression of molecular-level modeling. For molecular-level modeling a mass balance must be kept in a system, and if a modeler wishes to track molecular species, the quantities must be known. Techniques like gas chromatography (GC) and 2DGC can be used to identify and measure molecular quantities. As the adoption and advancement of analytical techniques continues, better models and modeling techniques can be applied to improve molecular-level kinetic models. The oil and gas

industries were some of the earliest adopters of GC technologies and have used them to heavily quantify feeds and reactor outputs. Therefore, if the feed and the reactor output can be quantified, understanding the intrinsic chemical kinetics of processes is not a farfetched goal. This thesis focuses on chemical processes that produce fuels and upgradeable bio-oils whose feeds and reactor output can be experimentally determined. Through the coupling of molecular experimental information and molecular-level kinetic models the molecules of system can be managed.

“Molecular management” is a term utilized by Exxon Mobil (EM) to describe the tracking of molecules across their fuels, lubes, and chemical businesses (Speeches, 2018). Many companies have followed suit in the chemical backdrop by also heavily valuing molecular information. Companies dedicated to modeling, like Aspen Technology Inc., have noticed this industry trend and began developing molecular-level modeling software tools (Aspen Assay Management, 2018). The management of molecular information allows for a deeper understanding of the mechanisms involved and product requirement specifications. The fundamental understanding of molecular mechanisms can be utilized to optimize a chemical process. In the case of fuels from renewable feedstocks, a fundamental understanding is dire in order to make these processes viable. An example of this is bio-oil, a fuel derived from the thermal processing of biomass, for which value is dependent on the molecules present in the oil. For fuels and chemicals in the petroleum industry a fundamental understanding of molecular mechanisms can lead to catalyst design, reactor design and system parameter changes to achieve a desired reactor output (Jaffe R. J., 1992). Through a molecular understanding the process design and development cycle of a chemistry can

be enhanced. Molecular-level kinetic models can be used to decipher the process parameters that control molecular product compositions.

Once a desired molecular output is achieved from a reactor unit, the product material can either be utilized directly by a consumer or upgraded in other reactor units. The product requirement specification (PRS) becomes extremely important at this stage since it details the expectations of that product. A typical example of PRS can be found in the fuel industry in which product specifications are key to meeting usage and environmental regulations. For diesel, sulfur amounts must be lower than 15 parts per million (ppm) to meet Environmental Protection Agency (EPA) standards in the United States (Diesel Fuel Standards and Rulemakings, 2018). Not only does the federal government in the United States set standards through the EPA, but states themselves also set standards on the fuels consumed in their own dominion (California Diesel Fuel Program, 2018). This creates a complex set of standards that fuel producers must meet. Besides meeting fuels specifications set by federal and state entities, fuels producers must also be meet performance specifications like octane number, cetane number, and cloud point temperature (Jaffe P. G., 2006). Variations in cloud point temperature for a diesel fuel can mean the difference between solid and liquid in the gas tank of a consumer. From a renewables perspective, the bio-oil produced from the pyrolysis of biomass must meet a certain molecular specification to determine its application. The product of lignin pyrolysis, yields high value products like meta-, ortho- or para-xylene and are very important to determine upgradability of the bio-oil. The complex molecular product specifications that must be met along with high number of variable parameters in chemical plants create the need for predictive tools to understand reactor variability.

Predictive tools based on molecular data provide information on the feed and process parameter changes. In a refinery environment feed variability is high and profits can be maximized by selecting the correct refinery feed. A prediction tool with molecular resolution can help determine the profitability of feeds based on experimental measurements. A refinery wide prediction at molecular resolutions can help determine which crude oil feed is the most profitable. Especially in an economically driven atmosphere, molecular prediction can track value in the reactor. For petroleum, a certain set of molecules might be preferred in the output of the refinery process based on their current value. In the case of diesel hydrocracking, the product fractions of naphtha, light naphtha, and light ends may be adjusted based on the prevailing product demand and market trends. Predicting the needed changes allows for on the fly modifications. Molecular prediction is very important to maximize profitability; especially in an ever changing chemical landscape, adaptability is strategic.

1.5 Modeling Large Systems

The molecules in biomass and petroleum have a large amount of possible reactive sites that can undergo a variety of chemistries. In turn, this leads to the creation of large reaction networks and species lists. Molecular-level kinetic modeling of biomass and petroleum feedstocks is quite difficult due to the sheer sizes and quantity of molecules in the systems. This thesis focuses on developing software tools and approaches to resolve the complexity brought on by large systems that reach mega sizes in model complexity. To grasp the difficulty, it helps to understand modeling's progression throughout the last few decades.

For years, chemical reaction engineering has been used to model chemical processes. Over this time period, deterministic modeling resolution has increased from lumped property information to full molecular detail. There is great difficulty involved in modeling at molecular scales due to the complexity of real systems. To a large extent, molecular-level modeling has focused on processes involving smaller molecules to keep the complexity tractable (Broadbelt, Stark, & Klein, 1994). Small model compounds have been used to represent the kinetics of larger systems. Small systems are not always representative and functional when matching real world chemical kinetics or product distributions. Especially with analytical techniques producing more information there is further constraints introduced to the model. Many industrialized chemical processes have large molecules or large sets of molecules as reactor feeds. Matching the complexity and the species output through small molecules may be an unsurmountable task. Therefore, it is necessary to model large molecules to capture the true essence of these chemical processes.

1.6 Challenging Regions in Molecular-level Modeling

A typical example of the complexity brought on by real processes can be exposed by crude oil. As mentioned crude oil is the feed of a refinery and can be described by thousands of molecular species. In fact the work by Hughey et al. was able to discern 11,000 unique molecular species in a sample of crude oil (Hughey, Rodgers, & Marshall, 2002). This did not account for the whole crude, only a measurable subset. To model a process of this size, the reactivity, and progression of each molecule through the reactor must be tracked. In a typical model development, first, a representative set of molecules must be chosen. Chemistry rules must then be applied to the set of molecules to produce a reaction network. This reaction network is

a set of reactions that describe molecular dynamics in the reactor. The reaction network, along with known initial compositions of the feed, can be used to model a chemical process. A large set of molecules and a large reaction network can cripple model development, specifically hampering model building, simulation, kinetic parameter estimation and editing phases. This is because the time required to analyze, solve, or change the model far exceeds the practical benefits of modeling. If any of the model development phases are hindered, development may come to a halt. This thesis introduces the software tools necessary to model large and small systems alike in a fashion where model development can precede in a streamlined manner independent of model size. Software tools and approaches will be illustrated by their application to three large molecule chemical processes in the petroleum and energy landscapes: lignin pyrolysis, thermogravimetric analysis (TGA) of cellulose and diesel hydrocracking.

The feeds in this thesis have a large molecular imprint. Cellulose and lignin vary from feedstock to feedstock. The actual structure of lignin cannot be determined solely by one experiment but approximated by many (Freudenberg, 1959) (Dellon, Yanez, Li, Mabon, & Broadbelt, 2017). Cellulose is a glucose polymer and lignin is a large macromolecule primarily composed of benzene rings with oxygenated linkages. Petroleum has been approximated as a large set of small and large molecules (Becker, Serrand, Celse, Guillaume, & Dulot, 2017). The main issue when modeling these large molecular systems is that they have many reactive sites. Reactive sites are areas on a molecule where bond breaking and/or bond forming occur during processing. To illustrate the reactivity issues posed by large molecules the pyrolysis of a lignin macromolecule will be discussed. An adapted molecular representation for lignin

molecule known as the Freudenberg molecule is shown below in Figure 1 (Freudenberg, 1959).

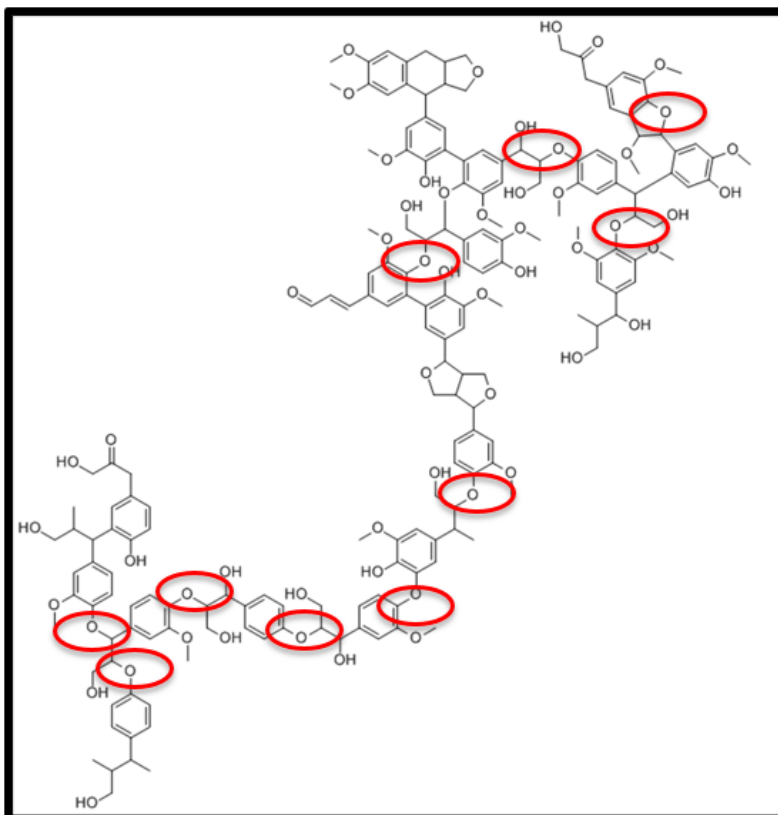


Figure 1: Large Molecule Representation (Freudenberg, 1959)

Notice in Figure 1 the red circles which depict a carbon-oxygen bond, highlighting ten reactive sites. On this molecule alone if ten reactive sites experience reactions, this can lead to ten factorial numbers of reactions depending the chemistry involved. Compounded with the fact that there are other reactive sites on the molecule and each site could undergo several different chemistries, millions of reactions and species could be formed just to represent the kinetics of one molecule. It is easy to imagine how extending the approach to more molecular species increases the number

of combinations leading to more species and more reactions. Leading to an intractable set of reactions depending on the molecules being modeled. Therefore, tools and techniques are needed to specifically address these large systems.

To model large systems, a large number of initial conditions and a large reaction network must be fed to a kinetic modeling environment. Large systems inherently take more work because they involve a greater number of components. They increase the time spent in every portion of model development. Starting with the model building phase, information from reactions is parsed and transferred to code in the form of representative rate laws and mole balances for each species. Higher number of reactions and or species means more time is spent writing code for the model. If model sizes are very large, compiling explicitly written code may become a rate limiting step or even impossible. The model simulation phase is also hampered by large models. Implicit numerical integration can be time consuming when solving high numbers of rate laws and species mole balances over differentiation lengths. The reason why simulation becomes slower is because during each time step, a Jacobian matrix must be solved as part of system of linear equations. The Jacobian matrix is an N by N matrix where N is the number of species. More species means a larger Jacobian must be solved increasing numerical integration time. The kinetic parameter estimation phase of large models is affected by the optimization heuristic, simulation time, number of datasets and number of parameters. This can be a very time consuming step that engulfs model development because of the number of repeated model simulations required for parameter estimation. This is especially true for larger models that have a high number of parameters with longer simulation times. The final phase of model development is editing, which is necessary if the model is not

capturing the desired trends. Editing can be used to add or remove reactions depending on the constraints of the reaction system. After editing the model, the model will need to be rebuilt and go through all model development phases in an iterative process until an adequate solution is met. Large models can increase the time spent in all phases of model development so this thesis has focused on developing the software tools to solve the unique challenges created when modeling mega molecular models.

1.7 Research Objectives

The research objectives of this dissertation are as follows:

1. Develop the capability to model large and complex systems
2. Reduce extraneous software dependencies
3. Improve model simulation/parameter estimation times by parallelization on a Windows PC and a High Performance Cluster (HPC)
4. Simplify user workflow by making the tools more intuitive

1.8 Thesis Scope

Structurally, this thesis can be explained by the idiom of *The Goose and Golden Eggs*. The idiom is about greed but may also be used to convey the idea that the goose is worth as much as the golden eggs. In this context this thesis can be thought of as a multi-faceted Ph.D. thesis in which software tools and reactor models were developed in parallel and are equally valued. The reactor model requirements and complexity inform the development of the software tool and the software tool reduces the burden of reactor model construction and evaluation. The software tool created to model large chemical processes was a kinetic modeling environment named Dynamic Model Builder (DMB). In this thesis, the software tool was utilized to create

models for three chemical processes: cellulose pyrolysis, lignin pyrolysis and diesel hydrocracking.

Ch. 2 provides an introduction into the existing software suite named the Kinetic Modeler's Toolkit (KMT). The chapter details the four software tools that make up KMT. An introduction to the Dynamic Model Builder will begin here since it is now an integral part of the KMT suite.

Ch. 3 provides details about the development of the Dynamic Model Builder and its functionality and usage. This chapter also illustrates the differences between DMB and the previous modeling framework, the Kinetic Model Editor (KME). It will also introduce technological advancements that make DMB a robust framework.

Ch. 4 provides details on the development of a pathways molecular-level kinetic model for lignin pyrolysis. The complexity behind modeling large molecular structures will be illustrated. An exhaustive reaction network merging approach will be introduced. Various reaction networks and model results will be mentioned.

Ch. 5 provides details on the development the simulation of a Thermogravimetric Analysis (TGA) experiment based on pathways molecular-level kinetic model. The model chemistry and kinetic information was based on available cellulose pyrolysis literature. A new approach to simulate TGA was introduced that took into account phase equilibria. TGA simulation and model results will be discussed.

Ch. 6 articulates the development of a pathways based molecular-level model for diesel hydrocracking. This project in particular used the complete arsenal of tools available to a molecular-level modeler in the Klein Research Group. Software

parallelization was utilized to decrease the heavy burden caused by kinetic parameter estimation.

Ch. 7 describes kinetic parameter approaches used during this thesis work to give future users ideas on how to perform their own parameter estimations

Ch. 8 describes the development of reliable communication between all the tools in the Klein Research Group. An approach for the streamlining of the entire development is will be discussed in detail.

Ch. 9 summarizes the developments of this thesis as well as discusses future areas for improvements before finishing with a closing remark.

Chapter 2

KINETIC MODELING TOOLKIT

2.1 Kinetic Modeler's Toolkit (KMT) Software Ecosystem

The Kinetic Modeling Toolkit (KMT) is a software suite developed to model chemical processes here at the University of Delaware. It is composed of four separate software tools that work together to enable molecular-level modeling. The four software tools are a property generator, an initial condition generator, a reaction network generator, and a kinetic model builder. The initial development of the software tools dates back to the 1990's and was developed by a variety of graduate students¹ (Broadbelt, Stark, & Klein, 1994) (Wei W. , et al., 2008) (Bennett, 2009) (Horton, Mohr, Zhang, Petrocelli, & Klein, 2016). The development of the tools continues as emerging technologies and advances in computers streamline development and performance. Figure 2 illustrates the four software tools as they are now: the Interactive Network Generator (INGen), the Property Database (PD), the Initial Condition Generator (ICG), and the Dynamic model Builder (DMB). Figure 2 illustrates the workflow one could use if all tools were utilized for a project, first INGen would be applied to produce a reaction network and species list. Secondly the properties of the molecules from the species list in INGen would be obtained by using

¹ The number of citations does not reflect the number of contributions to the KMT software suite. Every student mentored by Michael T. Klein has made contributions whether directly or indirectly.

the PD application. Thirdly, the initial conditions to the initial value problem in the kinetic modeling environment can be found using the ICG application. Finally, to simulate the molecular-level chemical kinetics, the reaction network, species properties, and initial conditions can be used by the kinetic modeling environment DMB.

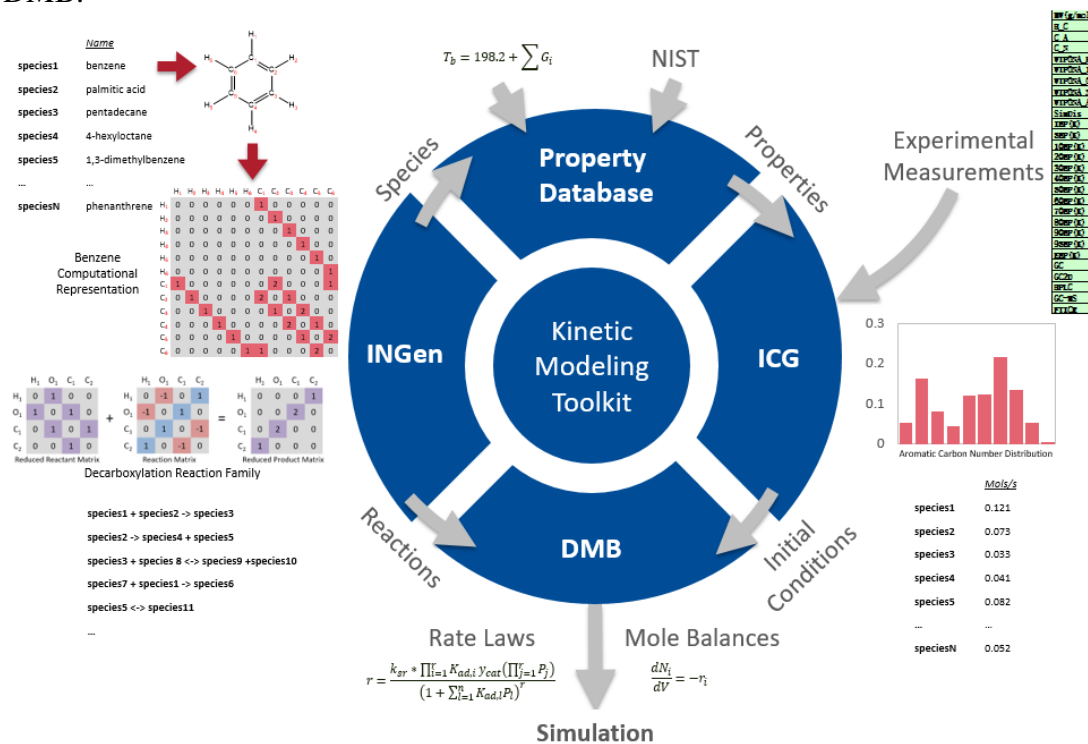


Figure 2: Kinetic Modeling Toolkit

2.2 Interactive Network Generator (INGen)

INGen is a C programming language based software tool utilized to develop reaction networks. It has graphical user interface designed in Visual Basic. For small systems, INGen may not be necessary to use since the reaction network may be easily written by hand. As the complexity of systems increases in terms of species and types of reaction that can occur, it becomes increasingly difficult to write all the reactions by hand. INGen can be thought of as a bookkeeping tool that can track all the species,

react them, and write all reactions iteratively. A computational tool that can accomplish reaction network development is really important since it alleviates a process that can otherwise be error prone and tedious. INGen does not inherently prune the network down; instead, it produces every possible reaction based on a set of user-defined rules. INGen functions on the idea that molecules can be represented computationally as matrices and this idea will be explained in the following sections.

2.2.1 Computational Representation of Molecules

A molecule can be represented as a bond electron matrix where bonding can be described by numerical values in which '1' describes a single bond, '0' describes the lack of a bond between atoms, and so on. Figure 3 illustrates a bond electron matrix for an isohexane molecule. Notice the matrix is sparse and contains a substantial amount of zeros. For this reason, it is stored as an adjacency list. An adjacency list is a file that only contains non-zero entries. An example of an adjacency list is shown in the Appendix section.

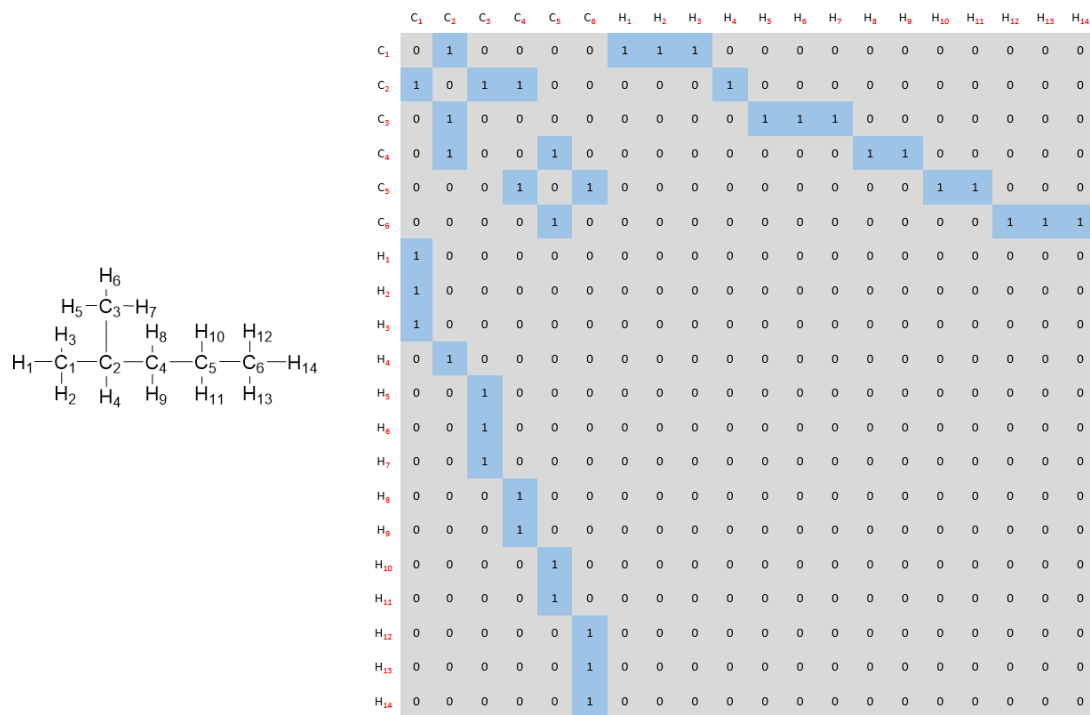


Figure 3: Bond Electron Matrix Isohexane Example

2.2.2 INGen Reaction Network Development

In INGen, a user can define a set of reaction rules that find and change reactive moieties on molecules. These rules can be applied to the entire subset of molecules that share similar features. To illustrate INGen functionality, Type C β -scission chemistry was applied to an isohexane molecule to break it into two propane molecules. Type C β -scission breaks the bond on the isohexane molecule shown in Figure 3 between C₂ and C₄. Each reaction requires atoms to be balanced, so diatomic hydrogen is cleaved into two hydrogen atoms as well. INGen reduces the reactive sites to only include the atoms that are undergoing bond forming and breaking. The reduced reactive site can be described in matrix form as the reduced reactive matrix shown in Figure 4. Subsequently, simple matrix algebra can be used to add the reduced reactant matrix to the reaction matrix to produce a product matrix. The reaction matrix

describes the bond forming and breaking in a molecule. In the reaction matrix a negative value signifies bond breaking while a positive value indicates bond forming, with '0' signifying no change. In the Figure 4 illustration, the bonds that are broken in the molecule are between C₂ and C₄. The bond forming occurs between C₂ and a hydrogen atom and C₄ and a hydrogen atom.

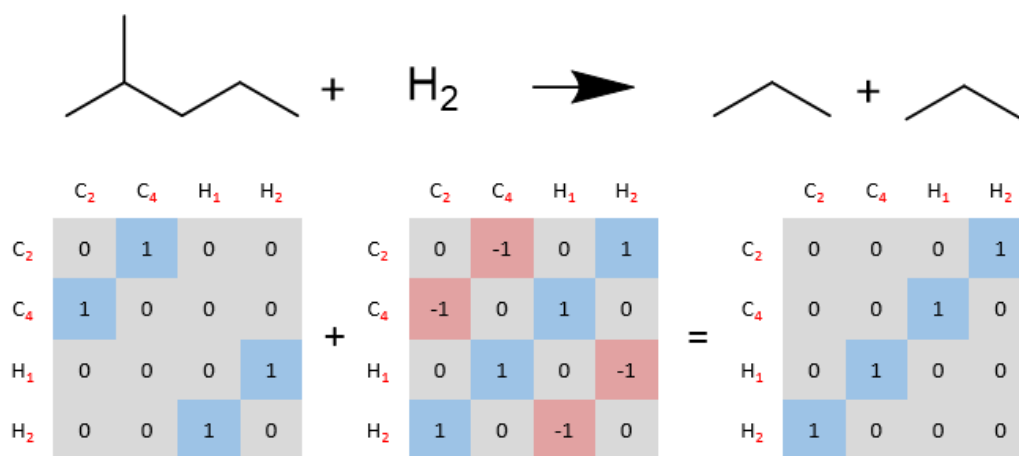


Figure 4: Isohexane Type C Beta Scission Example (Includes Reactant Matrix, Reaction Matrix and Product Matrix)

This same process is done iteratively for every molecule available in the INGen model. More than one reactive moiety can be reacted by applying the appropriate site selection routine and reaction matrix, making INGen a library of reactions. This library is extensive and has been compiled for several years by many graduate students (Bennett, 2009) (Moreno, Li, Lee, Huber, & Klein, 2013). The reaction network describes the reactivity of molecules through a reactor and is a blueprint for the rate laws and mole balances. After an INGen model has been developed, the typical workflow is to find the species properties using the property database application.

2.3 Property Database (PD)

The Property Database (PD) application stores relevant species properties in a SQLite database. If the molecule is not available in the database, then the PD application runs PropGen, a property generation tool (Hou, Software Tools for Molecule-based Kinetic Modeling of Complex Systems, 2011). PropGen uses Benson and Gani group contribution methods to estimate species properties (Benson & Cohen, 1993) (Gani, 1994). Propgen uses bond-electron matrices to determine the atom groupings necessary for group contributions methods. These bond-electron matrices are the same ones used by INGen. The database also allows for additional property information to be stored for each molecule based on literature or experimental work. This additional information is important because, sometimes, group contribution methods do not produce accurate molecular properties. If better properties are available, the need to calculate them is not necessary. Therefore, the database allows for many properties to be stored and accessed. Once the properties and the reaction network have been developed for a molecular-level model the final thing necessary is initial conditions.

2.4 Initial Condition Generator (ICG)

The numerical approximation methods of the solution for ordinary differential equations that describe the reactor kinetics are initial value problems (IVP) and requiring initial feed concentrations. In a simple molecular system, the input is generally known. This is not always the case as complex feeds are present throughout nature. Petroleum and biomass are complex feeds that are extremely variable and can be quantified through experiments. The Initial Conditions Generator (ICG) is a C# application that describes molecular feeds as probability density functions (PDFs) and

correlates that information to experimental bulk properties through an objective function (Horton S. R., 2016). For unknown feed areas, approximations can be made using continuous PDFs functions. When molecular information is known, histograms can be applied. An objective function of the form below can be used as a metric for goodness-of-fit between the model feed and the experimental measurements. Therefore, a heuristic method like simulated annealing can be used to vary tunable PDF parameters and minimize the objective function. The objective function is a metric quantifying the difference between experimental and composition model results weighed by a confidence of the measurement. The objective function in this work is of the chi-square form as shown in equation (2.4-1).

$$\chi^2 = \sum_{i=0} \left(\frac{Observed - Expected}{Weight} \right)^2 \quad (2.4-1)$$

This approximates areas of unknown information of the feed to fully quantify the feed. Once the feed is specified, the initial conditions, reaction network and species properties can be fed to the kinetic modeling environment.

2.5 Dynamic Model Builder (DMB)

The Dynamic Model Builder is the newest addition to the arsenal of software tools in the Kinetic Modeler's Toolkit. It is a next generation kinetic modeling environment that focuses on technological advances to increase performance in deterministic simulations and kinetic parameter estimation. DMB will be a focal point in this thesis and will be fully introduced in the following chapter.

Chapter 3

DYNAMIC MODEL BUILDER: A C++ REACTION ENGINEERING MODELING FRAMEWORK

3.1 Introduction

The Dynamic Model Builder (DMB) is a C++ modeling framework designed to model the chemical kinetics of small and large complex systems. The framework has two main functionalities:

- Simulation
- Kinetic Parameter Estimation

A simulation is achieved by solving an initial value problem for a set of ordinary differential equations (ODEs) that describe the material and energy balances in a chemical process. Kinetic parameter estimation is performed by using a probability-based metaheuristic method to approximate the global minimum of an objective function based on simulation output. DMB's output from a simulation is molecular by nature but can be transformed into bulk information via appropriate bulk property calculations. The knowledge that can be gained by navigating between molecular and bulk spaces can be used to optimize and/or control chemical processes. These functionalities make the DMB framework a powerful tool that can be utilized to predict, analyze, and optimize chemical processes at molecular-level resolutions. This overview section will compare the DMB framework to the previously employed environment in the Klein Research Group (KRG) and describe how it differentiates.

Advantages will be discussed along with short a description of some of the functionalities available.

DMB was created as a next generation kinetic modeling environment, a replacement to the Kinetic Model Editor (KME). Throughout the years KME has been the extensively utilized within the Klein Research Group at the University of Delaware to model chemical processes, but as the complexity of systems has increased, certain limitations in KME have been exposed. The consequence of these limitations is marked by an increase in model development time. Prolonging the time spent in model development cycle can adversely affect the model building, simulation, and parameter estimation phases, to the point where development can become severely crippled. The model development process is typically an iterative process in which the user has to reconstruct the model to match experimental results as illustrated in Figure 5 by the editing phase. A streamlined process that allows for simple model modifications and quick progression through development phases is critical. To illustrate the differences between KME and DMB the inner workings of each framework will be outlined.

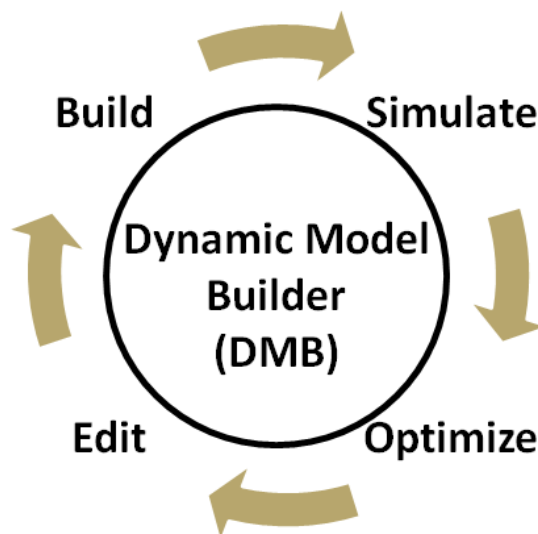


Figure 5: Dynamic Model Builder Model Building Process

KME development began in the 1990's as a software tool called Ordinary Differential Equation Generator (ODEGen). ODEGen can be utilized to automatically parse information from reaction network files to write material balances to programming code (Wei W. , et al., 2008). To allow for an easier user experience a Visual Basic Application (VBA) user interface in excel was developed to surround the ODEGen software. The package of ODEGen and the VBA Excel graphical user interface together is called KME and follows the workflow below:

1. Parsing information from reaction networks
2. Writing rate laws and mole balances to code files
3. Compiling the code files to make model executables

Dissecting KME's code illustrates its age and limitations since it relies on practices that were the norm more than 20 years ago. KME's three main issues is that it is dependent on languages facing adoption issues, tandem program design, and a UNIX virtual environment. Much of the programming code for KME was written using a blend of languages because during its development they were necessary to meet its

needs. ODEGen is written in Perl an interpreted language used for character searching and parsing. Perl is now the language is facing adoption issues since modern languages offer the same support with better performance. The decrease in performance is primarily due to Perl interpreting human readable code to machine language line by line, which is slower when compared to modern compiled languages which convert all the code into machine language at once. As model sizes increase the performance differential becomes more and more apparent and the model building process becomes a time consuming step.

In the current KME code structure, VBA is utilized in tandem with the Perl to write varying portions of C code that describe chemical rate laws. This implementation can lead to confusion when attempting to understand the code since the two programming languages both change the same code at varying times. Sometimes even overwriting methods written by the other language. Understanding how to change KME is important for the developers who may wish to change some functionality of the core software to suit their needs. These code developers may find it difficult to alter and debug the code since the debugging environments for Perl and VBA can be rather tedious, non-existent, or cryptic compared to what is used in modern integrated development environment (IDE).

Once the Perl and VBA programs are utilized to make C code a UNIX virtual environment is required by KME to compile this new code. There are two reasons why a UNIX virtual environment is employed. The first is that UNIX offers free compilers as part of the General Public (GNU) license. This means that code can be compiled without investing any capital. The second reason is that the KME framework was developed on older computers whose operating system was UNIX based. The most

common platform used by our collaborators and ourselves is now Windows therefore a virtual environment is necessary. Requiring a virtual environment adds both large permission requirements for the installation of KME and a virtual machine overhead while running the models.

The absence of Perl and VBA is a key differential between the DMB and KME frameworks. DMB was written in C++, a language with regular expression support which can perform the same tasks as Perl. C++ is also well supported and can be easily compiled on Windows operation systems (OS) via Visual Studio, which are both Microsoft products. DMB also uses technological advances to increase performance by parallelizing heavy compute portions of code and functioning in different environments from a Windows desktop to a Linux-based high performance cluster (HPC). DMB's design focused on achieving high performance and utilizing existing technologies to create a synergy across the model building process.

To perform a simulation DMB first parses a reaction network file and stores this information in memory. This footprint in memory can then be accessed dynamically to create rate laws and mole balances for simulations during run time. Due to the fact that DMB reads and utilizes all the necessary information for a model simulation during run time there isn't a need for a compiler. The DMB executable is also not model dependent because it relies on model configuration files to describe and build the model in memory. DMB scales to a wide range of reaction network sizes, easily reaching sizes greater than 25,000 reactions. A model larger than 25,000 reactions is prohibitively difficult to achieve in KME due to ODE file size reaching sizes larger than a GNU Compiler Collection (gcc) compiler can transfer into machine

code. The DMB framework on the other hand can model large reaction networks without issue.

Not only does DMB dynamically store reaction network information, it can also dynamically change the rate expression based on simple language set in the configuration files by the user. The DMB framework can handle non-catalytic and catalytic chemistries based on Langmuir-Hinshelwood-Hougen-Watson (LHHW) formalisms based on user inputs. DMB is also designed to be multi-phase framework, modeling gas, liquid, vapor liquid equilibrium (VLE). Therefore, DMB is a valuable improvement on the KME framework that works faster with a greater potential to handle the challenges of molecular-level kinetic modeling in the future. In the following sections the installation, theory, program, model configuration, and instructions for the DMB framework will be discussed.

3.2 Installation

The DMB framework can be utilized in two ways:

- Non-developer
- Developer

The utilization methods will be covered in detail in the upcoming sections.

3.2.1 Non-Developer

A DMB executable builds and simulates kinetic models during run time based on simple reactor configuration files. The non-developer option for DMB is designed for users that wish to use the core functionality of the software application to develop kinetic models without delving into programming. DMB can be packaged for distribution on Windows and UNIX operating systems (OS). When operating in the

Windows OS, the executable is dependent on a few system dynamic linking libraries (DLLs) which are outlined in the following Table 1. Table 1 also describes the structure of the modeling framework, including executables, model folders, and DLLs. In the UNIX OS the extraneous libraries can be statically linked into the executable bypassing the need for any external libraries only requiring a model folder.

Table 1: DMB Framework Design and Requirements

DMB Framework Requirement	Description
DynamicModelBuilder.exe	Dynamic Model Builder Framework in an executable
Model Directory	Model folder holds: <ul style="list-style-type: none"> • Reactions • Kinetic parameters • Dataset information • Etc.
sundials_cvodes.dll	ODE solver library
sundials_nvecserial.dll	Vector library for ODE solver
concr140.dll	Windows Exe Redistributable Package
msvcp140.dll	Windows Exe Redistributable Package
ucrtbased.dll	Windows Exe Redistributable Package
vccorlib140.dll	Windows Exe Redistributable Package
vcruntime140.dll	Windows Exe Redistributable Package
vcruntime140d.dll	Windows Exe Redistributable Package

3.2.2 Developer

As a DMB framework developer existing DMB code can be altered or new code subroutines added. The DMB framework can be compiled in two operating systems (OS) utilizing the following compilers:

- Visual C++ compiler in a Microsoft Visual Studio Project for Windows OS
- GNU/PGI/Intel C++ compilers in a Vanilla Linux OS for High Performance Cluster (HPC) utilization

Current performance metrics indicate that the Intel compiler is the most robust compiler solely based on lower simulation times. It was also observed that the Linux OS simulations outperform the Windows OS simulations on the same metric. Current code development workflow starts in, Microsoft Visual Studio (MSVS) which is an integrated development environment (IDE) that can be used to compile, debug, and profile code. Subsequently once the performance of the code has been deemed suitable in Microsoft Visual Studio environment it can be transferred and compiled in the Linux OS. This approach is used because the Microsoft Visual Studio is a robust code analysis tool that can be used to track all variables within the framework. As eluded, the source code of the software is available to be edited to fit the user's needs. This can be beneficial to determine how the model equations are being constructed and also allow the addition of new or different functionality in the software.

To install and run the source code on a Windows OS, MSVS is required. Currently the developer package must be installed in the C: directory. The path to the Dynamic Model Builder project is Path = C:\Codes\CVODES\DynamicKineticModel. Once in the DMB project directory double click on the MSVS project solution named Dynamic Kinetic Model. After the project solution is opened the project can be cleaned and rebuilt by utilizing the Build tab. After a successful build the user can access and play with source code.

The user may decide how much power to yield over the framework based on needs. As stated by Voltaire and famously rephrased by Ben Parker in the Spiderman comic book series: "*With great power comes great responsibility*". With this being said delving into the developer side of the DMB framework requires some programming experience with C++ object orientation. The framework is open to

alteration and additions to meet the user needs but the programmer should be aware of the effect the changes can have across the framework.

3.3 Chemical Engineering Principles in the DMB framework

The DMB framework utilizes fundamental chemical reaction engineering to simulate chemical kinetics. As pointed out by H. Scott Fogler in the Elements of Chemical Reaction Engineering the foundation of chemical reaction engineering stands on six pillars (based on the Pillars of Chemical Reaction Engineering) illustrated in Figure 6 (Fogler, 2006). The first four of the six pillars in Figure 6 can be managed by the DMB framework: mole balances, stoichiometry, rate laws, and energy balances. The versatility of the DMB framework makes it very strong tool for modeling chemical reactors.

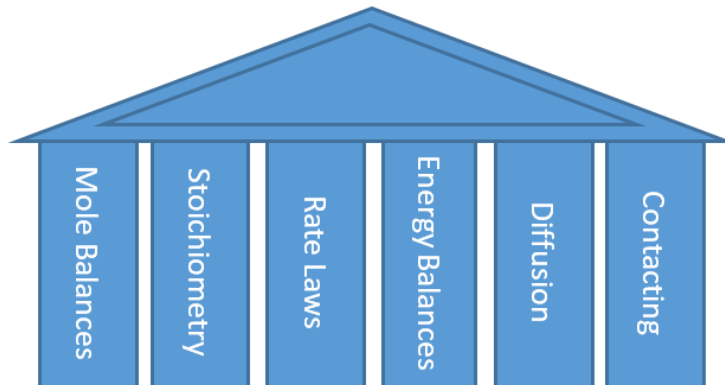


Figure 6: Adapted Pillars Chemical Reaction Engineering (Fogler, 2006)

As mentioned in the overview section the DMB framework is designed to handle variable systems sizes. Large systems have many variables involving mole balances, stoichiometry matrixes, rate laws, and energy balances. Consequently, large systems require kinetic parameter reduction to be tractable. Thus the DMB framework

is embedded with the concepts to handle this difficulty. The linear free-energy relationship (LFER) concept reduces the kinetic parameters by relating kinetics to structural properties (Bell, 1936) (Evans & Polanyi, 1935). The rate laws and the LFER concept require the enthalpy of reaction and Gibbs free energy of reaction. Therefore, the formulations utilized to find the thermodynamics of the system are presented here. After the chemical engineering principles have been introduced, a small primer into numerical integration will also be covered.

3.3.1 Mole Balances

The first pillar in the pillars of chemical engineering (illustrated in Figure 6), mole balances, defines the rate of flow in and out ($F_{j0} - F_j$), the rate of generation by chemical reactions ($r_j dV$), and the rate of accumulation (dN_j/dt) of species j within chemical reactors. The flow in and out of the reactor (F_j) is moles per unit time, molar flow. The rate of reaction (r_j) for a given species j is the number of moles of species j reacting per unit time per unit volume. The accumulation term (dN_j/dt) is in moles per unit time. The general mole balance equation for chemical reactor system is illustrated in equation (3.3.1-2). Equation (3.3.1-2)3.3.1-2 can be developed into three design equations based reactor configuration and assumptions. The three design equations covered in the upcoming sections define the three reactor types utilized within the DMB framework.

$$F_{j0} - F_j + \int_{V_i}^{V_f} r_j dV = \frac{dN_j}{dt} \quad (3.3.1-2)$$

The first DMB reactor type is a batch reactor, it does not have flow in or out of the system and the reactor volume is constant. Altering equation (3.3.1-2) based on the mentioned design needs for a batch reactor produces the following design equation (3.3.1-3).

$$\frac{dC_j}{dt} = r_j \quad (3.3.1-3)$$

The second DMB reactor type is a continuous stirred tank reactor (CSTR). The CSTR reactor has flow in and out of the reactor while maintaining a constant reactor volume. The equation (3.3.1-3) describes a non-steady state CSTR reactor:

$$\frac{dC_j}{dt} = \frac{v_0}{V} (C_{j0} - C_j) + r_j \quad (3.3.1-4)$$

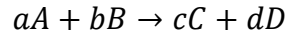
The third DMB reactor type is a plug flow reactor (PFR). The PFR reactor has flow in and out the system without any accumulation. The PFR design equation (3.3.1-5) assumes a constant cross-sectional area and ideal mixing radially.

$$\frac{dF_j}{dz} = A_c r_j \quad (3.3.1-5)$$

The batch and CSTR design equations are a function of concentration and time while the PFR is function of molar flow and the length of the reactor. As mentioned earlier, the rate of reaction for species j accounts for the moles of species j reacting in the system. The relative rate reaction for species j versus other species must be understood through stoichiometry to fully account for the formation and consumption of species j in the system.

3.3.2 Stoichiometry

The stoichiometry is the relationship of one species concentration to another within a reaction rate law. Stoichiometry can be illustrated by the use of a generic reaction:



In the generic reaction above the lower case letters denote the stoichiometry of the system; upper case letters denote the species. The equation below illustrates the relationship between the rates of reactions and the stoichiometry of the generic reaction.

$$\frac{r_{1A}}{-a} = \frac{r_{1B}}{-b} = \frac{r_{1C}}{c} = \frac{r_{1D}}{d}$$

Applying this to a simple batch reactor, we have the following mole balances for each species.

$$r_{1A} = \frac{dC_A}{dt} = -a * r$$

$$r_{1B} = \frac{dC_B}{dt} = -b * r$$

$$r_{1C} = \frac{dC_C}{dt} = c * r$$

$$r_{1D} = \frac{dC_D}{dt} = d * r$$

Through the use of stoichiometry, the relative rates of reaction can be calculated. The DMB framework stores the stoichiometric information from reactions in a 2 dimensional array known as a stoichiometry matrix. The first dimension of the

array is the species and the second dimension is the number of reactions. For the system of above the stoichiometry matrix is a single array as shown below in Table 2.

Table 2: Stoichiometry Matrix for $aA + bB \rightarrow cC + dD$

Species	Reaction 1
A	-a
B	-b
C	c
D	d

Adding the additional generic reaction increases the size of the matrix as shown in Table 3.

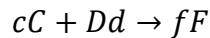


Table 3: Stoichiometry Matrix including $cC + dD \rightarrow fF$

Species	Reaction 1	Reaction 2
A	-a	0
B	-b	0
C	c	-c
D	d	-d
F	0	f

To decrease the size of the stoichiometry matrix stored in memory, a sparse matrix type is utilized omitting the zeros. The stoichiometry matrix can be utilized with the rate laws of the system to solve the net rate of reaction of a species i as shown in equation (3.3.2-6). This net rate expression is the general mole balance for each species utilizing the stoichiometry matrix, v_{ij} , where i is the species and j iterates over the reactions.

$$\frac{dC_i}{dt} = r_i = \sum_j^{rxns} v_{ij} r_j \quad (3.3.2-6)$$

3.3.3 Rate Laws

The DMB framework can handle three types of rate laws:

1. Microkinetic
2. LHHW
3. Power Law

The first type of rate law is the microkinetic/elementary reaction rate law, which can either be irreversible or reversible. Using the generic reaction mentioned in the stoichiometry section, $aA + bB \rightarrow cC + dD$, an irreversible micro kinetic rate law based on concentrations is shown in equation (3.3.3-7) below:

$$r = k_c C_A^a C_B^b \quad (3.3.3-7)$$

The irreversible microkinetic rate law is a function of both reactant concentrations and the concentration rate constant, k_c . The concentration terms have exponents which are referred to as the orders of reaction. For the microkinetic rate law, the orders of reaction are as follows: an order with respect to reactant A and b order with respect to reactant B. Note that the absolute value of the reactant stoichiometry coefficients and the reactant order for a micro kinetic rate law is equal. The overall reaction order is, $n = a + b$. The concentration rate constant, k_c , in the DMB framework is of the Arrhenius form. Equation (3.3.3-8) is a function of the pre-exponential (A), activation energy (E_A), and the temperature (T) in the system.

$$k_C = A e^{-\frac{E_A}{RT}} \quad (3.3.3-8)$$

The generic reaction can also be reversible, $aA + bB \rightarrow cC + dD$, meaning both the forward and reverse reactions are important. The rate expression displayed below (3.3.3-9) is for a microkinetic reversible reaction.

$$r = k_C \left(C_A^a C_B^b - \frac{C_C^c C_D^d}{K_{eq}} \right) \quad (3.3.3-9)$$

The reversible rate law includes the equilibrium constant (K_{eq}) which, at chemical equilibrium, equals $K_{eq} = C_C^c C_D^d / C_A^a C_B^b$. This solution for K_{eq} can be obtained by setting the rate in the reversible microkinetic rate law above to zero. The equilibrium constant can also be calculated from the Gibbs free energy of reaction by utilizing equation (3.3.3-10) below. The latter method is used in DMB to calculate the equilibrium constant by using a Gibbs free energy of reaction from quantitative structure property correlations for each reaction i .

$$K_{eq_i} = e^{-\frac{\Delta G_{rxn}(T)_i}{RT}} \quad (3.3.3-10)$$

When dealing with catalytic systems, LHHW formalisms can be applied to the microkinetic rate expression to transform it into a rate expression that accounts for catalytic effects. The LHHW rate expression employed in the DMB framework is a surface rate controlled rate law. The surface rate controlled expression is the most common LHHW expression as it is prominent in typical catalytic chemistry. A surface rate controlled rate law accounts for molecular adsorption as shown in equation (3.3.3-11). The LHHW rate expression requires the concentration of sites (C_T), adsorption

constants (K_{ads}), and hydrogen concentration (C_{H_2}) in the specific case of a hydro processing system. The LHHW rate expression contains Langmuir adsorption isotherms that describe the competitive adsorption of molecules to the catalytic sites. The denominator of the adsorption isotherm is the adsorption group and it is raised to the power of α . This term, α , can be determined from reaction mechanisms and experimental observations.

$$r = \frac{k_{sr} C_T K_{ad,A} K_{ad,B} \left(C_A C_B - \frac{C_C C_D}{K_{eq}} \right)}{\left(1 + K_{ad,A} C_A + K_{ad,B} C_B + K_{ad,C} C_C + K_{ad,D} C_D \right)^\alpha} f(C_{H_2}, \beta) \quad (3.3.3-11)$$

When catalytic systems involve hydrogen, the hydrogenation reactions of unsaturated molecules are thermodynamically favored at moderate temperatures and high hydrogen pressures, but the free energy of hydrogenation of different functional groups are not the same. This difference may affect the relative rate of hydrogenation in competitive reactions. For this reason, hydrogen dependence is handled by the term, $f(C_{H_2}, \beta)$, which is a function of the hydrogen concentration and tunable parameter β . The exact form used in the DMB framework is covered in the LHHW file section.

The third rate law type, power law rate expression, is used when the reactant orders are not equal to absolute value of the stoichiometric coefficients (Noted earlier for the microkinetic rate law). Using the generic reaction equation, $aA + bB \rightarrow cC + dD$, a power law rate expression can be developed as shown in equation (3.3.3-12).

$$r = k_C C_A^\alpha C_B^\beta \quad (3.3.3-12)$$

The power law rate expression (3.3.3-12) uses new reactant orders, α and β which are not necessarily equal to the a and b values in equation (3.3.3-7). If the

reaction is reversible then a reversible power law type (3.3.3-13) is employed where the new reactant orders are α , β , γ , and δ for a four species system of A , B , C , and D . The DMB framework allows for simple control over power law orders in reactions by utilizing easy to understand notation in the reaction network files.

$$r = -k_c C_A^\alpha C_B^\beta C_C^\gamma C_D^\delta \left(1 - \frac{C_C^c C_D^d}{K_{eq} C_A^a C_B^b} \right) \quad (3.3.3-13)$$

LHHW formalisms are also applied to the power law rate expression in a similar way to what has been done before for microkinetic rate law expressions. The inclusion of the power law rate expression to the arsenal of rate law expressions in the DMB framework is key to control rates that do not coincide with microkinetic affectations. The transfer of rate law to power law rate expressions is straightforward in the DMB framework and is a feature that was not supported in the previous framework KME. Current set of rate law expressions provide DMB users a variety of options when modeling large systems.

3.3.4 Linear Free Energy Relationship (LFER) Concepts

The DMB framework is designed to handle large systems that have many reactions and species. For a large system with 10^3 reactions or more, if each reaction were to have a unique rate constant, the amount of experiments necessary to find the values of the rate constants would be intractable. Studies on the reactions of complex mixtures have shown that classes of molecules within these mixtures react in similar ways and can be defined as reaction families. Within the reaction families' reactivity only differs between compounds by substituents effects. A quantitative structure property correlation (QSPC) can be related to the rate constant to account for the

substituent effect while reducing parameter numbers. Equation (3.3.4-14) is a general expression of quantitative structure property reactivity (QSPR) that includes a reaction index, a QSPC (Klein, 2006).

$$\ln k_i = a + b(RI)_i \quad (3.3.4-14)$$

An alternative explanation is that these reaction families represent homologous series of reactions that have been experimentally shown to have linearly related rate constants with respect to certain reaction indices, also shown in the equation below. The reactivity definition may vary depending on the system. DMB generally defaults to the Bell-Evans-Polanyi LFER where, $E^* = E_j + \alpha_j \Delta H_{rxn,i}$, as shown in equation (3.3.4-15) (Bell, 1936) (Evans & Polanyi, 1935).

$$\ln k_{i,j} = \ln A_j - \frac{(E_j + \alpha_j \Delta H_{rxn,i})}{RT} = \left(\ln A_j - \frac{E_j}{RT} \right) - \frac{\alpha_j \Delta H_{rxn,i}}{RT} \quad (3.3.4-15)$$

In the Bell-Evans-Polanyi LFER (3.3.4-15), i is the reaction index and j is the reaction family index. The reaction index in the LFER is the enthalpy of reaction. The reaction family LFER has three adjustable parameters: the pre-exponential, activation energy, and the alpha term. The alpha term controls the impact of the differences of the heat of reactions for each reaction, thereby differentiating the rates of the individual reactions in a reaction family.

The reaction index is a QSPC and can vary from system to system. The modeler must decide how to capture the substituent effects. It will be also noted that the enthalpy of reaction does not vary substantially in a reaction family when calculated from group contribution methods. This is because the enthalpies calculated

from group contribution methods use atomic groups to calculate properties. A set of molecules in a reaction family stereotypically have the same atomic groups that react in the same way therefore there is a low quantitative variance in enthalpies of reaction. The experimental work by Korre and Klein showed a variance in enthalpy of reactions for a homologous series of alkyl aromatics (Korre, Quann, & Klein, Hydrocracking of Polynuclear Aromatic Hydrocarbons. Development of Rate Laws through Inhibition Studies, 1997). The modeling work by Martinis and Froment showed that the carbon effect in the enthalpy of reaction is actually due to a stabilization effect from the olefin attaching to the metal site (Froment J. M., 2006). This stabilization is distributed along the carbon chain. This idea has been added to DMB framework as a modified enthalpy of reaction shown in equation (3.3.4-16) below:

$$\Delta H_{rxn,i}(Carbon\ Dependence) = \Delta H_{rxn,i} + \delta C_{Min\ Num}^{Prod} \quad (3.3.4-16)$$

In the modified enthalpy of reaction i , is the reaction index and delta, δ , is the multiplier to control the carbon number dependence. Currently the carbon number dependence is set to the minimum carbon number of the product side. This carbon number dependence can be changed in code based on experimental trends.

3.3.5 Thermodynamics

Group contribution methods are used to calculate the pure component properties that are fed into DMB. These properties come from an application known as Property Database, which will not be covered in depth here. It is mentioned though that the Property Database uses Gani and Benson group additivity methods along with experimental data corrections to automatically determine the properties of the species

of interest (Benson & Cohen, 1993) (Gani, 1994). Pure component thermodynamic properties are necessary to calculate the Gibbs free energy of reaction and the enthalpy of reaction. To calculate the enthalpy of reaction, the species enthalpies must first be calculated. The species enthalpies are calculated from the constant-pressure heat capacity. To describe the constant pressure heat capacity a polynomial equation (3.3.5-17) is shown below:

$$C_{P,i} = a_i + b_i T + c_i T^2 + d_i T^3 \quad (3.3.5-17)$$

The property database contains the values for a , b , c , and d for each species. The property database also has the standard state heat of formation at 298 K. The constant-pressure heat capacity and the enthalpy of formation are used in the following equation (3.3.5-18) to find the enthalpy of species i at the reactor temperature T .

$$\underline{H}(T)_i = \underline{H}(T_0 = 298K)_i + \int_{T_0=298K}^T C_{P,i}(T) dT \quad (3.3.5-18)$$

Inserting equation (3.3.5-17) into equation (3.3.5-18) and integrating from T_0 to T produces the following definition for the enthalpy of a species i , (3.3.5-19) at temperature T .

$$\underline{H}(T)_i = \underline{H}(T_0 = 298K)_i + \left[aT + \frac{b}{2} T^2 + \frac{c}{3} T^3 \right]_{T_0=298K}^T \quad (3.3.5-19)$$

By using Hess's Law, the species enthalpies and the stoichiometry can be utilized to calculate the enthalpy of reaction at the reactor temperature T , as illustrated in the

equation (3.3.5-20) where j is the reaction index and i iterates over the species in the reaction.

$$\Delta H_{rxn}(T)_j = \sum_i^{Species} v_{ij} \underline{H}(T)_i \quad (3.3.5-20)$$

Once the enthalpies of reaction are calculated the Gibbs free energy of reaction at a temperature T can be formulated using the van't Hoff equation (3.3.5-21) below:

$$\ln \frac{K_{eq}(T)}{K_{eq}(T_0 = 298K)} = \int_{T_0=298K}^T \frac{\Delta H_{rxn}(T)}{RT^2} dT \quad (3.3.5-21)$$

Inserting the definition of the equilibrium constant based on Gibbs free energy of reaction leads to equation (3.3.3-10):

$$\begin{aligned} \ln \frac{K_{eq}(T)}{K_{eq}(T_0 = 298K)} &= -\frac{\Delta G_{rxn}(T)}{RT} + \frac{\Delta G_{rxn}(T_0)}{RT_0} \\ &= \int_{T_0=298K}^T \frac{\Delta H_{rxn}(T)}{RT^2} \end{aligned} \quad (3.3.5-22)$$

Utilizing Hess's law derivations for the enthalpy and the Gibbs free energy of reaction leads to the equation (3.3.5-23) below:

$$\begin{aligned} \ln \frac{K_{eq}(T)_j}{K_{eq}(T_0 = 298K)_j} &= -\frac{\sum_i^{Species} v_{ij} \underline{G}(T)_i}{RT} + \frac{\sum_i^{Species} v_{ij} \underline{G}(T_0)_i}{RT_0} \\ &= \int_{T_0=298K}^T \frac{\sum_i^{Species} v_{ij} \underline{H}(T)_i}{RT^2} \end{aligned} \quad (3.3.5-23)$$

In the DMB framework the implementation of the van't Hoff equation (3.3.5-23) is calculated on a component basis to decrease the need for multiple nested loops. The Gibbs free energy for each species i must be calculated at a temperature T by using a

reorganized pure component arrangement of the van't Hoff equation (3.3.5-24)

illustrated below:

$$-\frac{\underline{G}(T)_i}{RT} + \frac{\underline{G}(T_0)_i}{RT_0} = \int_{T_0=298K}^T \frac{\underline{H}(T)_i}{RT^2} \quad (3.3.5-24)$$

Solving for the pure component Gibbs free energy at temperature T in equation (3.3.5-24) leads to equation (3.3.5-25) below:

$$\underline{G}(T)_i = RT \left(\frac{\underline{G}(T_0)_i}{RT_0} - \int_{T_0=298K}^T \frac{\underline{H}(T)_i}{RT^2} \right) \quad (3.3.5-25)$$

Once the species Gibbs free energy have been calculated they can be used with the stoichiometry to calculate the Gibbs free energy of reaction at a temperature T . In the equation (3.3.5-26) below j is the reaction index and i iterates over the species in the reaction.

$$\Delta G_{rxn}(T)_j = \sum_i^{Species} v_{ij} \underline{G}(T)_i \quad (3.3.5-26)$$

The formulations shown are utilized to calculate the Gibbs free energy and enthalpy of reactions for each reaction. These thermodynamic quantities are necessary for the rate law and LFER expressions. They are also necessary to calculate the energy balance of non-isothermal system which will be detailed in the next section.

3.3.6 Energy Balance

A non-isothermal reactor has a temperature variation; a molecular model must be able predict the temperature changes. To determine the change in temperature an

energy balance on the reactor system must be solved. This energy balance is in the form of an additional ODE that tracks the temperature in the reactor. The DMB framework is currently only designed to handle adiabatic reactors, meaning the energy balance does not contain additional heat to the system. The framework also does not consider pressure deviation. The energy balances are also for incompressible fluids at constant pressure. For a batch reactor the energy balance is shown in the equation (3.3.6-27) below:

$$\frac{dT}{dt} = \frac{-\sum r_i \Delta H_{Rxn,i}}{\sum C_j C_{Pj}} \quad (3.3.6-27)$$

Where i iterates over the number of reactions and j iterates over the species. The numerator is the summation of the product between the rate law and the enthalpy of reaction for each species i . The denominator is the summation of the product between concentrations and the heat capacity of each species j . Equation (3.3.6-27) can also be used to describe the energy balance of CSTR without any modifications.

The energy balance in a PFR system can be described by equation (3.3.6-28). Equation (3.3.6-28) can be utilized to describe the temperature change across the length of the reactor with a constant cross sectional area. The batch, CSTR, and PFR energy balances are very similar only differing in the usage of concentration vs. molar flow and different integration dimensions.

$$\frac{dT}{dz} = A_c \frac{-\sum r_i \Delta H_{Rxn,i}}{\sum F_j C_{Pj}} \quad (3.3.6-28)$$

3.3.7 Numerical Solutions to Ordinary Differential Equations (ODEs)

The mole and energy balances that describe chemical processes require solutions to first order ordinary differential equations (ODEs). In the DMB framework numerical integration methods are employed to approximate first order ODE solutions. To describe the numerical integration utilized in the DMB framework a generic ODE primer will be covered. A generic ODE equation is shown in equation (3.3.7-29), which illustrates the derivative of y with respect to t equal to a function f of t and y .

$$\frac{dy}{dt} = f(t, y(t)) \quad (3.3.7-29)$$

Numerical approximations to derivative can be made by utilizing finite differences. There are three finite differences that can be used to approximate a derivative: forward, central, and backward. The forward finite difference is shown below as a derivative approximation and has a time step h . As seen in equation 29 the limit is below as the time step approaches zero the approximation is equal to derivative, this is the general definition of a derivative. The derivative can be approximated as equivalent to the forward finite difference shown in equation (3.3.7-30) below.

$$\frac{dy}{dt} = \lim_{h \rightarrow 0} \frac{y(t+h) - y(t)}{h} \approx \frac{y(t+h) - y(t)}{h} \quad (3.3.7-30)$$

Substituting the forward finite difference approximation for a derivative into the generic ODE equation leads to the following equation (3.3.7-31).

$$\frac{y(t+h) - y(t)}{h} = f(t, y(t)) \quad (3.3.7-31)$$

Equation (3.3.7-31) can be rearranged and defined recursively as shown in equation (3.3.7-32):

$$y_{n+1} = y_n + hf(t_n, y_n) \quad (3.3.7-32)$$

For each species in a chemical process an ODE needs to be numerically integrated to interpret the dynamics. This means that equation (3.3.7-32) is the addition of column vectors where each row represents a species. The developed equation above is commonly referred to as the explicit forward Euler method. The forward Euler method has a smaller numerical stability region and it is not typically used for numerical integration of chemical processes. Parameters in chemical processes can vary in large order of magnitudes causing numerical instability. Instead a method with a higher numerical stability is preferred. The backward finite approximation of a derivative is more stable and now will be formulated. Utilizing the same approach to calculate the forward Euler method, the backward finite differences approximation (3.3.7-33) can also be utilized to calculate the derivative.

$$\frac{dy}{dt} = \lim_{h \rightarrow 0} \frac{y(t) - y(t-h)}{h} \approx \frac{y(t) - y(t-h)}{h} \quad (3.3.7-33)$$

Setting the backward finite difference ODE approximation (3.3.7-33) to the first generic ODE equation (3.3.7-29) develops to the equation (3.3.7-34) below.

$$\frac{y(t) - y(t-h)}{h} = f(t, y(t)) \quad (3.3.7-34)$$

Equation (3.3.7-34) can also be rearranged and defined as the recursive scheme shown in equation (3.3.7-35) known as the Backward Euler method:

$$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}) \quad (3.3.7-35)$$

The Backward Euler method involves an implicit equation where y_{n+1} is on the left and right hand sides of the equation. The y_{n+1} on the right hand side of the equation needs to be approximated to calculate the y_{n+1} on the left hand side. In the DMB framework an approximation to y_{n+1} is calculated by using the Newton-Raphson method whose equation (3.3.7-36) is shown below. Newton-Raphson method is typically utilized to find roots of functions using the known initial conditions y_n and t_n . The Newton-Raphson equation requires a derivative to approximate the root. As mentioned earlier the defined recursive schemes are for multi-component systems and are composed of column vectors. Therefore, the derivative is a matrix of first order partial derivatives of vector valued function f known as a Jacobian. Multi-reactions systems may not have analytical solutions to their derivatives therefore a numerical approximation must be used to approximate the Jacobian matrix. The Jacobian can be numerically approximated using finite differences.

$$y_{n+1} = y_n - \frac{f(t_n, y_n)}{J(t_n, y_n)} = y_n - J(t_n, y_n)^{-1} * f(t_n, y_n) = y_n + H \quad (3.3.7-36)$$

To solve the Newton-Raphson method (3.3.7-36) a system of equations must be solved involving the Jacobian matrix and the vector column f .

$$J(t_n, y_n)H = -f(t_n, y_n) \quad (3.3.7-37)$$

Equation (3.3.7-37) above is equivalent to the matrix form $Ax = B$ and can be solved using linear algebra methods. Typically, lower-upper (LU) decomposition is used to solve for the H column vector. LU decomposition can be viewed as the matrix form of Gaussian elimination. Once the system of equations is solved, the H column vector can be used to approximate the y_{n+1} column vector.

The Newton-Raphson method must be calculated at every time step adding an overhead to the implicit numerical integration method. This added overhead comes with an advantage since the Backward Euler method has a much larger stability region than its explicit counterpart. In fact, this method is L-stable making it suitable for problems that encounter numerical instability which is the case with chemical kinetics.

The forward and backward Euler methods are examples of single step numerical integration methods. This means that the numerical method only refers to the previous step to take the next time step. The DMB framework employs an ODE solver with a linear multi step method which utilizes the information from the previous steps to calculate the next time step. A linear multistep method uses a linear combination of y_n and $f(t_n, y_n)$ to calculate the value of y for the desired current step. The linear multi step methods used in the DMB framework are the backward differentiation formulas (BDF). The BDF methods have six stable order equations four of which are shown in equations (3.3.7-38), (3.3.7-39), (3.3.7-40), and (3.3.7-41):

$$\text{Order 1: } y_{n+1} - y_n = hf(t_{n+1}, y_{n+1}) \text{ (Backward Euler)} \quad (3.3.7-38)$$

$$\text{Order 2: } y_{n+2} - \frac{4}{3}y_{n+1} + \frac{1}{3}y_n = \frac{2}{3}hf(t_{n+2}, y_{n+2}) \quad (3.3.7-39)$$

$$\text{Order 3: } y_{n+3} - \frac{18}{11}y_{n+2} + \frac{19}{11}y_{n+1} - \frac{2}{11}y_n = \frac{6}{11}hf(t_{n+3}, y_{n+3}) \quad (3.3.7-40)$$

$$\begin{aligned} \text{Order 4: } y_{n+4} - \frac{48}{25}y_{n+3} + \frac{36}{25}y_{n+2} - \frac{16}{25}y_{n+1} + \frac{3}{25}y_n \\ = \frac{12}{25}hf(t_{n+4}, y_{n+4}) \end{aligned} \quad (3.3.7-41)$$

The ODE solver used in the DMB framework is the C-based Variable Order Differential Equation (CVODE) solver from Lawrence Livermore National Labs (LLNL) (Hindmarsh, 2005). The CVODE solver utilizes a variable order algorithm to increase accuracy while taking fewer steps. As the method order increases the method stability decreases. The integration of the ODE solver with the DMB framework will be discussed in the following sections.

3.4 Dynamic Model Builder Program

Up to this point the theory behind the new kinetic modeling environment has been covered (See Section 3.3). The following will describe the integration of first principles and computer science to achieve a robust molecular-level modeling environment. To aid in the understanding of the framework implementation, the program workflow, framework design, differences from the previous framework, and technological advances will be presented. A specific focus will be placed on illustrating the computer programming aspects realized in the new framework.

The DMB modeling environment utilizes C++ object-oriented programming to organize data. Through the utilization of C++ classes relevant model information is stored in computer memory. This object-orientated design allows for a singleton class

to share a memory structure across all member functions in the program code. The model class in the DMB framework serves three main purposes:

1. Organize data from model configuration files
2. Perform calculations
3. Interface between simulation and the metaheuristic methods

The benefits of a class design are that it allows for rapid development, easy maintenance, and code reusability. The model class is initialized globally every time a DMB application is executed. It is an integral component of the DMB framework that allows for code simplification and model data access/storage necessary for molecular-level modeling. A simplified model class example is shown in Figure 7 for a data structure with two public variables and two functions for parsing reactions and calculating the enthalpy of reaction.

```
class Model
{
public:
    int NumberOfReactions;
    int NumberOfReactionFamilies;
    Model ();
    void ParseReactions ();
private:
    void CalculateEnthalpyOfReaction ();
};
```

Figure 7: Simplified Model Class

To simulate the kinetics of chemical processes a set of ODEs must be solved. Thus, numerical approximations to the solutions of ODEs must be calculated in functions harnessed by the model class. The DMB framework utilizes an open source numerical integration library named C-based Variable Order Differential Equation

(CVODE) solver. The CVODE solver is part of the Suite of Nonlinear and Differential/Algebraic Equation Solvers (SUNDIALS) software tools developed by the computational department at Lawrence Livermore National Labs (LLNL) (Hindmarsh, 2005). The CVODE solver is a state of the art ODE solver utilized by industry and academia alike. The CVODE solver includes implicit and explicit numerical integration methods. Implicit methods are employed in the DMB framework to handle the numerical instability that can propagate through chemical systems. This instability is primarily due to large differences in the parameters that describe the mole balance ODEs and is commonly referred to as mathematical stiffness. When simulating the kinetics of chemical processes, a highly stable solution method is preferred i.e. implicit methods. The implicit linear multi-step method utilized in the DMB framework is the backward differentiation formula (BDF) which is L-stable at low orders making it suitable for stiff problems. The connection between the model class and CVODE library allows for reactor mole balances to be solved thus simulating the reactor kinetics. The connectivity between the CVODE solver and model class is illustrated by Figure 8.

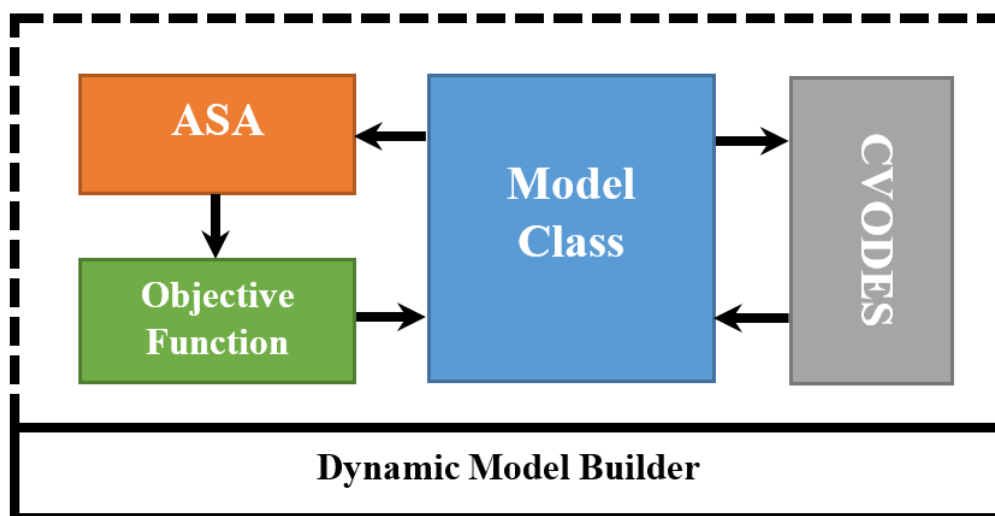


Figure 8: DMB Framework Design

The model class also has a link to the kinetic parameter estimation metaheuristic. To achieve kinetic parameter estimation, Adaptive Simulated Annealing (ASA) is utilized in combination with the model class and the CVODE solver. ASA is a metaheuristic method developed at the California Institute of Technology for global optimum/minima approximation. ASA is a variant of simulated annealing (SA) differing in that the parameters that control the temperature schedule and random step selection are automatically adjusted according to progress (Ingber, 1995). Kinetic parameter estimation can be very time consuming during model development; hence utilizing a robust metaheuristic like ASA is necessary to find an adequate solution space in timely manner. The interfaces between the model class, CVODEs, and ASA are shown in Figure 8, and together make strong tool to perform the kinetic parameter estimation of chemical processes.

Communication across the framework is critical when connecting various methods and libraries across one platform. This is especially true when all three components work together to achieve kinetic parameter estimation. ASA was designed

to approximate the minimum of an objective function without discontinuity. In the case of chemical kinetics, the objective function may not have a solution because of simulation failure. Simulation failure occurs when an integration method reaches unstable regions and cannot find a solution. As mentioned earlier, instability can be caused by large differences in order of magnitude between the kinetic parameters, concentrations, and or temperatures. When performing kinetic parameter estimation, parameters are perturbed; a certain combination may lead to instability and in turn simulation failure. Normally if a failure of this type occurs during an ASA objective function routine, the heuristic method may misinterpret the failure. In the DMB framework a simulation failure is communicated to ASA via the model class. This allows the heuristic method to navigate through complex solution spaces to find an acceptable solution space. This is type of smart design is a critical when performing kinetic parameter estimation and illustrates the framework strengths. A program workflow will now describe how all these methods work together to model a chemical process.

3.5 Program Workflow

To describe how a DMB application performs a simulation or kinetic parameter estimation a program workflow will now be defined in detail. This analysis will help describe how the framework works and a more in depth analysis on how it differentiates itself from the previous kinetic modeling environment. To begin the three main actions performed by the DMB framework during simulation are described:

1. Parse and organize information from model configuration files
2. Pre-calculate variables for a simulation
3. Solve ODEs

These three actions are critical in the program and will be discussed in this order, first discussing the parsing and organization followed by calculations and finally the solving of the ODEs.

3.5.1 Parsing and Organizing Model Data

The first file parsed by the DMB framework is the control file (Control.dat). The control file is a setting file that indicates the actions to be performed by the framework. This file contains the commands to perform either a simulation or kinetic parameter estimation. It also holds other important settings that are covered in more detail on Page 94. Once all the settings have been stored in the framework, the model class is initialized. This is done via a class constructor that defines the file paths for the modeling framework. These relative file paths search for the Model folder in the same directory as the DMB application. If the model folder cannot be found, DMB notifies the user and exits.

If the model folder is found by the DMB application, a function is called to read and parse the reaction network file. An example reaction network is presented below in Table 4 and will be utilized to describe the program workflow. Two levels of information are extracted during the parsing of the reaction network file. The first level is less detailed, storing the reactions and reaction families in separate character arrays. From these two character arrays the number of reactions and reactions families can be gathered and stored as integers within the model class. Once the number of reactions is known, important arrays are initialized. One such example of an important array is the rate law expression array that is used by the ODE solver routine and is the size of the number of reactions.

Table 4: Example Reaction Network

Index	Reactions	Reaction Families
0	$2\text{Species1} \rightarrow \text{Species2}$	RF1
1	$\text{Species3} \leftrightarrow \text{Species4}$	RF2

The second level of information is at finer detail and is extracted from each reaction. The stored reaction character arrays are parsed using regular expression (RegEx), separating variables by key reaction character types such as (+, ->, and <->). Through RegEx the reaction character array can be separated into reactant/product names, reactant/product orders, and the stoichiometry. Reactant and product names are stored in character arrays and reactant/product orders and stoichiometry are stored in double arrays. The reactant and product names are merged into one species list and a distinct species list is found algorithmically. The distinct species list is sorted, set, and used in the rest of the program to track molecular species within the model class. Table 5 illustrates the information that can be obtained from the second level of extraction on the reaction network from Table 4. Notice from the first two columns in Table 5 once a unique species list developed, a numerical index for each species can be established. This index makes it easy to track and sort species information across the framework. Immediately after this order is determined it is printed to a species mapping file (SolutionSpeciesMap.dat). The species map file lists the species in the order used in the DMB framework. For each species a storage container is created which holds the species index, the stoichiometry, and the order of each species for each reaction. This information is critical when creating a rate laws and solving mole balances pragmatically.

Table 5: Extracted Reaction Level Data

Species	Reaction 0	Reaction 1
----------------	-------------------	-------------------

List	Index	Stoichiometry	Order	Stoichiometry	Order
Species1	0	-2	2	0	0
Species2	1	1	1	0	0
Species3	2	0	0	-1	1
Species4	3	0	0	1	1

After the reaction network is deconstructed and its information stored in the model class, the experimental data such as the reactor inputs/outputs and conditions are loaded into the DMB framework. To DMB all this information falls under the category of dataset information. Where each dataset can be thought as an experiment: ten experiments equal ten datasets. The reactor inputs are the initial conditions (Initial_Condition_y.dat) to the initial value problem to be numerically integrated. The units of the initial conditions depend on the reactor being used: batch and CSTR have units of moles per liter and PFR has units of moles per second. In the same folder where the initial conditions are found, the differentiation length file is present. The units of this file depend on the reactor type chosen; batch and CSTR have units of seconds, while PFR has units of decimeters. The model reactor output is the information an experimentalist can provide based on their experimental output. This information can be measured species and/or bulk mixture properties. The user can load either or both into the framework. This data is stored by Dataset N (where N is the dataset index) where for each dataset there can be a set of observed species and mixture properties loaded in DMB. Once available in the framework they can be used as part of the objective function metric that describes the kinetic parameter performance. For each experiment there is also a set of reactor conditions (Reactor.dat) like reactor temperature, pressure, and dimensions.

Following the storage of dataset information in the model class, the species properties (proplist.dat) is read into the framework. The species properties allow for the calculation of thermodynamic and mixture properties pre, mid, and post simulation. Once the species properties have been read and stored, other information is read based on user settings. In the case where the system requires catalytic chemistry, LHHW is used. In the Model directory there is a directory named Chemistry which contains a directory called LHHW. The LHHW directory contains files which hold the number of sites, reactions that occur on sites, adsorption constant definitions/parameters, and hydrogen dependence. If the kinetic parameter estimation option has been chosen, the files from Optimization folder are also read by the framework. In the Optimization directory there is a directory called ASA. All the files necessary to run a kinetic parameter estimation using the ASA metaheuristic are stored in the ASA folder. The type of files found there have the following information: initial conditions, upper and lower bounds, tuning handle controls and kinetic parameter estimation results.

It should be noted that the species property file utilized by the DMB framework is created by the Property Database (PD). The PD is part of the Kinetic Modeling Toolkit (KMT) suite and is a database of quantitative structure property correlations for molecules calculated from Gani and Benson group contribution methods (Benson & Cohen, 1993) (Gani, 1994).

3.5.2 Pre-calculations

After all the necessary model information is stored in the model class, pre-calculations are performed to decrease compute time during simulations. The precomputation begins by first calculating the enthalpy and Gibbs free energy for each

species at the reactor temperature. Once the thermodynamic properties for each species are calculated the enthalpy and Gibbs free energy of reaction at the reactor temperature is calculated by using the stoichiometry matrix in equation (3.3.2-6). After which the thermodynamic Hess's Law equations (3.3.5-20) and (3.3.5-26) are utilized to calculate the enthalpy and Gibbs free energy of reaction. Once the enthalpy and the Gibbs free energy of reaction are calculated, the equilibrium constants are calculated via the Gibbs free energy of reaction and the reactor temperature as shown in the classical equation (3.3.3-10).

Following the calculation and storage of the equilibrium constants, the rate constants are calculated for each reaction. In an isothermal system these calculations are considered pre-calculations since they only need to be prepared once before the numerical integration begins. If a change of temperature occurs, then the Arrhenius equation (3.3.3-8) needs to be recalculated. To lower the compute time, the Arrhenius equation variables are always pre-calculated and not recalculated unless deemed necessary. This allows for less work to be performed in the integration function utilized to solve ODEs decreasing simulation time.

3.5.3 Solving Ordinary Differential Equations

After the equilibrium and rate constants are calculated and stored in the model class, the program is ready to simulate the chemical kinetics of reactors. The DMB framework elucidates the kinetics of chemical processes deterministically. The mole balances that describe the reactors in the framework illustrated by equations (3.3.1-3), (3.3.1-4), and (3.3.1-5) are all first order ODEs and must be numerically integrated. The DMB only considers numerical integration for initial value problems (IVP) of first order ODEs. Linear multi-step BDF methods (described in 3.3.7) are utilized for

numerical approximations of ODEs. The CVODE solver requires a function of ODEs in the format illustrated in Figure 9. The ODE function in Figure 9 has three inputs: time step (t), concentration array (y), and first order ODE array ($ydot$) which are all necessary for numerical integration. The objective of the ODE function is to calculate the $ydot$ array. The approach utilized to define ODEs differentiates the DMB framework from the previous environments in the Klein Research Group. To illustrate the differences between DMB and the previous system it helps to look at the both approaches, first starting with the former explicit approach shown in Figure 9.

```
int ODEfunction (double t, double[] y, double[] ydot)
{
    // Rate Laws
    ratelaw[0] = k[0]*y[0]*y[0];
    ratelaw[1] = k[1]*(y[2] - y[3]/Keq[1]);

    // Mole Balances
    ydot[0] = -2 * ratelaw[0];
    ydot[1] = ratelaw[0];
    ydot[2] = -1 * ratelaw[1];
    ydot[3] = ratelaw[1];

    return 0;
}
```

Figure 9: Explicit ODE Function for system with 4 species with 2 reactions (Rxn1: Irreversible, Rxn2: Reversible)

Figure 9 demonstrates the ODE code necessary to describe the reaction network in Table 4. For the purposes of this example, these reactions will be modeled as a batch reactor. The ODE code in Figure 9 explicitly defines each rate law for reactions and each species mole balance. The rate laws are calculated first and are a function of the concentration array (y), rate constant (k), and equilibrium constant

(K_{eq}). The mole balances are calculated afterwards by using the stoichiometry information and the previously calculated rate laws. This approach is a classical way of defining ODEs for the CVODE solver to perform numerical integration. When the CVODE solver performs numerical integration on the ODE function in Figure 9 the model is reaction network dependent. This means that any changes to the reaction network require a rebuild of the code in the ODE function. This is fine for small systems but as system sizes grow larger, code editing can become time consuming and error prone. Once the code in the ODE function is changed the code needs to be recompiled, further requiring the user to have access to compilers and programming experience. This issue is further exasperated because it has been observed that as system sizes grow to mega sizes compilation may not be possible due to the large ODE file sizes. Modern compilers are not designed to handle code files with lengths on the order of hundred thousand lines of code which can be easily reached for large reaction networks. Therefore, the goal of this work is to create a dynamic framework that can be utilized to model large and small systems alike without any dependencies or issues besides the customary mathematical ones. A follow up to the example in Figure 9 is the DMB version shown in Figure 13 where instead of having rate laws explicitly written, DMB utilizes two sets of nested for loops to describe the rate laws and mole balances. For loops are segmented code that is controlled iteratively and executed repeatedly.

To familiarize the idea that rate law can be developed numerically by nested loops it helps to separate the rate laws into two portions as shown in Figure 10, the irreversible and reversible portions.

$$r = k_C \left(\overbrace{C_A^a C_B^b}^{\text{Irreversible}} - \frac{\overbrace{C_C^c C_D^d}^{\text{Reversible}}}{K_{eq}} \right)$$

Figure 10: Rate Law Portions

Figure 10 illustrates a reversible microkinetic rate law for the reactions of $aA + bB \rightarrow cC + dD$. Notice the irreversible and reversible rate law portions can be built by multiplication of concentrations. Mathematically the irreversible and reversible portions can be defined for any microkinetic rate law by the equations (3.5.3-42) and (3.5.3-43) below:

$$\text{Irreversible Portion} = \prod_{i=0}^{\text{Reactants}} C_i^{\text{Order}_i} \quad (3.5.3-42)$$

$$\text{Reversible Portion} = \frac{1}{K_{eq}} \prod_{i=0}^{\text{Products}} C_i^{\text{Order}_i} \quad (3.5.3-43)$$

As shown in the mathematical equations (3.5.3-42) and (3.5.3-43) above, the irreversible portion can be calculated by multiplying the concentrations raised to the power of the order for each species i in the reaction. If the reaction is reversible the same approach can be applied to the products to calculate the reversible portion. The difference between the two portions is that after the reversible portion is calculated it is divided by the equilibrium constant of the reaction. Understanding that the rate law can be split into portions, calculated in pieces, and assembled into a numerical rate law value for each reaction is vital to comprehend the approach implemented in DMB. Equation (3.5.3-44) illustrates the general portioned rate law.

$$r = \text{Rate Constant} * (\text{Irreversible} - \text{Reversible}) \quad (3.5.3-44)$$

The approach of developing rate laws in portions can be extended into the pseudocode and presented in Figure 13. To calculate these irreversible and reversible portions, the code presented in Figure 13 utilizes a data class called Rxns to hold the information from Table 5 (Info on data classes see section 3.4). The Rxns class variables and functions can be accessed by the dot operator (“.”). The Rxns data class contains species indices and orders for the irreversible and reversible portions. By utilizing the class structure the data is retrieved via the dot operator. To calculate the rate laws of each reaction a set of nested loops are utilized. The outer loop, iterates over each reaction and the two inner loops iterate over the irreversible and reversible portions. The first inner nested loop calculates the irreversible portion, which loops over all the reactants in a reaction. The loop utilizes the Rxn data container to calculate the irreversible portion by multiplying the reactant concentration (y) raised to the power of the reactant order (Rxns.Order) in reaction i as shown in Figure 13. This same approach can apply to the reversible portion by a secondary inner nested loop. After the reversible portion concentration product is calculated it is divided by the equilibrium constant of the reaction. If the reaction is irreversible then the reversible portion is never calculated to decrease function overhead. At the end of the rate law loop, the calculated portions are combined to build the rate law by taking the difference of the portions and multiplying the difference by the reaction rate constant.

After the rate laws are calculated, the mole balances for each species are considered. The mole balance equations utilize the calculated rate laws with the stored stoichiometry matrix. In the example pseudocode illustrated by Figure 13 a batch

reactor mole balance is utilized. Mathematically, a general expression for mole balances is described by equation (3.3.2-6). Pragmatically, the batch reactor mole balance can be defined by a nested loop. The first loop iterates over the species and the second inner loop iterates over the dot product of the stoichiometry matrix and rate laws. This is illustrated in the second section of pseudocode in Figure 13 labeled mole balances. The row dimension of the stoichiometry matrix is equivalent to the number of species and the column dimension is equivalent to numbers of reactions. As the reaction networks sizes increase so does the size of the stoichiometry matrix. Large reaction networks have large stoichiometry matrices. For most large systems the stoichiometry matrix is sparse matrix where most matrix values are zero. Looping over a stoichiometry matrix with zeros can be computationally expensive. Therefore, the DMB implementation omits zeros by not storing them following compressed sparse row (CSR) format as shown in Figure 11.

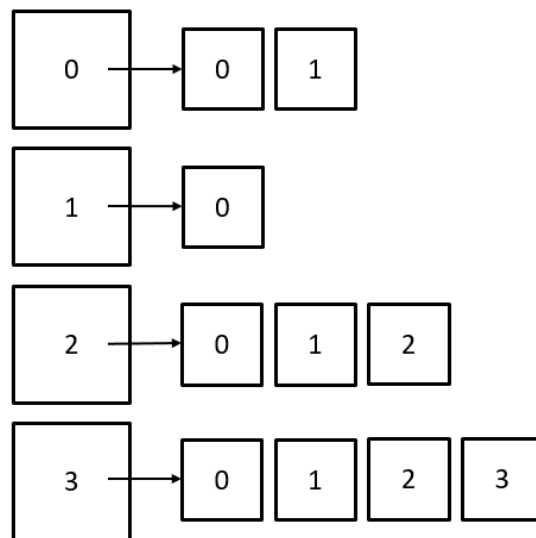


Figure 11: 2D Jagged Array

Jagged arrays are arrays of arrays as illustrated in Figure 11. In likening to a typical data matrix format, each row has a variable number of columns. This type of array is perfect for storing the non-zero values of a CSR stoichiometry matrix. The only issue with using a jagged array to store the stoichiometry matrix values is that the index for each reaction is lost. To counter this problem instead of just storing the stoichiometry value in the jagged array a pair of information is stored: reaction index and stoichiometry. This allows for the use of one array to store the necessary information. Figure 12 shows the implementation paired information from Table 5. Notice that for each species (row) there is a paired data container for each reaction it is involved in. For the example system in Table 4 this approach converted 4×2 stoichiometry matrix into a 4×1 array. All this information together is used to define the ODE file implicitly as shown in Figure 13. In this work an implicit ODE is implied but not plainly expressed as the explicit version.

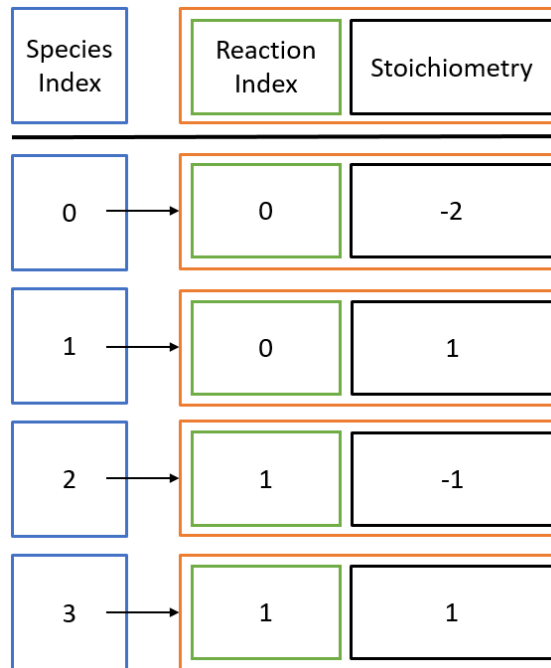


Figure 12: Jagged Array Stoichiometry Implementation with Paired Data

The pseudocode in Figure 13 can be utilized as a generic description to describe the rate laws and mole balances of any microkinetic system. Describing the rate laws and mole balances by nested loops allows for a static ODE function which is independent of the reaction network. This translates to a modeling application that does not require code recompilation. Therefore, models can be created and run by just using the DMB application. Model file size is also no longer an issue since the models reside in memory bypassing the compilation of large files. The pseudocode presented here is for a microkinetic system, the DMB framework further extends these ideas for power laws and catalytic chemistry which will be described in the later sections.

```

int function (double t, double[] y, double[] ydot)
{
    // Rate Law Loop
    double [] ratelaw;
    for (Rxns → Rxns.Total)
    {
        double Irreversible = 1;
        double Reversible = 1;
        // Inner Loop: Irreversible Portion
        for (Rxns.Species → Rxns.Reactant.Species.Num)
        {
            Irreversible *= pow(y[Rxns.Species.Index],
                Rxns.Species.Order);
        }
        // Inner Loop: Reversible Portion
        for (Rxns.Species → Rxn.Product.Species.Num)
        {
            Reversible *= pow(y[Rxns.Species.Index],
                Rxns.Species.Order);
        }
        Reversible /= Keq[Rxns.Index];
        ratelaw [Rxns.Index] = k[0] * (Irreversible – Reversible);
    }

    // Mole Balances
    for (Stoichiometry → Species.Total)
    {
        for (Stoichmetry → Stoichmetry.RxnTotal)
            ydot[Stoichiometry.SpeciesIndex] +=
                Stoichiometry.Value*ratelaw[Stoichiometry.RxnIndex];
    }
}

```

Figure 13: Implicit ODE Function

3.6 DMB Model Independent/Dependent Framework

Up to this point DMB has been discussed as a reaction network independent framework where ODEs are built by nested loops. In this approach the model information resides in the memory and is developed implicitly during run time. The DMB framework is designed to be versatile so it can revert to a reaction network

dependent framework with explicit ODEs if need be. A model dependent framework can be useful when the user would like direct to access rate laws and mole balance in the code. This allows for more control and the ability to make user specific changes or alterations that fit needs.

To switch from a model independent framework to a model dependent framework an existing model must be loaded in the DMB framework via the traditional route of model configurations files. To switch, the DMB framework will write a set of representative ODEs to a file based on a loaded model configuration. The outputted file can then be integrated into the DMB framework by placing it in the DMB source code. To complete the integration, all that needs to be done is project recompilation. This is designed to be an easy process where one file can be utilized to alter the framework to a reaction network dependent one. Once the ODE file is compiled into the application, one switch is utilized to transfer between the explicit set of ODEs and the static ones. To do this the ExeType is changed from DMB to Model in the control file. Once ExeType is changed to Model, the DMB application will proceed to use the explicit ODE file. Following these changes, the framework becomes reaction network dependent. The explicit ODEs can be utilized for both simulations and kinetic parameter estimations. Changing the reaction network means that a new set of explicit ODEs have to be written by the framework and reintegrated into the system via recompilation.

The explicit ODE file produced is designed to be easy to read and understand. It utilizes objected-oriented programming variables from the model class. The model class variables employed are appropriately named to match the variables they represent. Additional code can be added to the ODE file based on the user preference.

The additional code can implement data containers that exist in the model class or add new ones in the ODE function. A snippet of an explicit ODE example code is shown in the Figure 14 and Figure 15 for rate laws and mole balances in LHHW system.

```
rxns.ratelaws[23]=rxns.RateConstant[23]*0.010000*rxns.Kadsorption[118][0]*(Fi[111]*Fi[118])/rxns.AdsorptionGroup[0];
rxns.ratelaws[24]=rxns.RateConstant[24]*0.010000*rxns.Kadsorption[186][0]*(Fi[111]*Fi[186])/rxns.AdsorptionGroup[0];
rxns.ratelaws[25]=rxns.RateConstant[25]*0.010000*rxns.Kadsorption[191][0]*(Fi[111]*Fi[191])/rxns.AdsorptionGroup[0];
rxns.ratelaws[26]=rxns.RateConstant[26]*0.010000*rxns.Kadsorption[195][0]*(Fi[111]*Fi[195])/rxns.AdsorptionGroup[0];
rxns.ratelaws[27]=rxns.RateConstant[27]*0.010000*rxns.Kadsorption[203][0]*(Fi[111]*Fi[203])/rxns.AdsorptionGroup[0];
rxns.ratelaws[28]=rxns.RateConstant[28]*0.010000*rxns.Kadsorption[205][0]*(Fi[111]*Fi[205])/rxns.AdsorptionGroup[0];
rxns.ratelaws[29]=rxns.RateConstant[29]*0.010000*rxns.Kadsorption[207][0]*(Fi[111]*Fi[207])/rxns.AdsorptionGroup[0];
rxns.ratelaws[30]=rxns.RateConstant[30]*0.010000*rxns.Kadsorption[209][0]*(Fi[111]*Fi[209])/rxns.AdsorptionGroup[0];
rxns.ratelaws[31]=rxns.RateConstant[31]*0.010000*rxns.Kadsorption[214][0]*(Fi[111]*Fi[214])/rxns.AdsorptionGroup[0];
rxns.ratelaws[32]=rxns.RateConstant[32]*0.010000*rxns.Kadsorption[216][0]*(Fi[111]*Fi[216])/rxns.AdsorptionGroup[0];
rxns.ratelaws[33]=rxns.RateConstant[33]*0.010000*rxns.Kadsorption[218][0]*(Fi[111]*Fi[218])/rxns.AdsorptionGroup[0];
rxns.ratelaws[34]=rxns.RateConstant[34]*0.010000*rxns.Kadsorption[220][0]*(Fi[111]*Fi[220])/rxns.AdsorptionGroup[0];
rxns.ratelaws[35]=rxns.RateConstant[35]*0.010000*rxns.Kadsorption[226][0]*(Fi[111]*Fi[226])/rxns.AdsorptionGroup[0];
rxns.ratelaws[36]=rxns.RateConstant[36]*0.010000*rxns.Kadsorption[230][0]*(Fi[111]*Fi[230])/rxns.AdsorptionGroup[0];
rxns.ratelaws[37]=rxns.RateConstant[37]*0.010000*rxns.Kadsorption[232][0]*(Fi[111]*Fi[232])/rxns.AdsorptionGroup[0];
rxns.ratelaws[38]=rxns.RateConstant[38]*0.010000*rxns.Kadsorption[238][0]*(Fi[111]*Fi[238])/rxns.AdsorptionGroup[0];
rxns.ratelaws[39]=rxns.RateConstant[39]*0.010000*rxns.Kadsorption[240][0]*(Fi[111]*Fi[240])/rxns.AdsorptionGroup[0];
rxns.ratelaws[40]=rxns.RateConstant[40]*0.010000*rxns.Kadsorption[242][0]*(Fi[111]*Fi[242])/rxns.AdsorptionGroup[0];
rxns.ratelaws[41]=rxns.RateConstant[41]*0.010000*rxns.Kadsorption[244][0]*(Fi[111]*Fi[244])/rxns.AdsorptionGroup[0];
rxns.ratelaws[42]=rxns.RateConstant[42]*0.010000*rxns.Kadsorption[249][0]*(Fi[111]*Fi[249])/rxns.AdsorptionGroup[0];
rxns.ratelaws[43]=rxns.RateConstant[43]*0.010000*rxns.Kadsorption[251][0]*(Fi[111]*Fi[251])/rxns.AdsorptionGroup[0];
rxns.ratelaws[44]=rxns.RateConstant[44]*0.010000*rxns.Kadsorption[359][0]*(Fi[359]*Fi[111])/rxns.AdsorptionGroup[0];
rxns.ratelaws[45]=rxns.RateConstant[45]*0.010000*rxns.Kadsorption[223][0]*(Fi[223]*Fi[111])/rxns.AdsorptionGroup[0];
rxns.ratelaws[46]=rxns.RateConstant[46]*0.010000*rxns.Kadsorption[352][0]*(Fi[352]*Fi[111])/rxns.AdsorptionGroup[0];
rxns.ratelaws[47]=rxns.RateConstant[47]*0.010000*rxns.Kadsorption[375][0]*(Fi[375]*Fi[111])/rxns.AdsorptionGroup[0];
```

Figure 14: Explicitly Written Rate Laws Created by DMB Framework

```
// Mole Balances
Ith(ydot, 1) = rxns.gAc * ((1.000000)*rxns.ratelaws[578] + (-1.000000)*rxns.ratelaws[700] + (-1.000000)*rxns.ratelaws[775] + (-1.000000)*rxns.ratelaws[776]);
Ith(ydot, 2) = rxns.gAc * ((1.000000)*rxns.ratelaws[44] + (1.000000)*rxns.ratelaws[47] + (1.000000)*rxns.ratelaws[55] + (1.000000)*rxns.ratelaws[68]
+ (1.000000)*rxns.ratelaws[69] + (1.000000)*rxns.ratelaws[81]
+ (1.000000)*rxns.ratelaws[86] + (1.000000)*rxns.ratelaws[92] + (1.000000)*rxns.ratelaws[102] + (1.000000)*rxns.ratelaws[103] + (1.000000)*rxns.ratelaws[106]
+ (1.000000)*rxns.ratelaws[108] + (1.000000)*rxns.ratelaws[115] + (1.000000)*rxns.ratelaws[128] + (1.000000)*rxns.ratelaws[138] + (1.000000)*rxns.ratelaws[150]
+ (1.000000)*rxns.ratelaws[174] + (1.000000)*rxns.ratelaws[175] + (1.000000)*rxns.ratelaws[176] + (1.000000)*rxns.ratelaws[177] + (1.000000)*rxns.ratelaws[180]
+ (1.000000)*rxns.ratelaws[219] + (1.000000)*rxns.ratelaws[223] + (1.000000)*rxns.ratelaws[226] + (1.000000)*rxns.ratelaws[230] + (1.000000)*rxns.ratelaws[231]
+ (1.000000)*rxns.ratelaws[260] + (1.000000)*rxns.ratelaws[264] + (1.000000)*rxns.ratelaws[272] + (1.000000)*rxns.ratelaws[293] + (1.000000)*rxns.ratelaws[294]
+ (1.000000)*rxns.ratelaws[306] + (1.000000)*rxns.ratelaws[311] + (1.000000)*rxns.ratelaws[317] + (1.000000)*rxns.ratelaws[329] + (1.000000)*rxns.ratelaws[330]
+ (1.000000)*rxns.ratelaws[333] + (1.000000)*rxns.ratelaws[335] + (1.000000)*rxns.ratelaws[342] + (1.000000)*rxns.ratelaws[352] + (1.000000)*rxns.ratelaws[361]
+ (1.000000)*rxns.ratelaws[367] + (1.000000)*rxns.ratelaws[372] + (1.000000)*rxns.ratelaws[384] + (1.000000)*rxns.ratelaws[409] + (1.000000)*rxns.ratelaws[410]
+ (1.000000)*rxns.ratelaws[411] + (1.000000)*rxns.ratelaws[412] + (1.000000)*rxns.ratelaws[415] + (1.000000)*rxns.ratelaws[459] + (1.000000)*rxns.ratelaws[463]
+ (1.000000)*rxns.ratelaws[464] + (1.000000)*rxns.ratelaws[467] + (1.000000)*rxns.ratelaws[471] + (1.000000)*rxns.ratelaws[472] + (1.000000)*rxns.ratelaws[478]
+ (-1.000000)*rxns.ratelaws[578] + (-1.000000)*rxns.ratelaws[787] + (-1.000000)*rxns.ratelaws[788] + (1.000000)*rxns.ratelaws[855]);
```

Figure 15: Explicitly Written Mole Balances Created by the DMB Framework

It should be noted that performance can be affected by additional user code in the ODE file. The user should also be aware of code memory allocation in the ODE function as this surplus code may hamper performance. This being said testing has

shown explicit ODEs outperform implicit ODE counterparts in simulation speed by a tenth of second.

Not all models can be transferred to explicit versions and compiled by the C++ compiler in the Visual Studio integrated development environment. If the reaction network has reactions and species greater than on the order of 10^4 the additional ODE code may reach a compiler limit during compilation. There may be various reasons why a compiler may have issues, but it has been observed that file size, memory taken up defined variables, and optimization options pose the main issues. The scope of this work primarily focused on bypassing the issue through the implicit ODE approach. Code compiler changes or alterations could be a route to compile complex code but is out the space of this work. Currently the only route for simulation of very large networks is through the implicit ODE code in DMB.

3.7 Performance Differences from the Previous Framework

The DMB framework was designed to replace the Klein Research Group's previous kinetic modeling software tool KME. As mentioned in Section 3.1, KME is a model dependent framework where ODEs are written explicitly to files. DMB can act as both model independent and dependent making it extremely versatile. Whenever developing a replacement to an existing software tool a question is always asked: Is the new framework better than the previous? DMB can easily scale to larger reaction networks spaces where KME cannot. Therefore, the software tools can't be compared for very large reaction networks. Still the question remains, is the new framework better where both the old and the new can be compared? This is the small to medium size reaction network range. For a kinetic modeling environment, a key performance metric is simulation speed. Faster simulation speed can mean less time spent in the

time consuming steps like kinetic parameter estimation. This is especially true for large systems. So how do the frameworks compare? Well in a once through simulation of the same model, the KME model took ~50 seconds where the DMB framework took ~10 seconds when running in implicit mode. This is a performance gain of five times. This performance gain cannot be pinpointed to one difference between the frameworks but an agglomeration of all the differences. The differences between the modeling frameworks are development environment, operating system (OS), compilers, pre-calculations and design.

There are many reasons why DMB outperforms KME, a key one being the development environment. DMB was developed in the Microsoft Visual Studio (MSVS) integrated development environment (IDE) for Windows OS. An IDE is an application that facilitates software development. It typically includes a source code editor, build automation tools, and a debugger. The MSVS IDE allows the user to track and understand the code quickly. It provides variable visualizations as well as advanced error information. From a novice to an advanced programmer MSVS provides the tools necessary for the development of a robust program. Not to mention that MSVS also has code profiling tools that help find weak points in the code during execution. Understanding where the weak points lie in the code allows for code modifications to increase program performance. The profiler can also be used to track memory utilization as a function of time to determine memory leaks and manage the application heap memory size. KME was not developed in an IDE, making its development and debugging a difficult and ongoing task. KME has never been profiled to find the weak points in its code structure. The availability of an IDE

allowed for a more robust development of the DMB framework when compared to KME.

KMEs compilation and executables are built and run in open source UNIX-like environment named Cygwin. Cygwin is a virtual UNIX environment that operates on a Windows OS. This means that KMEs performance is tied into the Cygwin environment performance. Cygwin is managed by an open source community and is distributed from a variety of makers. There isn't really a standard for the Cygwin UNIX virtualization performance on Windows OS. Therefore, the fact that there is an intermediary between the OS and the KME model makes it difficult to determine if peak performance is achieved. DMB doesn't have this issue because the executable is created to function directly on the Windows OS. DMB is created on compilers that operate on Windows OS and take advantage of system optimizations to increase application performance.

As mentioned KME was designed to compile the code it creates in Cygwin to access the open source GNU compiler collection. The GNU compilers are free and are great to use when resources are not available. Since they are free their executables don't perform as fast as the non-free options. This is because the pay-to-use compilers have been highly optimized. In a UNIX environment the Intel and PGI compilers (not-free) surpass the performance of GCC (free). The MSVS IDE comes with the Visual C++ compiler. For the most part Visual C++ resembles the C++ standard except for availability of managed functions that help control memory usage. On the Windows OS the Visual C++ compiler performs better than the Cygwin virtualized version of a GCC compiler. DMB was also designed to operate in UNIX high performance cluster (HPC) environment. The performance between compilers and different OS has been

measured; the best performance achieved between compilers has the following order: Intel (HPC) > PGI (HPC) > Visual C++ (Windows) > GCC (Windows & HPC).

Compilers also have optimization options to increase the performance of the executables. Compiler optimizations use algorithms that take the program and transform it to a semantic equivalent output that utilizes fewer resources and increases performance. There are varying levels of optimization that can be applied to a program. Typically, the higher amounts of optimizations performed increase the performance of the application. These optimizations also increase the amount of memory used by the program. This is one of the reasons why large models cannot be compiled with a high amount of optimizations or if any. The amount of memory required to optimize large code files can exceed the available system memory. The optimizations options are the reason that the implicit ODE performs as well as the explicit versions. Current metrics show that the implicit ODE approach is only a tenth of second slower than the explicit counterpart in the DMB framework.

Pre-calculations in DMB framework are very important as they are performed to lessen the calculation overhead. This is especially important for calculations that occur in the heavily computed ODE function. The ODE function may be called by the ODE solver on the order of thousand times. When extraneous calculations are present in the ODE function this translates to a marked increase in simulation time. To decrease this issue DMB pre-calculates variables like rate constants and equilibrium constants. These variables only change when temperature is changed therefore there isn't a need to calculate them at every time step. Decreasing the amount of code in the ODE function decreases simulation time.

DMB was designed to be a robust framework. Substantial development time was spent profiling and examining performance metrics of the framework during development. During development specific data types were chosen due to their performance advantages. The most prominent approach is the use of jagged arrays which were implemented with performance in mind. Jagged arrays can have controlled memory allocations decreasing the memory allocation overhead during simulation. All these implementations together make DMB a strong framework that can be utilized to model large and small systems alike in a way that has never done before.

3.8 Multi-Phase Framework

Since a new framework was built, new functionalities were added that were not present in the previous modeling environments. Specifically, the multi-phase functionality which allows for the modeling of gas and liquid homogenous and heterogeneous mixtures. This new functionality was implemented by adding the capability to flash material streams in the DMB framework. It should be noted that this flash capability is a class object that can be applied in other code sections. This flash design object is extremely versatile and can be utilized to perform material flashes that do not involve multi-phase numerical integration. For example, the input or output streams can be flashed post simulation to match experimental designs.

To simulate a multiphase system, the flash method is placed in the numerical integration ODE solver function. At every time step the reactor stream is separated into gas and liquid phases. The gas and liquid streams are utilized to calculate corresponding phase rate laws and their contributions are added to a single overall rate

law utilized in the mole balance. The flash ODE solution scheme implemented in DMB is illustrated in Figure 16 below.

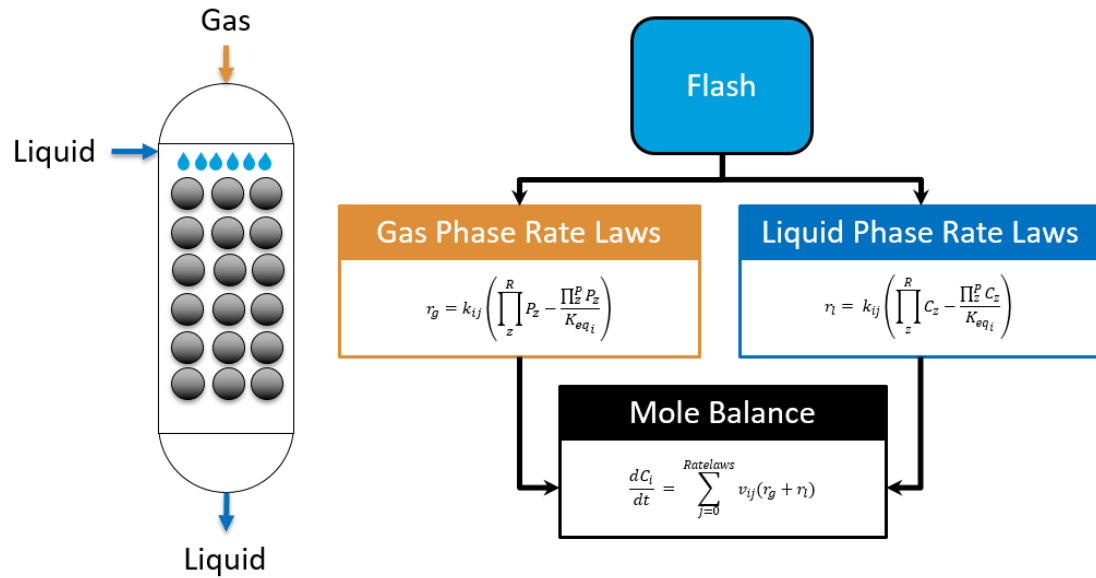


Figure 16: DMB Multi-Phase Approach

To achieve a multi-component distillation an equilibrium flash is utilized in the flash object. In the equilibrium flash the component separation is a function of pressure and temperature. An iterative process is necessary to find the vapor and liquid fraction of a multi-component system. To find the vapor fraction the root of the Rachford-Rice equation (3.8-45) below must be approximated.

$$\sum_i \frac{x_i(K_i - 1)}{1 + v_f(K_i - 1)} = 0 \quad (3.8-45)$$

The mole fraction of the liquid in the feed is known x_i , the equilibrium constant of component i can be approximated by Rault's Law as the division of saturated vapor pressure by the reactor pressure shown in equation (3.8-46).

$$K_i = \frac{P_{sat,i}}{P} \quad (3.8-46)$$

The vapor pressure of each species can be approximated by using the Lee-Kessler equations (3.8-47), (3.8-48), and (3.8-49) below which use the critical pressure (P_c), critical temperature (T_c), acentric factor (ω), system pressure (P), and system temperature (T).

$$\ln \frac{P_{sat}}{P_c} = f^0 + \omega f^1 \quad (3.8-47)$$

$$f^0 = 5.92714 + \frac{6.09648 * T_c}{T} - 1.28862 * \ln \frac{T}{T_c} + 0.169347 * \left(\frac{T}{T_c}\right)^6 \quad (3.8-48)$$

$$f^1 = 15.2518 + \frac{15.6875 * T_c}{T} - 13.4721 * \ln \frac{T}{T_c} + 0.43577 * \left(\frac{T}{T_c}\right)^6 \quad (3.8-49)$$

The only unknown in the Rachford-Rice equation (3.8-45) is the vapor fraction (v_f). To solve for it an iterative root finding method like Newton-Raphson bisection method is applied at every time step. The implementation of the new flash object in the numerical integration increases solution time substantially. The modeler should distinguish if this performance hit is necessary based on experimental information.

3.9 Parallel Computing

Creating a high performing kinetic modeling environment is necessary to replace the status quo. Several avenues have been investigated to increase the performance of the DMB framework, one of them being parallel computing. The advent of increasingly sophisticated computer hardware creates the momentum for

parallel computing implementations. Before exploring the parallel computing implementations, it is important to understand hardware computing options. A typical computing option is serial computing which is the processing of instructions sequentially one after another. Serial computing is executed on a single processor. The DMB framework has been discussed to this point as a serial compute program which functions on one processor or thread as shown in Figure 17.

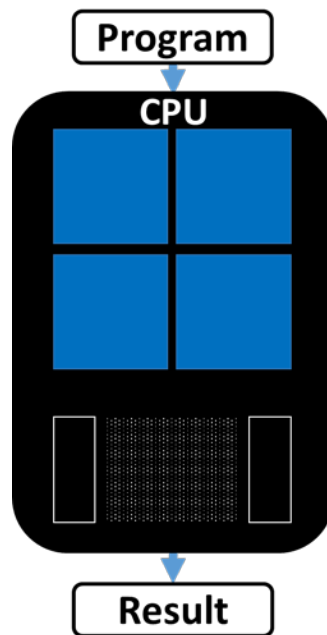


Figure 17: Program Processing on the CPU

Now parallel computing is idea that large tasks can be broken down into smaller discrete tasks which can be solved simultaneously. Parallel computing requires more than one processor. Generally, it is harder to code parallel algorithms because it introduces complex bugs and memory race conditions. This section will now introduce the hardware, reasoning, and the parallel algorithms utilized in DMB framework.

Parallel computing requires integration between computer hardware and code. In modern computers there are two areas where parallel computing can be accomplished, the central processing unit (CPU) and the graphical processing unit (GPU). The work described so far functions on a single processor on the CPU. State of the art CPUs have more than one processor; typically, 4 to 10 processors are available for use. The CPU's processors are meant for high serial compute performance with high clock speeds extending from 3 to 5 *Ghz*. The clock speed is the number of calculations that can be performed per second, the larger the number the better. The GPU has on the order of 1000 to 2000 processors but lower clock speeds usually around 1 to 2 *Ghz*. Because the GPU has lower clock speeds they are not appropriate for performing everyday operations but instead for compute intensive operations such as graphical processing. The GPU also lacks interrupts and the virtual memory necessary for the OS. Therefore, the CPU and the GPU work in combination where heavy compute portions can be transferred to the GPU as shown in Figure 18. The parallel algorithms added to the DMB framework one occurs on the CPU and the other on the GPU.

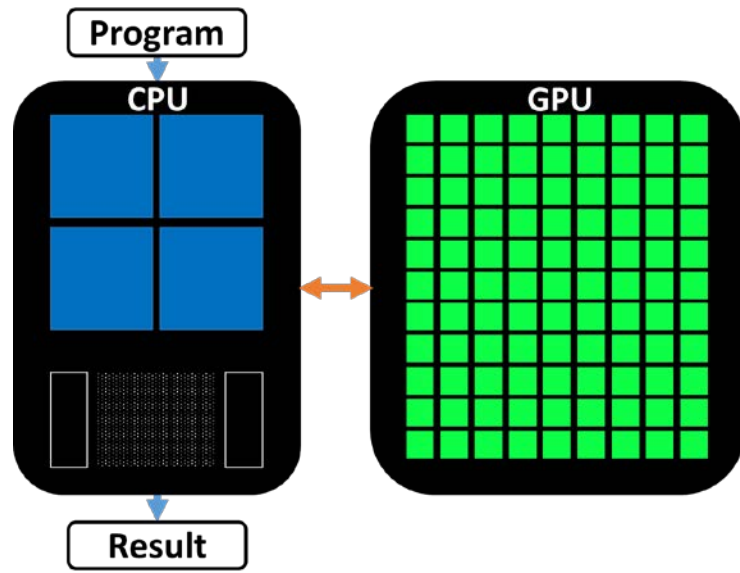


Figure 18: CPU-GPU Program Processing

Numerical analysis on the GPU is an ongoing area of research (Hernández Pérez, et al., 2018). GPU utilization is seen as a way to meet the expectations set by Moore's Law. To meet this standard, the CPU clock speeds have been increased incrementally over time. These higher clock speeds have been achieved by passing higher voltages through the processor elevating the temperatures, requiring more heat dissipation. A CPU material thermodynamic and performance threshold has been found as the temperatures for higher clock speeds are now surpassing material melting temperatures. Therefore, to follow the trend set by Moore's Law, a hybrid CPU and GPU load approach must be employed. It is projected GPU code parallelization will help meet Moore's Law in the coming years. Figure 19 illustrates the expected GPU extrapolation, notice that single thread performance is tapering off. NVidia believes that this tapering of performance can be regained through the use of GPUs and Amdahl's Law (NVidia, 2018).

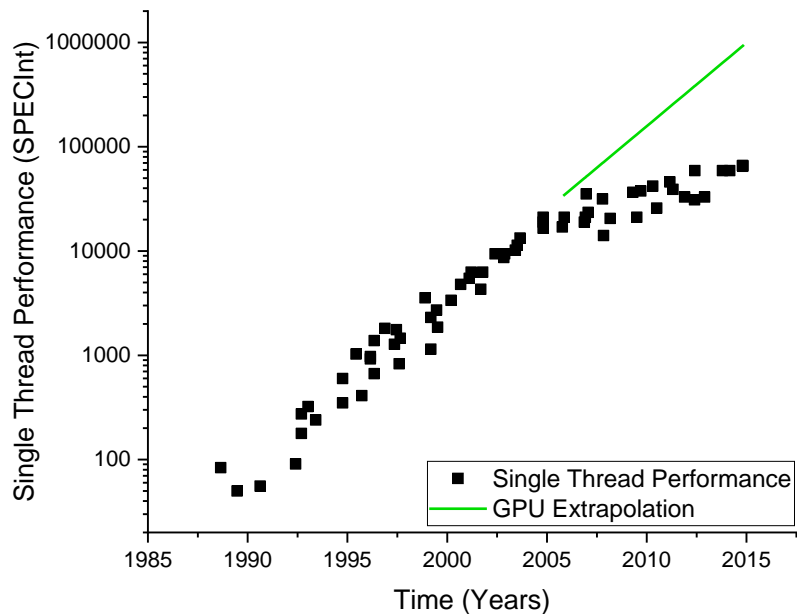


Figure 19: 35 Years of Microprocessor Trend Data (Adapted from (Harrod, 2012))

To find areas that could gain from code parallelization the MSVS performance profiler was utilized on the DMB framework. Two key areas were found, one in the kinetic parameter estimation routine and the other in ODE solver routine. The kinetic parameter estimation is one of the most time consuming steps in the model building process increasing its performance is crucial. In the kinetic parameter estimation routine, the objective function is called numerous times to determine the effect of a perturbed set of parameters on simulations. Two factors increase the time spent in the objective function, the time it takes to run a simulation and the number of dataset simulations that need to be run. A quantifiable example of the issue will now be described. If each dataset simulation takes 5 seconds and there are 4 datasets, sequentially the time spent in the objective function is 20 seconds. The sequential workflow in the heuristic methods objective function is illustrated by Figure 20. If

more datasets are added to the framework the time increases linearly. In the case where the user has a large amount of data points, multiple simulation in the objective function may lead to a large amount of time to be spent in the objective function making a kinetic parameter estimation solution intractable.

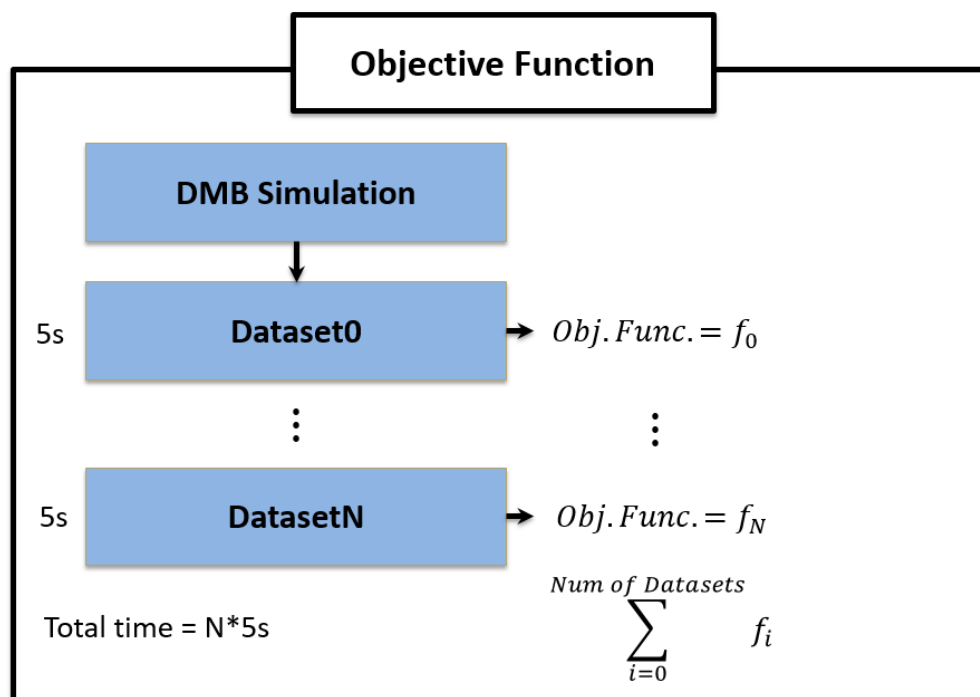


Figure 20: ASA Sequential Objective Function Calculation

The second key area is in the ODE solver which is used for simulations. As the number of species increases so does the time spent in the ODE solver. This is because a linear system of equations involving a Jacobian matrix needs to be solved at every time step. The dimensions of the Jacobian matrix scale to the number of species in the system as an N by N matrix. As the Jacobian matrix size increases the sequential LU decomposition algorithm takes longer to perform calculations. This leads to 80% of a

simulation time to be spent in the ODE solver. To parallelize each code segment, the requirement of each must be considered.

To parallelize a piece of code it must be discretized into pieces of work that can be solved simultaneously. In regards to the dataset simulations in the objective function, the ideal situation is that each dataset can be run parallel. Because the dataset simulations require access to the model class the parallelization hardware requires a considerable amount of memory. For this reason, dataset parallelization in the objective function was realized on the CPU through processor multi-threading. Open Multi-Processing (OpenMP) is an application programming interface in C++ that supports shared memory multi-processing programming. It uses a set of compiler directives and library functions to pass information to the processors. The shared memory aspect is very important since all the processors must access information from the model class memory. The new parallel workflow is illustrated in Figure 21 where CPU parallelization decreases simulation time in the objective function by the following equation (3.9-50).

$$t_{Objective\ Function} = \frac{\sum_{i=0}^{N_{Datasets}} t_{simulation,i}}{N_{Processors}} \quad (3.9-50)$$

The implemented parallelization allows for tractable solutions to be obtained during kinetic parameter estimation by decreasing time spent in the objective function.

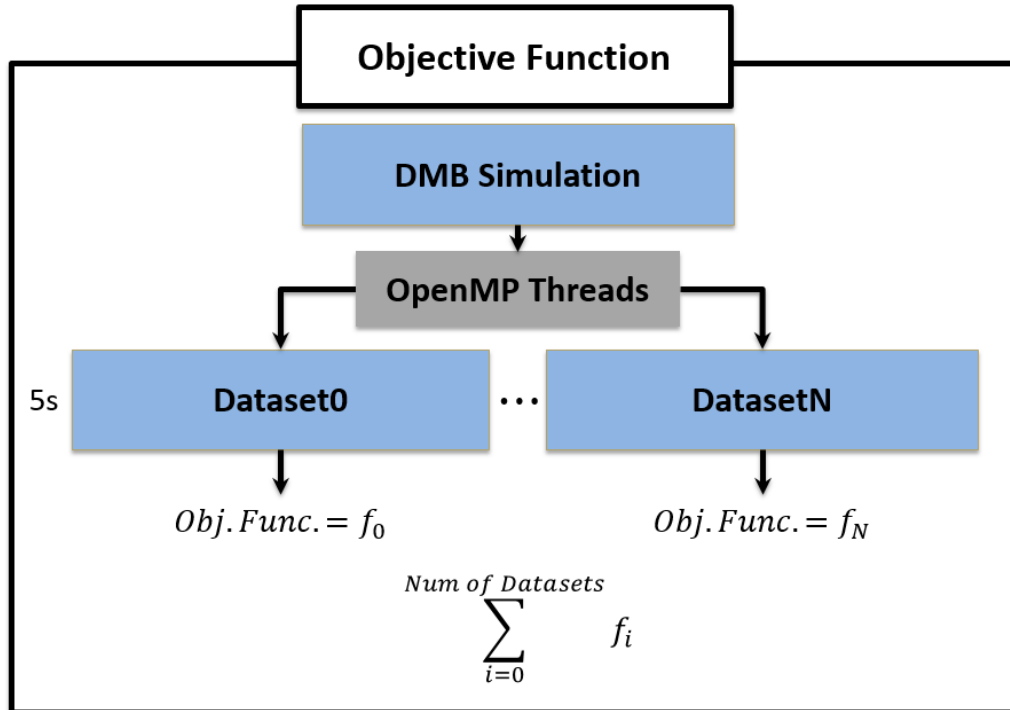


Figure 21: Parallel ASA Objective Function Calculation

Now a large number of species leads to a large Jacobian matrix. As the Jacobian matrix sizes increase so does the time spent performing serial lower-upper (LU) decomposition for numerical integration. The memory requirement is not as huge as that of simulation; the memory utilization is for an $N \times N$ matrix. A considerable amount of research has gone into parallelizing LU decomposition; significant breakthroughs in performance have been achieved through the use of GPUs (Baboulin, 2010). The work on the Matrix Algebra on GPU and Multi-Core Architectures (MAGMA) library has seen large performance increases when using GPU LU decomposition over its CPU counterpart with matrix sizes greater than a thousand (Baboulin, 2010). The MAGMA library is widely available for usage on HPC's. This made it a perfect candidate library for the GPU parallelization of the ODE solver algorithm on the University of Delaware's Farber HPC. Hence a GPU

parallel algorithm for the LU decomposition was implemented into the CVODE solver. The Farber HPC hardware utilized in this work was the following: 2.5 GHz Intel E5-2670 v2 (“Ivy Bridge”) coupled with 2 NVIDIA K80 GPUs. MAGMA is designed for multi GPU utilization and defaulted to using the available system GPUs which were two in parallel.

To achieve ODE solver GPU parallelization, the CVODE library was altered to include the MAGMA library. The change occurred in the `cvode_dense.c` file and required a call to the LU decomposition routine in MAGMA library instead of the one supplied in CVODEs by default. To bind the two libraries together into one archive file recompilation was required. The archive files are a type of library file that can be compiled as an object to produce one binary. The recompiled product CVODE-MAGMA GPU-enabled archive file was then compiled with the DMB framework. This work primarily focused on performance so the best compiler was utilized to compile the DMB binary which was the Intel C++ compiler.

The CVODE MAGMA GPU-enabled library outperformed its CPU counterpart at matrix sizes greater than 1k. The added performance decreases the time spent in the ODE solver for large systems. It should be noted, systems under a thousand species the GPU LU decomposition routine does not outperform its CPU version. This is because even though the GPU has thousands of processors they have low clock speeds. The test undergone in this work focused on once through simulations. The transfer of information from host the CPU to the device the GPU did not impede solution times. Figure 22 illustrates the difference in solution times between the CPU and the GPU for varying model species sizes.

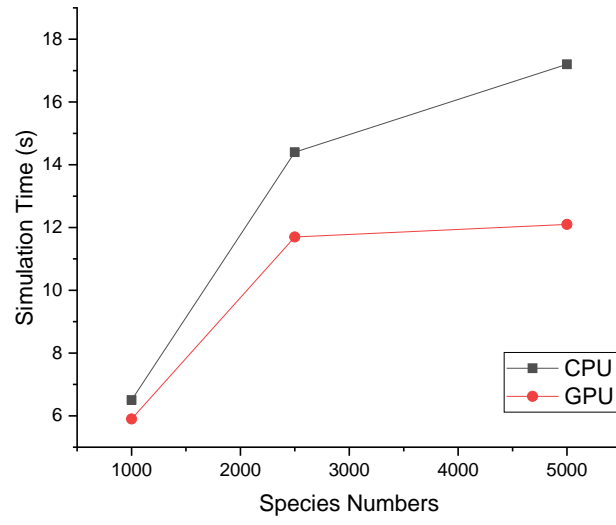


Figure 22: LU Decomposition Performance Comparison

3.10 Model Analysis

As the model complexity and rigor increase the comprehension of the model simultaneously decreases as illustrated in Figure 23. This paradox plagues the development cycle for models with complex and large reaction networks. Large models include many competing reactions that differentiate from one another by several parameters like concentrations, temperature, kinetic parameters, and QSRCs. The model development cycle of large systems can be halted due to a misunderstanding of the reaction network dynamics.

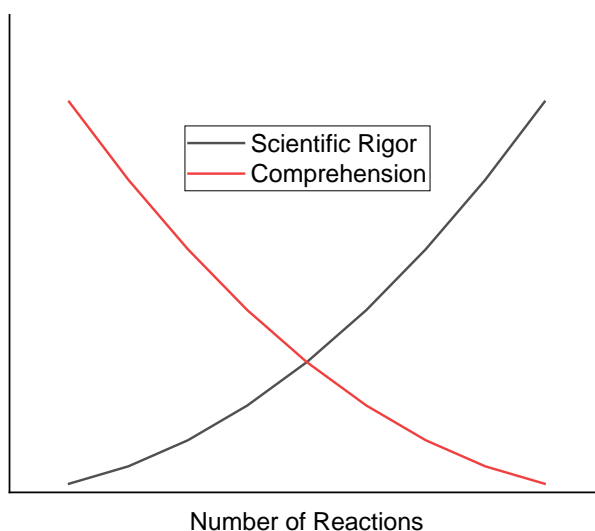


Figure 23: Comprehension vs Scientific Rigor Paradox

Analysis of the reaction fluxes in a model can help achieve a qualitative and quantitative understanding parameter contributions. Visualization of a molecular-level information can help determine problematic regions in the model. Previous model visualization efforts focused on illustrating rate law fluxes as 2D pie plots (Billa, et al.,

2017). The work presented here focused on adding another layer complexity to the previous efforts by visualizing reactor dynamics through 3D surface plots.

The work by Billa et al. focused on averaging the contributions from reaction rate law fluxes along the differential length of a reactor to obtain a qualitative understanding through pie plots. This approach visualizes the output but abstracts the dynamics of the reactor. Equilibrium affects are also averaged and making difficult to determine thermodynamic issues. This averaging approach implemented by Billa et al. was employed because the KME profiles are granular in nature. KME calls the CVODE solver once between the initial and final differential lengths. The CVODE solver steps through the interesting areas in the ODE functions but extrapolates the “un-interesting regions” to solve the numerical integration quickly. The CVODE solver utilizes a variable time step, changing step size based on interest. KME prints information from the ODE file during CVODE solver calls. Many points along the reactor length are missing because of the variable step size and extrapolation. Therefore, Billa et al. developed an averaging approach to capture a qualitative understanding without requiring the entire dynamics.

The DMB framework performs simulation in a different way. Instead of calling the ODE solver once, DMB calls the ODE solver several times based on the number of data points the user wants to print from the reactor. This translates into DMB displaying a discrete and continuous simulation profile. The profile includes all the concentrations along the reactor length. Obtaining a continuous profile instead of granular one like KME allows the user to extract more information from the output. The following approach can only be achieved by the DMB framework because of how it cultivates rate laws implicitly. The new approach is rate law flux reconstruction post

simulation which utilizes a simulation profile to rebuild the rate laws fluxes. This requires very little overhead since the ODE solver is not being run. Since the rate laws can be reconstructed so can the mole balances along the reactor length. Moreover, each term utilized to calculate the mole balance is now available. The mole balance is summation of the product between the stoichiometry and the reaction rate law. In this work the product of the stoichiometry and the reaction rate law is considered the reaction rate law flux. The DMB framework can print the rate law flux for a species of interest as a function of reactor length. Generally, in large reaction networks species are consumed and generated by multiple reactions leading to a substantial number of terms in the mole balance equation for species. For each species i , DMB can display the reaction rate law flux in a matrix form. Where the rows represent the reactor length and each column represents the rate law flux of certain reactions. To understand this new approach an example for benzene hydrocracking will be illustrated. In the example, 44 reactions either generate or consume benzene during hydrocracking. Benzene can be either be formed by dealkylation and ring opening reactions or consumed by saturation reactions. To perform model analysis DMB simulation must be accomplished. Post simulation the DMB framework mode can be switched to perform rate law and mole balance reconstruction. The modeling result produces a matrix that can be plotted as a 3D surface plots illustrated in Figure 24 and Figure 25. The 3D surface plots illustrated in Figure 24 and Figure 25 have the reaction rate law fluxes in the z -direction. The magnitudes of the fluxes are illustrated by the color map that ranges from red to purple, red being the highest quantity and purple being the lowest. A positive value in the z -direction illustrated by a red color map shows that the species is being produced by a reaction and negative value illustrates that the species

is being consumed. The x - and y -directions are illustrated by reactor length (dm) and reaction index. DMB outputs the index of each reaction so that this information can be related to the reaction network.

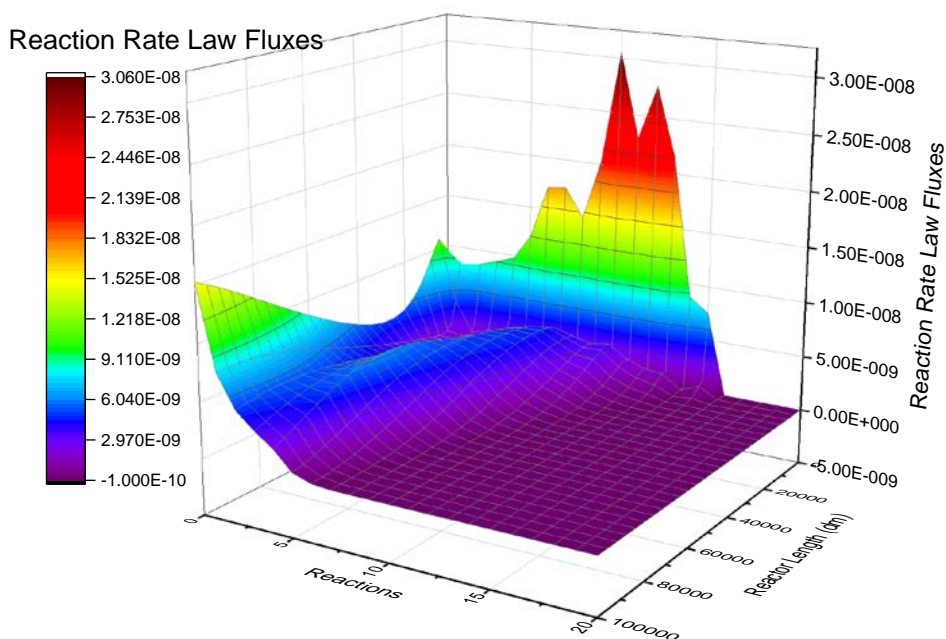


Figure 24: Reaction Rate Law Flux 3D Map (Reaction 0-20)

The 3D surface plots in Figure 24 displays the information for the first 20 reactions that involve benzene. The first 20 reactions are dealkylation reactions. Notice the largest reaction flux comes from the reaction zero. Reaction zero also shows an intermediary phase at the beginning by having an apex early in the reactor length. This modeling approach allows the user to understand the importance of certain reactions and may allow for some pruning of the reaction network. Notice that reactions 17 to 20 don't have significant flux changes. The reaction rate law flux for the first 20 reactions looks like it is an important region but it is actually not. The

reaction rate law fluxes in Figure 24 are two orders of magnitude smaller than the reactions 20 through 44. Figure 25 illustrates the more important reactions and pin points to a heavy producer of benzene which is another dealkylation reaction number 23. It also illustrates the consumption of benzene through the saturation reaction number 44. In this case the dealkylation reaction number 23 was singled out as problem since it was over producing benzene which did not match with molecular experimental information. This approach allowed for a qualitative understanding of the evolution of benzene through a reactor. Since it was observed that the dealkylation reactions of the system were overproducing benzene, the dealkylation reaction family activation energy was increased by a value of 1 which decreases their overall rate constant. The model was simulated again and the benzene concentration of the model output now matched that of the experimental. The final models reaction rate law fluxes were analyzed as shown in Figure 26 and it became apparent that the saturation was stronger by an order of magnitude according to reaction rate law flux after the increase of the activation energy. While the dealkylation rate law flux decreased by an order of magnitude.

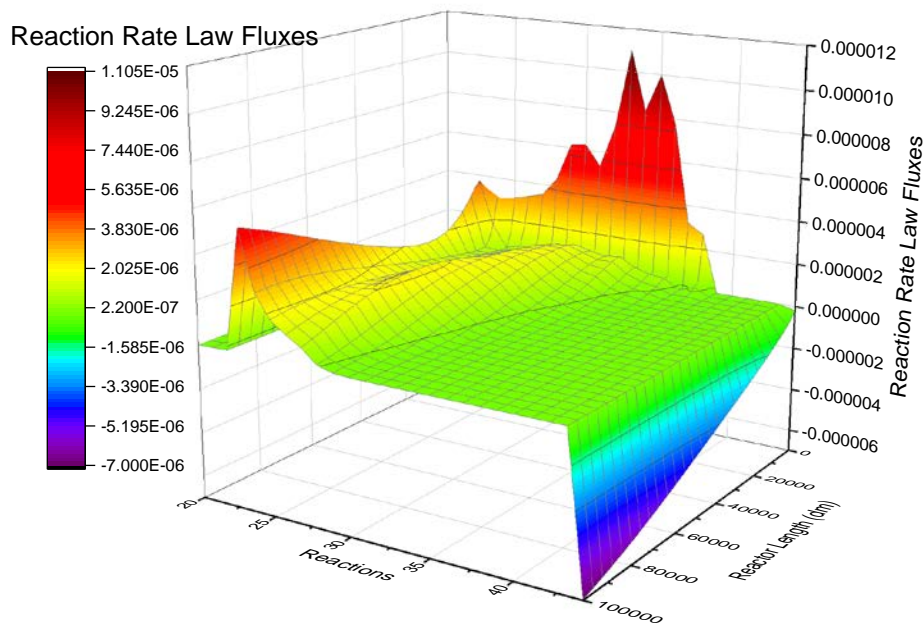


Figure 25: Reaction Rate Law Flux 3D Surface Map Initial (Reactions 20-44)

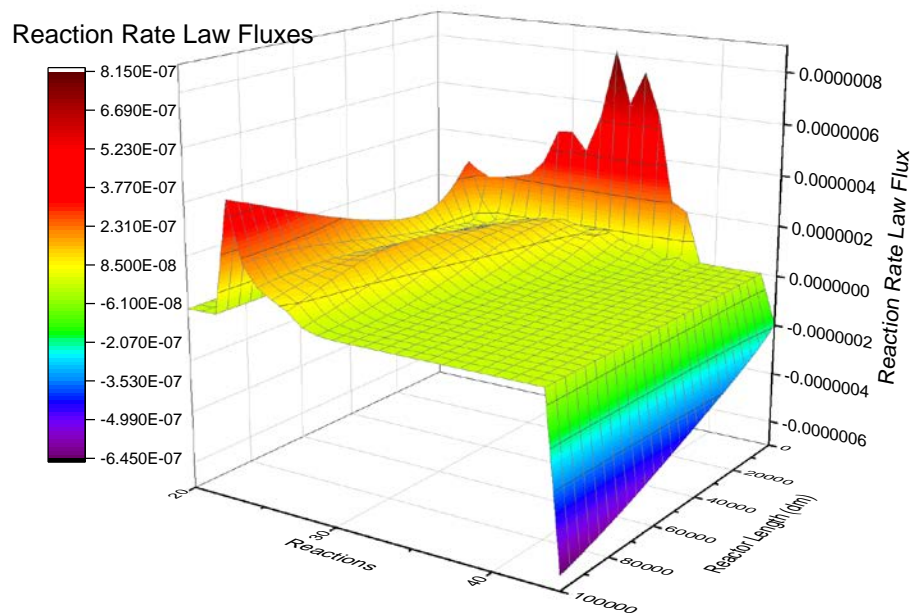


Figure 26: Reaction Rate Law Flux 3D Surface Map Final (Reactions 20-44)

3.11 Directory and File Configurations

When the DMB framework is utilized as a model independent framework it does not require code compilation to run various models. In fact, all that is needed is one DMB executable and a set files that describe the chemical process. These files are model configurations that the DMB framework requires to build and solve models from memory. A specific language must be followed for the DMB framework to correctly parse the chemical process information from the configuration files. This language creates a uniform model design where coding experience is not necessary and is only dictated by a DMB standard. It also makes it easy to track all the changes made to a model since all the inputs given to the DMB framework are in human readable format. This organization allows models between users to be easily analyzed and compared. The upcoming sections will be covering the model configurations in more detail.

3.11.1 Model Format

To further understand how the DMB framework works and which options are available to the user, the file input/output (IO) that describes the configuration of a model will now be covered. The input files for DMB have been organized in a user friendly way within a Model directory. Upon DMBs execution, the program first looks for the Model directory in the location where the DMB application exists. If it cannot find the model directory, an error is thrown and the program exits. If it finds the model directory, it accesses and reads the files required for simulation and kinetic parameter estimation. The information extracted from these files is stored in memory. The contents of Table 6 describe the model directory hierarchy in the DMB framework. A description of the subjects in each directory is also included.

Table 6: DMB Model Directory Hierarchy

Main Directory	Sub Directory	Directory Description
Model	Mode	DMB setting control file
	Reactions	Reaction network file
	Kinetic Parameters	Kinetic parameter file
	Properties	Species properties and mixture property names
	Datasets	Multi-Dataset Information: <ul style="list-style-type: none"> • Loading information • Product fractions • ODE solver absolute tolerances • Tuning information • Initial conditions • Reactor conditions • Observed output: <ol style="list-style-type: none"> 1. Species quantities 2. Mixture property
	Chemistry	LHHW Information: <ul style="list-style-type: none"> • Hydrogen dependence information • Adsorption constants information • Reaction and site information • Adsorption group exponent
	Optimization	Adapted Simulated Annealing: <ul style="list-style-type: none"> • Tuning handle controls <ol style="list-style-type: none"> 1. Kinetic parameters 2. Adsorption constants 3. Hydrogen dependence • Optimization initial conditions • Upper and lower bounds • Best solutions • Objective function information
	Solutions	Multi-Dataset Solutions: <ul style="list-style-type: none"> • Reactor profile • Mixture property result for fractions • Equilibrium and adsorption constant information

3.11.2 Model Directories

This section focuses on describing the sub directories in the Model folder. The model directory has eight sub directories: Mode, Chemistry, Reactions, Kinetic Parameters, Datasets, Optimization, and Solutions. The information each directory holds follows its naming convention. For example, if the user wanted to know where kinetic parameters were stored Kinetic Parameters directory is the first place to look. The organization of these sub directories makes it easy to work in the DMB framework and follow a logical flow when altering files. All these directories are required for the DMB framework to function.

3.11.2.1 Mode Directory

The mode directory holds the file that controls all the user settings available to a user. The control file can be considered the backbone of DMB and dictates how it will function. The file path for the Control.dat is Path = Model/Mode/Control.dat. The options set for the control file are illustrated in Table 7 below.

Table 7: Control File Option Information

Setting	Options
Exe Type	DMB/Model
Chemistry	Microkinetic (MK)/LHHW
Temperature Profile	Isothermal/Energy Balance/ User Defined
Property Calculation	ON/OFF
Simulated Distillation Calculation	ON/OFF
Phase	Gas/Liquid/VLE
Mode	MultiSim, ASA, RNA, WriteODE
Total Dataset Number	Number of Datasets
Datasets Reactor Type	CSTR/Batch/PFR
ODE Solver Maximum Steps	1000 [Default]
ODE Solver Relative Tolerance	1E-4 [Default]
Tuning Handles	LFER, Kads, and All
Number of Fractions	Product Fractionation Number

Parallel Optimization	ON/OFF
Number of Processors	Number of threads to be used

Exe Type (Executable Type) - This is the first setting in the control file and it dictates how the framework will function: model independent or dependent. The two options available are: DMB or Model. The “DMB” option is the model independent selection where compilation is not necessary. It carries the namesake DMB because it is the dynamic way to build models. The model dependent option is “Model” and is typically used if the user would like to use explicit ODEs. There are advantages to both approaches in terms of added control for the model dependent option and ability to model larger systems with the model independent option. As mentioned earlier the DMB framework can accommodate the need for explicit ODEs and become model dependent. In order to achieve a model dependent framework, a set of explicit ODEs must be written to a file. The DMB framework can convert itself into an ODE generator and write the explicit ODE to a file called ModelODE.cpp. This file must then be placed in the same location as other project code files, usually replacing a default or previous versions of ModelODE.cpp. To integrate the new file into the framework the project must be recompiled. After compilation is complete the Exe Type can be set to Model to use the ModelODE.cpp instead of the usual static DMB ODE.cpp.

Chemistry - This option controls the type of chemistry that will be used in the DMB framework. The two current options are microkinetic (MK) and LHHW.

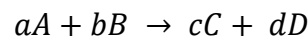
Temperature Profile - The three options available in the DMB framework are Isothermal, Energy Balance, and User Defined. The isothermal option is chosen when the user wishes to simulate a reactor in which temperature is constant. The energy

balance selection is for a chemical reactor in which the temperature varies. An energy balance adds an additional ODE to the mole balance ODEs. The solution of the energy balance is at the end of simulation profile. The user defined option allows the user to define a temperature profile. The points between the temperature profile points are interpolated.

Property Calculation - This controls whether the framework will calculate a mixture property. This option can always be left on.

Simulated Distillation Calculation - This controls whether the framework will calculate the simulated distillation calculation. This option can also always be left on.

Phase - This option controls the phase of the chemical process being modeled in the framework. If the gas phase is chosen, then the Arrhenius rate constant is in terms of pressure and has the following relationship to concentration rate constant.



$$k_p = k_c(RT)^{a+b+c+d} \quad (3.11.2-51)$$

The options for Phase are as follows G (Gas), L (Liquid), and VLE (vapor liquid equilibrium)

Mode – The mode option controls the task the framework will perform upon execution. Three options are currently available for users:

- Multi Simulation (OPTION = MultiSim)
- Adaptive Simulated Annealing (OPTION = ASA)
- Reaction Network Analysis (OPTION = RNA)
- Write ODEs (OPTION = WriteODE)

MultiSim runs once through simulations of loaded datasets and shows objective function values based on the difference between model and observed values. ASA mode uses the Adaptive Simulated Annealing parameter estimation heuristic to find the kinetic parameters that fit an objective function criterion.

Total Dataset Number - Number datasets that are in the Model/Dataset directory.

Dataset Reactor Type - This option determines the reactor type used in the framework. The options are Batch, CSTR, and PFR.

ODE Solver Maximum Steps - Controls the maximum numbers of steps the ODE solver can take.

ODE Solver Relative Tolerance - Controls the ODE solver relative tolerances, currently set to 1E-4. This value determines the accuracy of the ODE solver.

Tuning Handles - This option allows the user to control the parameters that can be tuned during the parameter estimation mode of ASA. Three types of parameters can be tuned in DMB:

- Linear free energy relationships (OPTION = LFER)
- Adsorption constant parameters (OPTION = Kads)
- β exponent of hydrogen partial pressure dependence (OPTION = All).

If LFER is chosen, then asatuningswitches.dat information will be read. If the K_ads option is chosen, then asakadstuningswitches.dat information will be read by the DMB framework to determine the adsorption tuning handle. If the All option is chosen, then both files will be read and everything can be tuned in the modeling framework.

Number of Fractions - The product in DMB can be cut into fractions based on boiling points.

Parallel Optimization - During the ASA objective function calculation, multiple dataset simulations are typically run in series, from which their individual objective function values are summed to equal an overall objective function value (For more information see the Dataset folder section). This may be time consuming for large systems in which there is a large number of experimental information available. By choosing the ON option multiple datasets can be run in parallel by using multiple central processing unit (CPU) threads. This is achieved by using OpenMP programming directives.

Number of Processors - The number of CPUs or threads used in parallel optimization mode.

3.11.2.2 Chemistry Directory

The chemistry folder holds additional chemistry information important to rate laws. It is designed so that a developer can add different chemistry information or build upon the existing chemistry types already in the DMB framework. The DMB framework defaults to microkinetic and power law type rate laws. If the user wishes to simulate catalytic systems, the extra information needed formulate an LHHW rate law is stored in the Chemistry sub directory LHHW. If the LHHW chemistry option chosen in the Control.dat file three files in Table 8 must exist in the LHHW folder. These files contain the information to build surface rate controlled LHHW rate laws with adsorption constants, catalyst concentrations, hydrogen dependence, and adsorption group exponents. It should be mentioned that if the microkinetic chemistry option is chosen, then these three files are never read by the DMB framework and don't have to be present in the LHHW directory.

Table 8: LHHW Directory File Hierarchy

File	Contents
H2Dependence.dat	Initial P_{H_2} and β (Tunable)
KadsConstants.dat	Contains: <ul style="list-style-type: none"> • Site • Adsorption constants • Property Index
RxnSiteMap.dat	Contains: <ul style="list-style-type: none"> • Site • Adsorption Group Exponent

The H2Dependence.dat file includes the parameters that control the hydrogen dependence of an LHHW rate law. In the theory section the hydrogen dependence is shown to be a function of hydrogen concentration and tunable parameter β . The equation (3.11.2-52) below shows the hydrogen dependence form for a gas phase system in the DMB framework. Equation (3.11.2-52) shows the hydrogen partial pressure, P_{H_2} , a measured reference hydrogen partial pressure, $P_{H_2,Ref}$ and a tunable β parameter. All these parameters work together to control the hydrogen dependence effect on the rate law.

$$f = \left(\frac{P_{H_2,Model}}{P_{H_2,Ref}} \right)^\beta \quad (3.11.2-52)$$

An example of the comma separated file H2Dependence.dat is shown below in Table 9.

Table 9: Hydrogen Dependence File Format

Name	$P_{H_2,Ref}$	β
PH2beta	1	1

The adsorption constant file, KadsConstants.dat, stores the site number, site concentration, site adsorption parameters, and site property indexes. All these parameters together can be used to calculate the adsorption constant value. The natural log of the adsorption constant is a linear function of temperature that can be described by the linear equation (3.11.2-53) below:

$$\ln K_{ads,i,Site=x} = m \frac{1}{T} + b \quad (3.11.2-53)$$

The adsorption constant defines the adsorption of a molecule to a catalyst site. The adsorption of the molecules is structurally dependent and varies with temperature. Therefore, the equation above can be developed into the equation (3.11.2-54) below to account for quantitative structure property correlations.

$$\ln K_{ads,i,Site=x} = a_x + \sum_{j=0}^{Constants} b_{j,x} \frac{QSPC[Property Index]_{j,x}}{RT} \quad (3.11.2-54)$$

The value of the adsorption constant for each species, i , on site, x , can be described by the temperature independent tunable parameter a_x , a set of temperature dependent tunable parameters $b_{j,x}$, and quantitative structure property correlations. In the DMB framework the user can dynamically decide the quantitative structure property correlations needed to distinguish the temperature dependent molecular adsorption terms. An example of the KadsConstants.dat file is shown in Table 10 for a two site system. The values in the KadsConstants.dat file are separated by commas. The DMB framework is designed to give the user a lot of flexibility in terms of the types of rate laws that can be used in a chemical process. The system can actually

handle a situation where a user can have non-catalytic and catalytic chemistries all in one process. Even if the LHHW option is chosen non-LHHW microkinetics and power law rate laws can still be defined for specific reactions. For this reason, the site index starts at 1 for catalytic systems. The zero value of the site index is reserved for defining microkinetic rate laws. The adsorption constant file outlines the number of sites in the system. Following the site index, the concentration of sites is stated. The concentration of sites is constant multiplier on the numerator of the LHHW rate law. The unit of the concentration of sites depends on the reactor and phase, for a gas phase PFR reactor the units are kg-cat/Liter. After the site concentration is defined the adsorption constants and molecular structure dependencies are described. The first term is the temperature independent term a . The definition of adsorption constants always requires a temperature independent term a , if the user does not wish to use it can easily be zeroed out rendering it function useless. Following the temperature independent term is the temperature dependent terms and property indexes. This information is coupled since the value of interest is product of the temperature dependent term and the molecular property. To define molecular property a species property index is used. A list of available species properties in the DMB framework and their indexes are shown in Table 39. The DMB user may define as many catalytic sites as deemed necessary for heterogeneous catalyst. The user may also define one site in the case of homogenous catalyst.

Table 10: Adsorption Constant File Example

Index	C_T	a_x	$b_{1,x}$	$PropertyIndex_{1,x}$	$b_{2,x}$	$PropertyIndex_{2,x}$
1	0.01	0.1	0.1	28	0.32	29
2	0.01	0.4	0.2	28	0.23	29

Once the site information is available, the framework must be directed to know which site each reaction occurs on. The file that contains this information is RxnSiteMap.dat which holds the reaction index, the site it occurs on, and the exponent term of the adsorption group. The RxnSiteMap.dat is comma delimited file that corresponds with the rxns.eqn file. An example of RxnSiteMap.dat is shown in Table 12 for a two reaction system. Notice the first reaction in the rxns.eqn file corresponds to reaction index zero in the RxnSiteMap.dat file. The reaction site map file allows the user to control the site each reaction occurs on and adsorption group exponent value. The exponents of the adsorption group vary based molecular adsorption, desorption, dissociation and reaction homogeneity, the table below illustrates several values for the exponent of the adsorption group (Yang, 1950) (Froment, Bischoff, & de Wilde, 2011).

Table 11: Adsorption Group Exponents (Adapted (Froment, Bischoff, & de Wilde, 2011))

Chemistry Type	Exponent of Adsorption Group
Adsorption of Reactant A controlling without dissociation	$n = 1$
Desorption of Product R controlling	$n = 1$
Adsorption of Reactant A controlling with dissociation	$n = 2$
Impact of A without dissociation $A + B \leftrightarrow R$	$n = 1$
Impact of A without dissociation $A + B \leftrightarrow R + S$	$n = 2$
Homogenous reaction	$n = 0$

As mentioned earlier the framework allows for the situation where catalytic and non-catalytic chemistry occurs simultaneously. To use non catalytic rate law the site value can be turned to zero defaulting the rate law to a standard microkinetic.

Table 12: Reaction Site Mapping File Format Example

Reaction	Reaction Family	Rxn Index	Site	Adsorption Group Exponent
<i>A</i> → <i>B</i>	R1	rxn0	1	1
<i>B</i> → <i>C</i>	R2	rxn1	2	1

The H2Dependence.dat, KadsConstants.dat, and RxnSiteMap.dat files allow the user to dynamically alter the LHHW rate law to a preferred variety. This type of control did not exist in the previous framework but is necessary for a molecular level modeler to build robust models. The LHHW section described information that relates to the reaction network utilized in framework which will be covered in the following section.

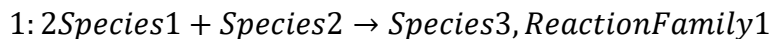
3.11.2.3 Reactions Directory

The rxns.eqn file in the Reactions directory is critical to the DMB framework. It is read every time a model is run and is the second file to be parsed after the setting file Control.dat. The rxns.eqn file must be located in the Reactions directory. The format of the file is vital to having a uniform model design. The user can write reactions in the file, define reaction families, and set the rate law types. Generally, all the reactions default to a microkinetic rate law, if the LHHW option is chosen LHHW formalisms are applied to the microkinetic rate laws. In the reactions file the user can decide to default between microkinetic and power law rate law types. Either can have LHHW formalisms applied to them. The user can have combinations of microkinetic and power law types with or without LHHW formalisms applied to them all in one model. This versatility sets the framework apart from anything before it. To have this

type of control a standard was created. To understand the format an example of an irreversible reaction is shown below.



The first thing to point out is that a number ahead of a character string is parsed as the stoichiometry of the species. For the reaction above, 2*Species1*, illustrates an absolute stoichiometry value of 2 for *Species1*. Any number after the character string is considered part of the name the species. In the example previously mentioned the species name is *Species1*. The reactants are separated by the (+) character. The reactants and products are separated by a forward arrow which means the reaction is irreversible (->) or a double sided arrow (<->) which means the reaction is reversible. The products follow the arrow type and are also separated by the (+) character. After the reaction, a coma must separate the reaction family from the reaction. The reaction family can have any name or number in it. The DMB framework is built around the reaction family concept in order to use LFER to describe the rate constant. So far a description for the default microkinetic reactions has been covered. To change the rate law to a power law type rate law, a colon is placed in front of the stoichiometric value as shown below.



For the example shown above the colon signifies that there is an override on the order of reaction for *Species1* which is being changed from reactant order 2 to 1. Even though the reaction order for *Species1* has been changed the stoichiometry of 2 remains for mole balances. Once the colon is found, the rate law type for the reaction

converts to a power law type. For a more concrete example see Figure 27 for a set of saturation reactions.

```
918 1:3species2.+species256.<->.species257,Saturation6H-AromaticCR LF
919 1:3species2.+species205.<->.species250,Saturation6H-AromaticCR LF
920 1:3species2.+species267.<->.species225,Saturation6H-AromaticCR LF
921 1:3species2.+species271.<->.species230,Saturation6H-AromaticCR LF
922 1:3species2.+species275.<->.species232,Saturation6H-AromaticCR LF
923 1:3species2.+species282.<->.species234,Saturation6H-AromaticCR LF
924 1:3species2.+species284.<->.species238,Saturation6H-AromaticCR LF
925 1:3species2.+species286.<->.species240,Saturation6H-AromaticCR LF
926 1:3species2.+species288.<->.species242,Saturation6H-AromaticCR LF
927 1:3species2.+species292.<->.species244,Saturation6H-AromaticCR LF
928 1:3species2.+species294.<->.species246,Saturation6H-AromaticCR LF
929 1:3species2.+species296.<->.species248,Saturation6H-AromaticCR LF
930 1:3species2.+species298.<->.species258,Saturation6H-AromaticCR LF
931 1:3species2.+species302.<->.species278,Saturation6H-AromaticCR LF
932 1:3species2.+species306.<->.species290,Saturation6H-AromaticCR LF
933 1:3species2.+species308.<->.species300,Saturation6H-AromaticCR LF
934 1:3species2.+species313.<->.species311,Saturation6H-AromaticCR LF
935 1:3species2.+species315.<->.species321,Saturation6H-AromaticCR LF
936 1:3species2.+species317.<->.species253,Saturation6H-AromaticCR LF
937 1:3species2.+species319.<->.species163,Saturation6H-AromaticCR LF
938 1:3species2.+species323.<->.species265,Saturation6H-AromaticCR LF
939 1:3species2.+species325.<->.species335,Saturation6H-AromaticCR LF
940 1:3species2.+species310.<->.species342,Saturation6H-AromaticCR LF
941 1:3species2.+species327.<->.species343,Saturation6H-AromaticCR LF
942 1:3species2.+species330.<->.species344,Saturation6H-AromaticCR LF
```

Figure 27: Example DMB Reaction File

3.11.2.4 Kinetic Parameters Directory

In the kinetic parameters directory, there is only one file, the KineticParametersLFER.dat. This file holds the numerical values for the kinetic parameters used in the DMB framework during simulation. This file is critical for a DMB simulation. It holds the reaction family name and five corresponding kinetic parameters that control each family. The number of reaction families in the LFER file (KineticParametersLFER.dat) must match the number of reaction families in the reaction network file (rxn.eqn). If there isn't a match, DMB will exit and throw an error during run time. For each reaction family j there is a $\log_{10} A$, E_a , α , a , and b

parameters. The kinetic parameters file follows the following comma separated format shown in Table 13.

Table 13: Example Kinetic Parameters File

ReactionFamily_j	log₁₀ A_j	E_j	α_j	a_j	b_j
RF1	10	30	0.1	1	1
RF2	10	30	0.1	1	1
RF3	10	30	0.1	1	1

Up to this point the pre-exponential, activation energy, and alpha parameters have been mentioned in the Theory section under the LFER concept subsection. In the microkinetic rate law equation (3.11.2-55) below the three Arrhenius LFER parameters affect the rate constant expression alone.

$$r_i = A_j e^{-\frac{(E_j + \alpha_j \Delta H_{rxn,i})}{RT}} \left(C_A^a C_B^b - \frac{C_C^c C_D^d}{K_{eq,ij}} \right) \quad (3.11.2-55)$$

The previous framework KME only utilized three Arrhenius reaction family parameters to control the dynamics in the reactor. It has been observed that the equilibrium constant for reversible reactions has a strong control over the dynamics in the reactor, especially at higher temperatures (Korre, Quann, & Klein, Hydrocracking of Polynuclear Aromatic Hydrocarbons. Development of Rate Laws through Inhibition Studies, 1997). The DMB and KME frameworks compute the equilibrium constant by employing the van't Hoff equation with values for the enthalpy and Gibbs free energy of reaction calculated through group contribution methods. Group contributions methods are approximations and may not be completely representative. In the case that experimental data is showing an opposing trend controlling the equilibrium may be the answer.

The work presented here describes two new tunable parameters, a_j and b_j , that modify the van't Hoff equation for each reaction family (j). This idea is not novel but differentiates it from the previous framework in that the equilibrium constant for each reaction family can be controlled (Korre, Quann, & Klein, Hydrocracking of Polynuclear Aromatic Hydrocarbons. Development of Rate Laws through Inhibition Studies, 1997). Equation (3.11.2-56) illustrates the modified form of the van't Hoff equation (3.11.2-56) with the equilibrium constant as function of reaction family and the new modifiers a_j and b_j .

$$\ln K_{eq_{ij}} = -\frac{b_j * \Delta H_i}{RT} + \frac{a_j * \Delta S_i}{R} \quad (3.11.2-56)$$

Equation (3.11.2-56) allows for manipulation of the temperature dependent (enthalpy) and independent terms (entropy) of the van't Hoff Equation. The example in Figure 28 illustrates the manipulation of van't Hoff equation by changing the a_j parameter to shift the line vertically of the benzene hydrogenation. Note the temperature dependence remains unchanged but the equilibrium values do change. Conversely the temperature dependence can be altered as well. This approach must be coupled with an extent of reaction analysis to understand the proximity of the reaction to equilibrium.

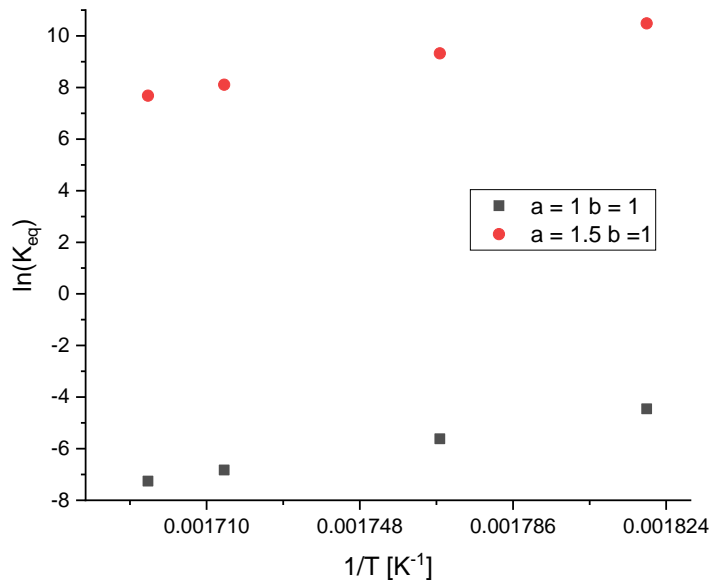


Figure 28: van't Hoff Plot Manipulation through Modifiers

3.11.2.5 Datasets Directory

Typically, a DMB user is trying to simulate experiments, for which the user may have a lot of parameters and information. The dataset folder allows the user to store and access the experimental information they are trying to model. The dataset folder is one of the most complex folders in the DMB framework. The directory is meant to hold the information that is experiment specific in an organized and logical format. With this being said the following information from experiments can be stored here:

1. Reactor conditions
2. Reactor input
3. Reactor measurements

The dataset folder also contains other framework specific files such as species ODE tolerances, product fractionation settings, and dataset loading. The Datasets directory file descriptions are shown in Table 14.

Table 14: Datasets Directory File Information

Files Dataset Directory	Description
DatasetAbstol.dat	Each species in the DMB model has an absolute tolerance. This is the accuracy the ODE solver uses for each species. This option defaults to 1E-6. If tighter accuracy constraints are required this is the location the user should control.
DatasetLoading.dat	The dataset loading file indicates the dataset that will be loaded into the DMB framework. The structure of the file is as follows: $DatasetN, 1$ Where $N = 1, \dots$, Total Number of Datasets. The number after the comma should be either a 1 or 0. The numbers act as a switch where 1 is load and 0 is don't load.
DatasetProductFractionation.dat	DMB output can be split into fractions based on boiling point. The file format is as follows: $Fraci, Use, T_{min}, T_{max}$ Where $i = 1, \dots$, Total Number of Fractions. The <i>Use</i> value can be 1 (ON) or 0 (OFF). It dictates whether the fraction will be used in framework. If it is set to ON an objective function will be calculated from the observed fraction information file. A default setting for the removal of only hydrogen concentration from the product is available, shown below. $Frac0, 1, 0, 0$

The dataset loading file allows the user to control which datasets are being loaded into the DMB framework. The user may choose to insert one experiment and has the option to load more. An example of a dataset loading file is shown in Table 15 below for four datasets. In the example only Dataset0 is loaded of four dataset system. Notice a Boolean format is used where 1 is load and 0 don't load. This design allows the user to decide which datasets to simulate or perform kinetic parameter estimation on. This control allows the framework to reduce unnecessary overhead.

Table 15: Dataset Loading Example

Dataset Name	Load [Load = 1, Don't Load = 0]
Dataset0	1
Dataset1	0
Dataset2	0
Dataset3	0

The product of a chemical reactor in DMB can be fractionated into boiling point cuts via the Dataset Product Fractionation file. This functionality is available because the measurements of experimental reactors are not typically on the entire reactor output but portions that can be measured experimentally. Therefore, DMB allows for product fractionation to match the output being measured. DMB also allows the user to split the reactor output into more than one boiling point cut. This can all be done from file IO via the DatasetProductFractionation.dat and the Control.dat files. In the control file, the number of fractions are set. In the dataset fractionation file the user must set the fractions and their corresponding boiling cut ranges. An example of a three boiling cut system is shown in Table 16. The actual file is comma delimited. In the example only Frac0 is being utilized by DMB since the number one is set in the Boolean Used column. Also notice that when the values for Min and Max temperature

are both set to zero this defines an internal setting to expel hydrogen from the reactor output. The other fractions in the file are not utilized since Used Boolean value is zero. If the Used value was set to 1 for any of the other fractions the DMB framework will need min and max values for each fraction utilized. DMB will also print a final composition solution if the Used column is set to one. Remember the number of used fractions must match the number of fractions in the control file. The example below only has one fraction in the control file.

Table 16: Product Fractionation Example

Fraction Name	Used [Boolean]	Min Temperature[K]	Max Temperature[K]
Frac0	1	0	0
Frac1	0	400	600
Frac2	0	200	300

So far information about the files in the Datasets directories have been touched upon but nothing has been mentioned about where the experimental information is stored in the Model directory. The idea is that there is a subdirectory that follows the naming convention, DatasetN, where N is number of experiments. The DatasetN subdirectory holds all the experimental information and must have the name “Dataset” followed by a numeric value. The numeric value is used as an index of the data set in the DMB framework. The first dataset will always have the name “Dataset0”. When adding datasets to the Datasets directory, the number line must be followed to keep order, therefore the succeeding dataset would be “Dataset1” then “Dataset2” and so forth.

Using the first dataset sub directory as an example, “Dataset0”, the experimental information storage can be defined. The Dataset0 directory contains

three sub directories, InitialCond, ReactorInfo, and Observed as shown in Table 17. In terms of levels this can be thought at 2 levels from Datasets directory.

Table 17: Datasets Directory Hierarchy

Directory	Sub Directory	Sub [Sub-Directory]
Datasets	Dataset# (First: Dataset0)	<ul style="list-style-type: none"> • InitialCond • ReactorInfo • Observed

3.11.2.5.1 Initial Condition Directory

The initial conditions directory contains files that hold the species initial values and the differential length of the system. The information from both of these files is required to solve the initial value problem in a DMB simulation. The initial conditions file is called, Initial_Condition_y.dat and contains the species names followed by the concentration of that species fed into the reactor. DMB will search the file, specifically looking for species names that match those present in the reaction network. If a species is missing from this file, and is present in the reaction network, an error will be thrown. Extra species that are not present in the reaction network can exist in the file but will not be used. An example of the information found in the initial condition file is found in Table 18 below. The initial conditions file is separated by commas. The units of the initial conditions depend on the reactor system being used. The batch and CSTR reactors have units of mols per liter while the PFR reactor has units of moles per second.

Table 18: Initial Condition File Format Example

Species	Separator	Concentration
A	, (Comma)	1
B	, (Comma)	0

The second file in the initial condition directory is the DifferentiationLengthInfo.dat which contains the differential length values. The reason why this is called the differentiation length file is because the mole balances take derivative with respect to either reactor length or time in reactor. The difference between which is used in the DMB framework depends on the reactor type used. The format of the differential length file is shown below in Table 19.

Table 19: Differentiation Length File

Initial Differentiation	Separator	Final Differentiation	Separator	Intervals Number
0	, (Comma)	1	, (Comma)	50

In the Table 19 example format, if the information was applied to a PFR reactor then the differentiation would occur along the length of the reactor from zero to one decimeter. As eluded earlier, depending on the reactor type the differentiation length has different units as well, seconds if CSTR or Batch and decimeters if PFR. The interval number is the number of points that will be acquired along the differentiation length as well as the number of points displayed in the output simulation profile. A default value for the number of intervals is 50. It should be noted that DMB has a set number of steps that the solver can take between each interval. These steps are defined as the maximum number of steps the ODE solver can take. The number of steps can be defined in the setting file Control.dat. For stiff systems more steps must be taken by the ODE solver to numerically integrate the system. If the number of intervals is changed then the user should also change the ODE solvers

maximum number of steps accordingly. In the Table 19 example if the maximum step size is equal to 1000, the ODE solver can take a maximum up to 50,000 steps in total.

3.11.2.5.2 Reactor Information Directory

The reactor information directory contains the reactor conditions file. The name of file follows the following naming notation ReactorType.dat, where Reactor Type can be a Batch, CSTR, or PFR. Within every reactor file, indifferent to the reactor type, three parameters separated by a new line character should be present. The three parameters are temperature, pressure, and cross sectional area. A Reactor file description is shown in Table 20 below along with the required units.

Table 20: Reactor Info File Format

Reactor Parameter	Units
Temperature	Kelvin
Pressure	Atmospheres
Cross Sectional Area	Cross sectional area of the reactor in decimeters squared

3.11.2.5.3 Observed Directory

The observed directory contains observed values from experimental output as well as the objective function control information for the system. The DMB framework can handle two types of experimental output: molecular and/or bulk mixture properties. Species concentrations and mixture properties can both be included in the objective function of the DMB framework. The objective function in the DMB framework is Chi square function, equation (3.11.2.5.3-57) shown below, and can be used as a performance metric for the kinetic parameters employed.

$$\chi^2 = \sum_{i=0} \left(\frac{\text{Observed} - \text{Expected}}{\text{Weight}} \right)^2 \quad (3.11.2.5.3-57)$$

The DMB output stream can be fractionated based on a temperature range as mentioned in the dataset product fractionation section. This is important because a lot of experimental measurements are performed on fractionated reactor output. Since the reactor output can be fractionated based on boiling point range, an objective function for each product fraction can also be developed in the DMB framework. DMB is designed around the idea that different product fractions can be part of the same objective function. In Table 21 the naming convention for an Observed directory is illustrated for a system with 2 product fractions.

Table 21: Observed Directory File Description

File Name	Description
SpeciesObsFrac0.dat	Observed file for observed species fraction 0
SpeciesObsFrac1.dat	Observed file for observed species fraction 1
MixPropObsFrac0.dat	Observed file for observed mixture properties fraction 0
MixPropObsFrac1.dat	Observed file for observed mixture properties fraction 1

When the output has one fraction then the Frac0 files are the only files read in the observed directory. So far, the objective function type and file structure used in the DMB framework for the Observed directory has been covered. Now the information found in the observed files will be detailed with a quantitative example shown in Table 22.

Table 22: ObsSpeciesFrac0 and MixPropObsFrac0 Inputs

Species or Mixture Properties	Observed Value (Units)	Alpha (α)	Weight (ω)
Species1	1 (mols/s)	0	1
Paraffin	0.4 (wt. %)	0	1

Ensuing what has been mentioned, the Table 22 example would have the following Chi square objective function equation (3.11.2.5.3-58) form:

$$\chi^2 = \left(\frac{1 - \text{Expected}_{\text{Species1}}}{1} \right)^2 + \left(\frac{0.4 - \text{Expected}_{\text{Paraffin}}}{1} \right)^2 \quad (3.11.2.5.3-58)$$

DMB supports two forms of the Chi square objective function in equation (3.11.2.5.3-57), one where the denominator is defined as a weight value and one where the weight is defined as, $\text{Weight}=\alpha*\text{Observed}$:

$$\chi^2 = \sum_{i=0} \left(\frac{\text{Observed} - \text{Expected}}{\alpha * \text{Observed}} \right)^2 \quad (3.11.2.5.3-59)$$

In the observed file, if there is a zero value for the weight and the alpha term then the species or mixture property will not be part of the objective function. If there is a non-zero value for the weight, the weight type objective function will be used. The weight type objective function is utilized even if the alpha term also has a non-zero value. Essentially whenever there is value for the weight it overrides the alpha value. The only time the alpha type objective function will be used is if the weight is zero and the alpha is non-zero. On the off chance that the user has a zero observed value when using an alpha type objective function DMB will not calculate that objective function

term and will instead display a warning message with information about the issue. DMB is designed to be a robust framework that can acknowledge when a user has chosen to tune species or mixture properties, if the user has only chosen to tune one, for example mixture properties, the framework will not calculate the species objective function bypassing an unnecessary overhead. This increases the performance because unnecessary objective functions calculations are not performed.

3.11.2.6 Optimization Directory

The optimization directory has a sub folder for each optimization metaheuristic method. Currently there is only one method implemented in the DMB framework and that is adaptive simulated annealing (ASA) from the California Institute of Technology (Ingber, 1995). ASA is a variant of simulated annealing for which heuristic parameters control the temperature schedule and the random step selection algorithmically². The ASA folder contains the information necessary to control kinetic parameter estimation. The directory contains the following files shown in Table 23.

Table 23: ASA Control Files

File	Description
asa_opt.file	This file contains ASA controls for: <ul style="list-style-type: none"> • Number of Tuned Parameters • Initial Conditions • Upper and Lower Bounds • ASA controls
asa_out.file	Results from Kinetic Parameter estimation

² More information about the metaheuristic method can be found at the following link: <https://www.ingber.com/ASA-README.html>.

	<ul style="list-style-type: none"> • Best Objective Function Values • Best Tuned parameter Values
asa_usr_out.file	Final Results Kinetic Parameter estimation
asatuningswitches.dat	LFER kinetic parameter tuning handle switches
asakadstuningswitches.dat	Adsorption constant tuning handle switches
CurrentBestKineticParametersLFER.dat	DMB LFER Results Format
CurrentBestKadsParameters.dat	DMB Kads Results Format
ObjValue.dat	Quick Glimpse at the Best Objective Function Value

3.11.2.6.1 ASA Tuning Switches File

There are two files that allow the user to control tuning handles in the DMB framework. The two file are `asatuningswitches.dat` and `asakadstuningswitches.dat`. They allow the user to control which LFER and Adsorption constant parameters will be estimated during an ASA kinetic parameter estimation. The format of the two mentioned files generally follows the same format found in `KineticParametersLFER.dat` and `KadsConstants.dat`. In the case of `asatuningswitches.dat` the format is reaction families followed by tuning switch values. The reaction families and tuning switch files are separated by commas. The files `KineticParameterLFER.dat` and `asatuningswitches.dat` look identical the difference being instead of values for the pre-exponential, activation energy, and alpha, there are tuning switch values of Boolean format. For example, when tuning the pre-exponential for certain reaction family the value is one for tuned and zero non-tuned. An illustration of the example above is shown in Table 24 where the user has chosen to tune all $\log_{10}A$ s in a system. For the system below when a tuning switch is zero it is

not tuned, the default value used for the parameter in the DMB framework comes from the information read from KineticParametersLFER.dat file.

Table 24: LFER Tuning Switches

Reaction Families	$\log_{10} A$	E_a	α
ReactionFamily1	1	0	0
ReactionFamily2	1	0	0
ReactionFamily3	1	0	0
ReactionFamily4	1	0	0
ReactionFamily5	1	0	0

As mentioned asakadstuningswitches.dat is the tuning control file for the adsorption constants of the system. It also must follow the format defined in the KadsConstants.dat file. The only difference is that in the asakadstuningswitches.dat file there isn't a need for the site concentration. The site concentration is not a tunable parameter in the DMB framework. Table 25 illustrates a 2 site catalytic system where the user wishes to tune on all the adsorption constants. Notice from the example in Table 19 that the hydrogen dependence tunable parameter α control is also included in the adsorption constant tuning control file. Instead of creating a separate file for the hydrogen dependence tuning switch, it was just added to the adsorption constants tuning switches file.

Table 25: Kads Tuning Switches File Format

Site	a	b	c
1	1	1	1
2	1	1	1
Ph2Alpha	0		

3.11.2.6.2 ASA Options File

The ASA options file controls the settings of the ASA metaheuristic method. The ASA options file is tab delimited and requires a specific tab format. The options available in ASA are shown in the Table 26. More information about these options can be found in the following link: <https://www.ingber.com/ASA-README.html>.

Table 26: ASA Options and Default Values

Option	Default Value
Limit Acceptances	1000
Limit Generated	99999
Limit Invalid Generated States	1000
Accepted To Generated Ratio	1.00E-04
Cost Precision	1.00E-18
Maximum Cost Repeat	5
Number Cost Samples	5
Temperature Ratio Scale	1.00E-05
Cost Parameter Scale Ratio	1
Temperature Anneal Scale	100
Include Integer Parameters	0
User Initial Parameters	1
Sequential Parameters	-1
Initial Parameter Temperature	1
Acceptance Frequency Modulus	100
Generated Frequency Modulus	10000
Reanneal Cost	1
Reanneal Parameters	1
Delta X	0.001
User Tangents	0
Curvature0	0

For an unexperienced user it is recommended to use the default options for initial kinetic parameter estimation runs. If the user wishes to change these options after not experiencing the preferred results that is also welcome. Besides storing the

options for an ASA run, the asa_opt file also stores the initial values and the upper and lower bounds for the kinetic parameter estimation routine. Before specifying these variables, the user must first identify the number of parameters to be tuned. Once the number of tuned parameters is set the user can define values that state a start and range for the kinetic parameters to be perturbed about during the kinetic parameter estimation. In the case of DMB framework the parameters that can be perturbed are the log10 based pre-exponential, activation energy, alpha, adsorption constants, and hydrogen dependence beta. Table 27 shows an example format that can be used to describe the tuning of LFER parameters. In Table the parameter index is followed by lower and upper bounds which are all separated by a single tab. The bounds are followed by the initial value again separated by a tab. The optimizer needs to know if the initial condition, lower and upper bounds are for a float or integer type so the initial value is followed by three tabs and a negative one to designate a float.

Table 27: Example 1: Initial Value, Lower and Upper Bounds File

Parameter Index	Lower Bound (Minimum)	Upper Bound (Maximum)	Initial Value
0	9	11	10
1	25	35	30
2	0	0.5	0.25
3	9	11	10
4	25	35	30
5	0	0.5	0.25

Example 1 illustrated in Table 27, is for a two reaction family system in which the user wishes to tune the all of the log10 based pre-exponentials, activation energies, and the alphas, essentially all the LFER parameters. The tuning switches file for the

system in Table 27 is illustrated in Table 28 below. It should be noted that parameter index shown is only for illustrative purposes in Table 28.

Table 28: Example 1: LFER Tuning Switches File

Reaction Families	$\log_{10} A$ [Parameter Index]	E_a [Parameter Index]	α [Parameter Index]
ReactionFamily1	1[0]	1[1]	1[2]
ReactionFamily2	1[3]	1[4]	1[5]

Notice that the parameter index is incremented horizontally and then vertically after each row. The parameter indices in the tuning switches file, shown in Table 28, directly coincides with the parameter indices in the Table 27 example. If the user decides not to tune on the pre-exponential terms as seen in Example 1, the user can change the tuning switches file to the one shown in Example 2 illustrated below in Table 29.

Table 29: Example 2: LFER Tuning Switches File

Reaction Families	$\log_{10} A$ [Parameter Index]	E_a [Parameter Index]	α [Parameter Index]
ReactionFamily1	0	1[0]	1[1]
ReactionFamily2	0	1[2]	1[3]

The change made in the tuning switches file changes the parameter indices in the ASA option file. The description of the initial value, lower and upper bounds must be changed accordingly to match the tuning switches file as illustrated in Table 30.

Table 30: Example 2: Initial Value, Lower, and Upper Bounds

Parameter Index	Lower Bound (Minimum)	Upper Bound (Maximum)	Initial Value
0	25	35	30

1	0	0.5	0.25
2	25	35	30
3	0	0.5	0.25

So far the relationship between the LFER tuning switches file and the ASA options file has been covered. From this base idea the complexity can be increased. The additional complexity is introduced when tuning adsorption constants and the beta term of the hydrogen dependence simultaneously while tuning the LFERs. The upcoming Example 3 will depict how a user would add adsorption constants to the tunable parameter list by altering the tuning switches files and the ASA options file.

Table 31: Example 3: Adsorption Constant Tuning Switches File

Site	a [Parameter Index]	b [Parameter Index]	c [Parameter Index]
1	1[?]	1[?]	1[?]
2	1[?]	1[?]	1[?]
Ph2Alpha	0		

To begin the user first must change the Tuning Handles option in the Control.dat file found in the mode directory to: All. Utilizing the existing ASA options and tuning switches files shown in Example 1 (Found in Table 27 and Table 28) the user can add adsorption constants to the tuning list by altering the asakadstuningswitches.dat file to the one illustrated in Table 31 for a two site catalytic system. In the DMB framework the parameter index is incremented first by all the LFER tuning parameters then adsorption constants and finally the beta hydrogen dependence term. Table 32 shows the parameter indexes if all the adsorption constants were added to the tuning list of Example 1.

Table 32: Example 3: Kads Tuning Switches File Update

Site	a [Parameter Index]	b [Parameter Index]	c [Parameter Index]
1	1[6]	1[7]	1[8]
2	1[9]	1[10]	1[11]
Ph2Alpha	0		

The required changes to the ASA options files for Example 3 in terms of initial values, lower, and upper bounds are illustrated in Table 33 below. Notice that final parameter index in Table 33 matches with final parameter index in the tuning switches file for the adsorption constants in Table 32.

Table 33: Example 3: Initial Value, Lower, and Upper Bounds

Parameter Index	Lower Bound (Minimum)	Upper Bound (Maximum)	Initial Value
0	9	11	10
1	25	35	30
2	0	0.5	0.25
3	9	11	10
4	25	35	30
5	0	0.5	0.25
6	0	0.5	0.25
7	0	0.5	0.25
8	0	0.5	0.25
9	0	0.5	0.25
10	0	0.5	0.25
11	0	0.5	0.25

If the user plans to also tune on the hydrogen dependence, then the hydrogen dependence term in tuning switches file for adsorption constants is switched from zero to one. The parameter index in the ASA options file is also incremented accordingly to include a new parameter in the tunable parameters list.

3.11.2.7 Properties Directory

DMB is designed to be a truly dynamic framework where the user can define multiple variables without adding any extra code. So far the user defined adsorption constants have been described, now user defined mixture property correlations will be covered. The capability of defining user mixture properties makes the framework flexible to different types of experimental information. To realize this dynamic mixture property environment four files are used:

- DMBDefinedMixPropDefinitionsList.dat
- MixPropDefintions.csv
- proplist.dat
- SpeciesPropertyNames.dat.

The DMB defined mixture properties definitions list contains all the DMB framework hard coded mixture properties. This list can be thought of as a library of mixture properties that can be used with in the MixPropDefinitions.csv. The Table 38 found in the Appendix shows list of defined mixture properties. The user just has to write the name of the mixture property, the basis, and that is DMB defined in the mixture property file and the framework will adapt and calculate the mixture properties. The order in which the mixture properties from Table 38 are defined in MixPropDefinitions.csv does not matter as long as the name matches the one in the library file DMB Defined Mix Prop Definitions List file. The user has three options for the basis of the fraction shown in for the DMBdefined mixture property: moles, mass, and volume. It should be noted that the simulated distillation can be in terms of mole, mass, and volume fraction. The basis must be for the entire simulated distillation range, therefore VIBP(K) controls the basis for simulated distillation.

Through the mixture property definitions file the DMB user can also define user defined mixture properties by outlining a set of molecular constraints. These constraints are based on species properties in the SpeciesPropertyNames.dat. An example is shown in Figure 29 where the user changed the definition of normal paraffin's to not include hexane. The mixture property definitions file is a comma separated value file (.csv) that can be opened in excel. The first column (Column A) in the file is the mixture property name; the second column (Column B) is the fraction basis which can be mass, mole, or volume fraction. The third column (Column C) notifies the framework if the mixture property is in the defined list of available mixture properties or if it is user defined. If the third column is user defined any number of columns after can be used for molecular constraints. The example in Figure 16 requires paraffin's so the z number is greater than zero, side chains are equal to zero and total carbon number is not equal to six. Notice that you use species property then logical statement followed by a value to describe molecular constraints. This is always the format you cannot describe a property by another property.

	A	B	C	D	E	F
1	Density(gg/cc)	Mass	DMBDefined			
2	MW(g/mol)	Mass	DMBDefined			
3	PONA_ModifiedP_WT	Mass	UserDefined	ZNum > 0	SideChains == 0	TotalCarbonNum != 6
4	PONA_ModifiedI_WT	Mass	DMBDefined			
5	PONA_O_WT	Mass	DMBDefined			
6	PONA_ModifiedN_WT	Mass	DMBDefined			

Figure 29: DMB Mixture Property Definition Example

The species property list is static and is based off the one used by the property database. These species properties are tracked in DMB by index in which they appear. Table 39 in the Appendix illustrates all the species properties that can be assessed by the DMB framework. It also illustrates the DMB frameworks property index. Finally,

proplist.dat file contains the species properties for each molecule. The species property file is created by the Property database by using group contribution methods. The properties from Property database include thermodynamic and structural information about every molecule. In the case the adsorption constants the user has the control the structural information that describes adsorption of molecules to sites. For example, if the molecules with long side chains attach more to site than aromatic compounds the user can decide how to express the adsorption constant correlation.

3.11.2.8 Solutions Directory

The solutions directory is where solutions from once through simulations and write ODE options are stored. Upon opening the Solutions folder two files are visible, the SolutionSpeciesMap.dat and SolutionTime.dat. The solution species map is the order in which species are organized and used in the DMB framework. DMB stores the species in its own order based on a sorting algorithm. The order is important because this is the order present in the profiles and species output. The file SolutionTime.dat stores the time the framework took to simulate or perform kinetic parameter estimation. The Solutions directory holds sub directories named after the dataset subfolder in the Datasets directory. The solution files in these sub directories are reactor output based on the dataset information in the Datasets directory. Table 34 below describes the files and the information they hold.

Table 34: Current Solution Files

File Name	Description
ConversionProfile.dat	Hydrogen Conversion profile across reactor length
deltaHrxnValues.dat	Enthalpy of Reaction for each reaction, reactions match up with rxns.eqn file
KadsValues.dat	Adsorption constant for each species and site

Keqvalues.dat	Equilibrium constants for all reactions, reactions match up with rxns.eqn file
SpeciesOutputFrac0.dat	Species output results at the final differential length in the reactor based on Frac# Definitions
MixturePropertyResultsFrac0.dat	Mixture Property results at the reactor output with objective function value based Frac# Definitions
SimulationProfile.dat	Simulation profile for DatasetN. First column is the Differential Length data. The columns after follow the order from the SolutionSpeciesMap.dat
RateLawFluxesSpecies#	Model analysis file that illustrates the rate laws fluxes as a function of reactor length. The file is organized as matrix for which each column describes the flux of specific reaction
RateLawFluxesMapSpecies#	Contains the reactions indexes that match the columns in the RateLawFluxesSpecies#.dat file

The simulation profile contains the differential length in the first column, the rest of the columns represent each species. The solutions species map contains the header of each column. Rows represent the differential length. The data from simulation profile can be plotted or used to analyze the model by the DMB framework. Figure 30 illustrates the use of simulation profile plot for a large number of components.

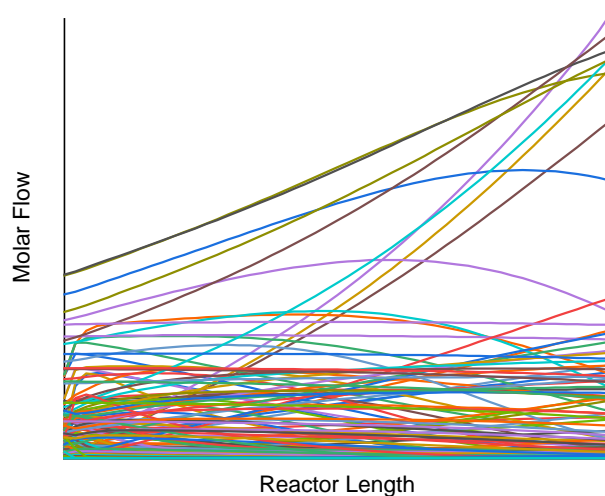


Figure 30: Example Simulation Profile Plot for a Large System

3.12 Conclusion

A new kinetic modeling tool was developed that caters to the next generation of molecular-level modelers where large systems are the norm. The new kinetic modeling environment streamlines the development of large systems but can scale well to a model of any size. The tool development had an emphasis on creating a dynamic tool that can handle microkinetic, catalytic, and multi-phase systems. The coupling of hardware and software through parallelization allowed for faster simulations and kinetic parameters estimations. In comparison to the previous framework DMB advances the KMT software capabilities to better handle difficult problems of the future.

Chapter 4

APPROACHES AND SOFTWARE TOOLS TO AID THE DEVELOPMENT OF PATHWAYS BASED MOLECULAR-LEVEL KINETIC MODELING OF LIGNIN PYROLYSIS

Juan C. Lucio-Vega and Michael T. Klein

4.1 Abstract

A molecular-level kinetic model by means of an adapted Freudenberg lignin structure was developed for lignin pyrolysis at the temperature of 600 °C. A detailed reaction network was constructed taking into account primary and secondary lignin pyrolysis chemistries from literature. An exhaustive reaction network merging approach was utilized to handle high structural reactivity. An adapted Freudenberg lignin structure was modified based on comparisons to reaction networks and experimentally observed products. The validation of the kinetic model consisted of a comparison between molecular species and measured experimental work. The model output showed agreement with experimental results from Zhang et al. (Zhang, Resende, & Moutsoglou, 2014).

4.2 Introduction

It is estimated that over the next 20 years fuels consumption will increase by ~60% (Matyas Kosa, 2011). As our reliance on the world's finite petroleum supply increases there is motivation for unconventional chemical routes. The thermochemical conversion of lignocellulosic biomass to fuels and high value chemicals is seen a sustainable option. Biomass is composed of three components: cellulose,

hemicellulose, and lignin. Lignin is the second largest constituent in biomass and is abundant in large quantities as a byproduct of the wood pulping industry. The fast pyrolysis of lignin produces a bio-oil rich in aromatic and phenolic compounds that can be used for fuel or commodity chemicals. The bio-oil composition produced from lignin pyrolysis varies by origin and processing conditions. Heating rate, reactor temperature and residence time are some of the processing parameters that affect the product distribution. Understanding the temperature dependence of this kinetically controlled process is imperative to increase its practicality.

The complexity of the lignin macromolecule complicates the development of a molecular-level kinetic model due to its large molecular size and its many reactive sites. A large number of reactive sites can cause a combinatorial explosion during reaction network development leading to an intractable reaction network. The work by Hou et al. utilized the attribute reaction modeling (ARM) approach to control the reactive site complexity of lignin by splitting the lignin molecule into components and reacting the components separately (Hou, Bennett, Klein, & Virk, 2010). This approach breaks the molecule into molecular segments and requires the molecule to be rebuilt from these segments losing molecular information. Specifically, the molecular substituent effects are lost.

Furthermore, the lignin macromolecule has a variable molecular structure based on origin and extraction from biomass. When modeling lignin pyrolysis, the unknown feed structure as well as the plethora of reactive sites complicates the model development. The work presented here focused on reacting large molecular representations of lignin with thermal degradation chemistry. Due to the reactive site complexity an exhaustive reaction network merging approach was developed to

produce reaction networks from large chemical structures. The reaction network merging approach was implemented on a large molecule adapted Freudenberg as well as several other large molecule adaptations. The produced lignin pyrolysis reaction networks were thereafter deterministically modeled and showed agreement with experimental gas chromatography results.

4.3 Lignin Molecular Description

Lignin is an oligomer composed of three monolignols: coniferyl, sinapyl, and coumaryl. Through radical oxidative coupling chemistry, the three monolignols polymerize to form lignin (Vermerris, Thompson, & McIntyre, 2002). The variability of the lignin macromolecule is due to the multiple reactive radical sites a monomer can have. The monomer radical reactivity can lead to a variety of polymerized products. Figure 31 illustrates the number of radical reactive sites a coniferyl alcohol may have. The many radical reactive sites lead to various types of ether-based inter-linkages during polymerization. When the molecule labeled R5 shown in Figure 31 binds to a monolignol phenol group a specific linkage is formed between the β carbon on the R5 and the oxygen attached to carbon-4 of the monolignol, this linkage is referred to as a β -O-4 linkage, which will have priority in the thermal degradation chemistry.

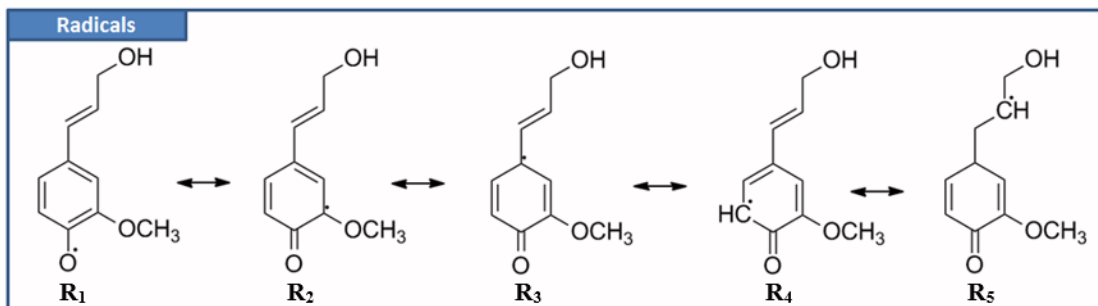


Figure 31: Coniferyl Alcohol Radicals

In fact the β -O-4 linkage has been found to comprise up to 50-70% of the linkages within the lignin macromolecule (Lewis, 2000). For some time this was thought to be random but now it is theorized that dirigent proteins mediate the specificity of the oxidative coupling chemistry during formation (Vermerris, Thompson, & McIntyre, 2002). The importance β -O-4 linkage is magnified by the ample reactive thermal degradation chemistries that can be applied to it. It has been observed by Faravellia et al. that the β -O-4 bond has the lowest bond energy of all the bonds in the ether linkage. The adapted Figure 32 illustrates the bond energies in the β -O-4 linkage with the beta carbon and ether oxygen having the lowest bond energy equal to 242 kJ/mol. This makes the β -O-4 linkage site a primary location for reactions to occur.

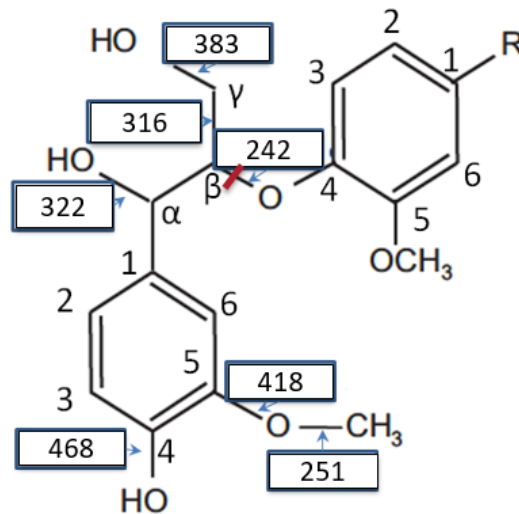


Figure 32: β -ether linkages typical bond energy [kJ/mol] atoms numbered for use, (Adapted) (Faravellia, Frassoldatia, Migliavacca, & Ranzia, 2010)

The oxidative coupling chemistry can also affect structural properties of the lignin based on the specific monomer reacting to extend the oligomer. The structural property affected is the differentiation between hardwood and softwood lignin. The difference between them is that hardwood is composed of both synapyl and coniferyl linked units and softwood is primarily composed of coniferyl linked units (Faravellia, Frassoldatia, Migliavacca, & Ranzia, 2010). Both types have low amounts of coumaryl units (Faravellia, Frassoldatia, Migliavacca, & Ranzia, 2010). These structural properties lead to very different product distributions during pyrolysis. Measurements of these properties are widely attainable and once coupled with a model, predictions can be made to determine the type of lignin structure required for pyrolysis. The work by Dellon et al. focused on using Monte Carlo simulations to determine a library of representative lignin structures (Dellon, Yanez, Li, Mabon, & Broadbelt, 2017). This work will not focus on developing a library of structures but altering an existing Freudenberg structure to model lignin pyrolysis.

Over time many representative structures have been proposed to describe the lignin macromolecule (Freudentberg, 1959) (Dellon, Yanez, Li, Mabon, & Broadbelt, 2017). An adaptation of the classical molecular structure created by Freudentberg is pictured in Figure 33 (Freudentberg, 1959). The large structure shown in Figure 33 has 185 carbon atoms and 210 hydrogen atoms. There are 18 benzene rings with 3 α -O-4 and 9 β -O-4 linkages in molecule which match the original interpretation. The adapted structure is of the hardwood type. The work presented here draws from the Freudentberg's lignin interpretation to develop lignin structures based on product distributions. Synapyl, coniferyl, and coumaryl units were linked together via β -O-C4 and α -O-C4 linkages to compose lignin structures. The produced lignin structures were thereafter reacted based pyrolysis chemistries to yield reaction networks. The product distribution of the reaction network that matched the observed experimental products was utilized.

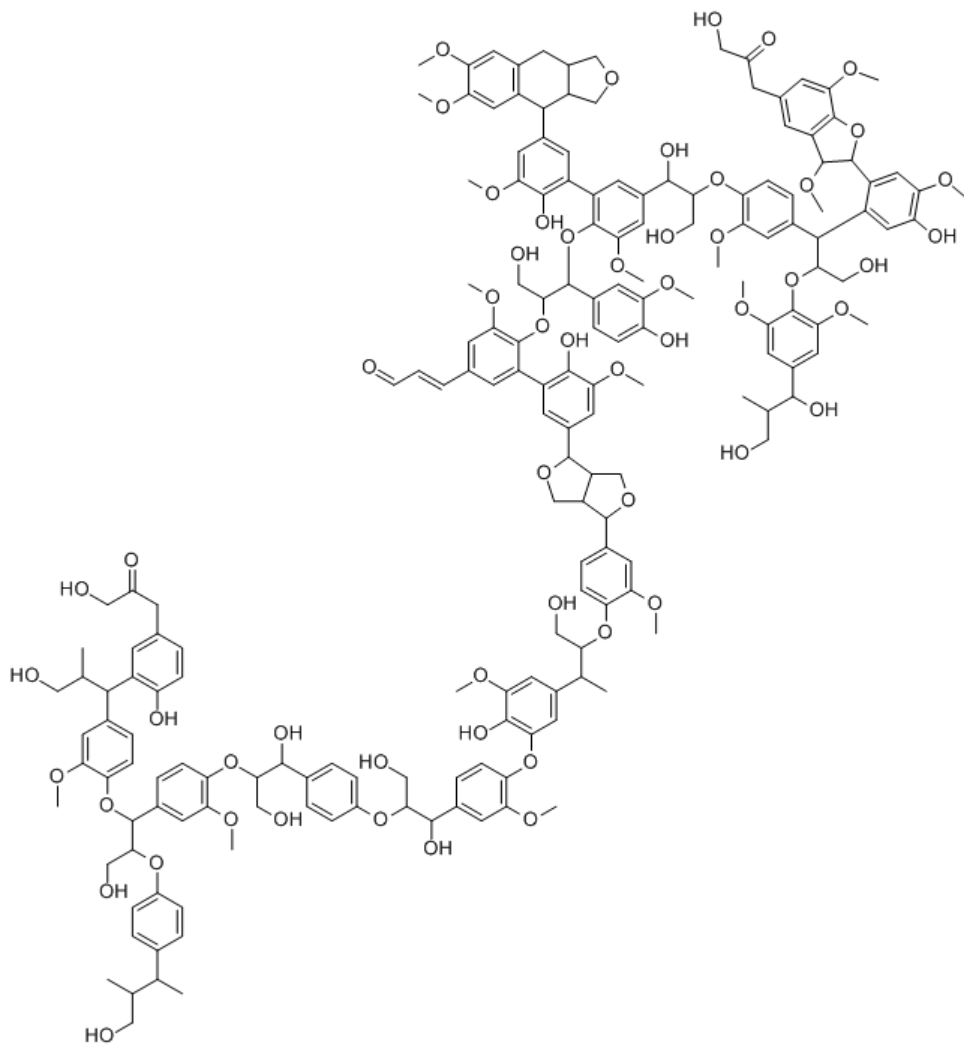


Figure 33: Adapted Freudenberg Lignin Molecule (Freudenberg, 1959)

4.4 Computational Methods

There is an abundance of chemistries that can be applied to the plethora of reactive sites on a lignin structure. To limit the complexity of the lignin pyrolysis chemistry this study focuses on using pathways chemistry. The pathways chemistry is separated by primary and secondary chemistries, for which the primary chemistries cleave the large structure into smaller components which thereafter experience secondary chemistries. The primary chemistries cleave the lignin molecule via β -O-C4

linkages and α -O-C4 linkages. The secondary chemistry further degrades the resulting smaller molecules post ether cleavage by enol-keto tautomerization, decarbonylation and vinyl degradation.

4.4.1 Reaction Network Construction

To create a reaction network an in-house software tool was utilized, the Interactive Network Generator (INGen). INGen can be programmed to automatically generate reaction networks. INGen constructs a reaction network by searching through all the reactive moieties of a seeded molecule and based on user-defined reaction chemistries a reaction network can be created. Simple matrix algebra is used to produce the products through reaction matrices that algorithmically describe the bond breaking and bond forming.

4.4.1.1 Primary Reactions: Ether cleavage and Alcohol Dehydration Pathways

In the studies by McDermott et al. and Klein et al. various pathways-level chemistries for the thermal degradation of lignin linkages were suggested. Their work focused on applying pathways chemistries to smaller molecules that resembled linkages found in lignin. The pathways-level chemistry from these two works motivated the primary reactions in this study. To retain atomic tractability, an atom balance was kept within all the reaction pathways adapted. Klein's pathway chemistry will be the first to be presented, as lignin pyrolysis chemistry is further introduced (Virk & Klein, 1983).

Klein and Virk's linkage cleavage pathway was presented in their work on a phenethyl phenyl ether (PPE) molecule (Virk & Klein, 1983). The PPE molecule contains a β -O-4 linkage that bridges two benzene rings and can be illustrated by the

reactant molecule in Figure 34. The difference between PPE and the conventional lignin β -O-4 linkage is the R groups, for PPE they are hydrogen atoms, for lignin they are alcohol side chains. The alcohol side chains in a β -O-4 linkage are either hydroxyl or methyl hydroxyl. This work assumes that the reactivity is the same even with the differences in R groups. Figure 34 illustrates the cleavage pathway, which will be elucidated in terms of bond breaking and bond forming. The bonds broken are between hydrogen (H_1) and alpha carbon (C_1) as well as beta carbon (C_2) ether oxygen (O_1). The bond formation occurs between (C_1) and (C_2) to form a double bond and (H_1) and (O_1) form an alcohol group. This bond breaking/forming can also be depicted by a reaction bond-electron matrix (0: no bond, -1: loss of bond, and 1: bond present) illustrated in Figure 34. In the figure the red color identifies bond breaking while the blue identifies bond forming.

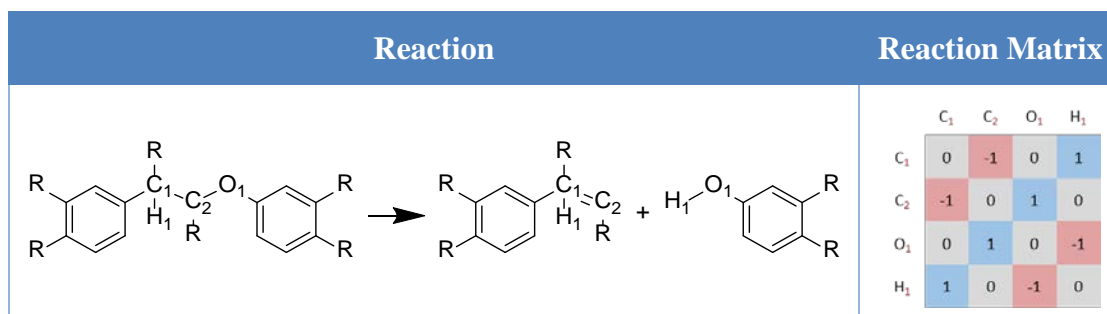


Figure 34: Klein β -O-4 Linkage Cleavage Pathway and Reaction Matrix

McDermott et al. proposed two pathways for the cleavage of ether linkages in lignin molecule. The first shown in Figure 35 is very similar to the previous pathway but with one exception, the R group on the alpha carbon of the β -O-4 linkage must be a hydroxyl. If this criterion is met then the following bonds in Figure 35 are broken, the bonds between H_1 and O_1 , H_2 and C_1 , and C_2 and O_2 . Bond forming occurs

between H₁ and O₂ forming an alcohol, O₁ and C₂ form ketone moiety, and H₂ and C₂ form a carbon-hydrogen bond. This pathway is illustrated in Figure 35 to form a ketone and a phenol while cleaving the β-O-4 linkage.

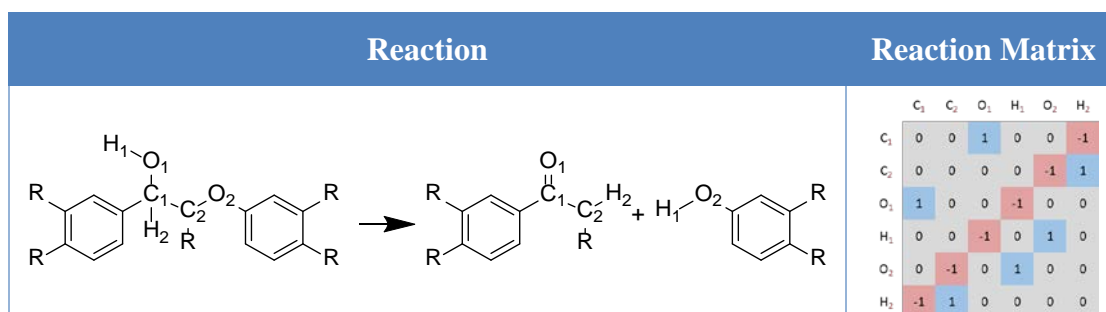


Figure 35: McDermott Ketone Forming β-O-4 Primary Cleavage Pathway and Reaction Matrix

The second pathway introduced by McDermott et al. for the breakdown of the β-O-4 linkage produces a vinyl group. The hydroxyl R group prerequisite on the alpha carbon must again be met. In the pathway, the β-O-4 linkage is dehydrated as shown in Figure 36. Two bonds that are broken to achieve this reaction are between the O₁ and C₁ and H₁ and C₂. The bonds formed are between O₁ and H₁ forming water, C₁ and C₂ forming an olefin. Figure 36 illustrates the mechanism in which water and a vinyl moiety is created in the β-O-4 linkage.

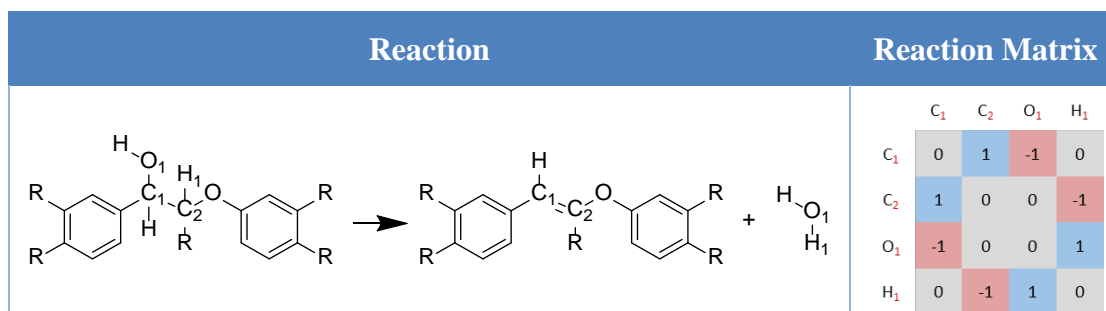


Figure 36: McDermott β-O-4 Linkage Alcohol Dehydration Pathway and Reaction Matrix

Following the creation of a vinyl group on the β -O-4 linkage, the linkage can experience a secondary cleavage pathway. The second pathway cleaves off the beta carbon and the ether oxygen, requiring diatomic hydrogen to be atomically balanced. The second cleavage pathway is illustrated in Figure 37.

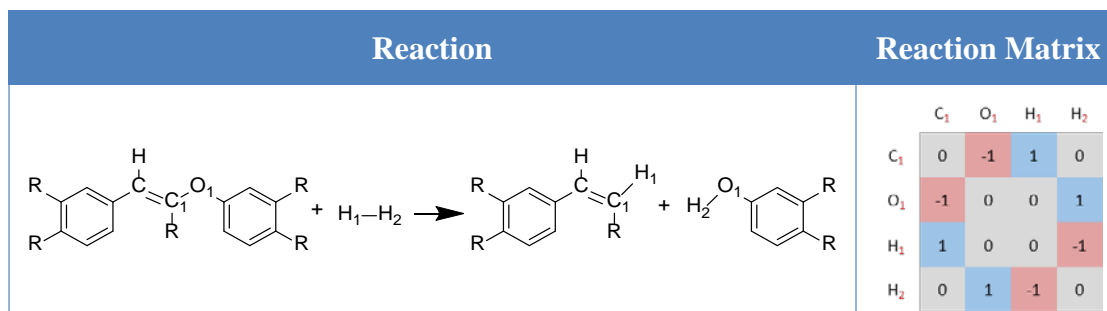


Figure 37: McDermott β -O-4 Linkage Secondary Cleavage Pathway and Reaction Matrix

The β -O-4 linkage thermal degradation within lignin has been thoroughly covered but other linkages account lignin degradation as well. The α -O-4 linkage also experiences an important thermal degradation pathway in lignin pyrolysis. Within this pathway, illustrated in Figure 38, the α -O-4 linkage cleaves the bond between the α -carbon and the ether oxygen. Hydrogen is again needed to keep an atom balance in the pathway.

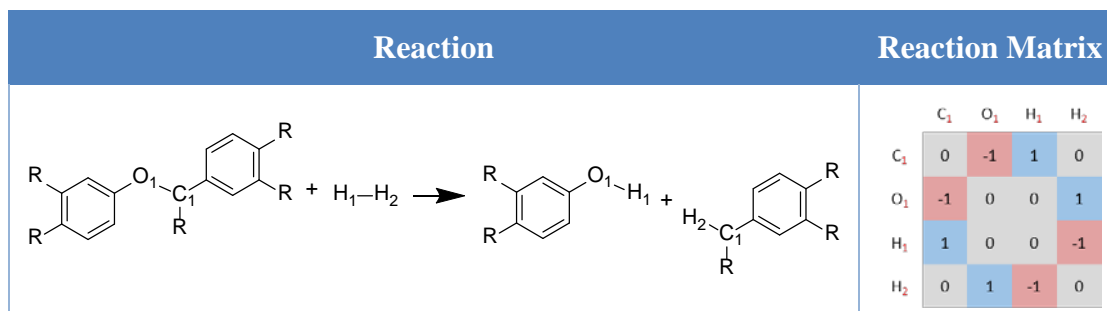


Figure 38: α -O-4 Ether Linkage Cleavage Pathway and Reaction Matrix

4.4.1.2 Secondary Reactions: Decarbonylation, Enol-Keto Tautomerization, and Styrene Degradation Pathways

Secondary reactions include decarbonylation, enol-keto tautomerization, and styrene degradation. Decarbonylation is the severing of carbon monoxide from an aldehyde group. Enol-keto tautomerization is the equilibrium reaction between enol (hydroxyl group) and the keto (ketone and aldehyde group). Lignin involves a large amount of phenolic and aldehyde groups therefore it is postulated that enol-keto tautomerization occurs. The work by Klein et al. also proposed styrene degradation as a secondary chemistry for the breakdown of lignin molecule (Virk, 1983). Styrene degradation, illustrated in Figure 39, begins by the breaking the double bond between the styrene moiety. Hydrogens balance the transformation from olefin to paraffin. The paraffin side chain is then further hydrogenated yielding methane. This pathway occurs until side chain is completely cleaved by alkyl units leaving only benzene.

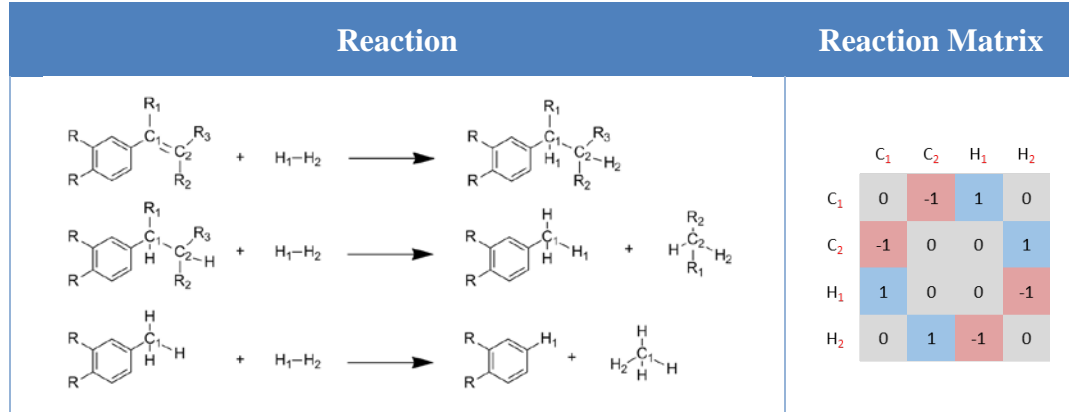


Figure 39: Styrene Degradation Pathways and Reaction Matrix

4.4.1.3 Hydrogen Incorporation

Many of the pathways based pyrolysis chemistries shown in this work require hydrogen to be atomically balanced. Two options can be used to incorporate hydrogen

into the system, the first through the involvement of dehydrogenation reactions, the second through the addition of hydrogen into the feed. It is hypothesized that the hydrogen aiding in the lignin pyrolysis comes from the lignin itself (Faravellia, Frassoldata, Migliavacca, & Ranzia, 2010). The dehydration reaction, illustrated in Figure 40, shows the dehydration of an alkyl side chain. In the dehydration reaction, bonds between the C₁ and H₁ as well as those between C₂ and H₂ are broken. The hydrogens cleaved sequentially form a hydrogen molecule and the carbons that lost the hydrogens form a double bond. The addition of dehydrogenation reactions causes more combinatorial issues in reaction network development due to the high number of reactive sites. The addition of dehydrogenation reactions may also be insufficient to capture the mechanistic transfer of hydrogen; therefore, the modeling work presented here utilizes hydrogen in the feed.

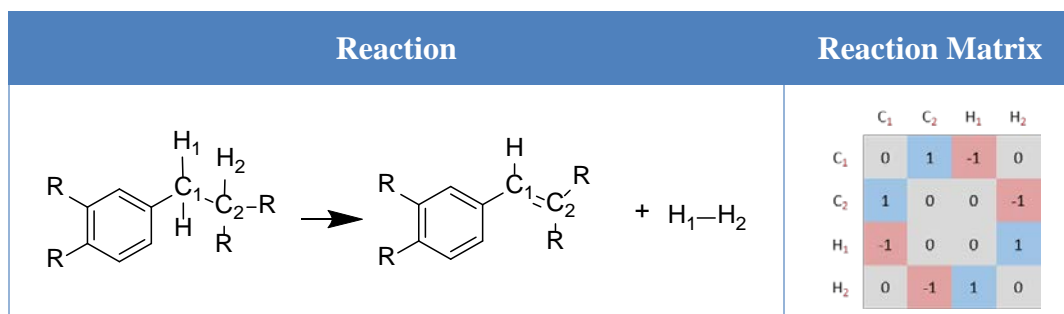


Figure 40: Dehydration Reaction Pathway and Reaction Matrix

4.4.2 Reaction Network Merging Approach

With such a large molecule experiencing eleven degradation pathways, site reactivity becomes an enormous problem. As mentioned, the large amount of allowable reactions on the many reactive sites creates a combinatorial explosion that leads to intractable reaction networks. Within our system, the amount of species and reactions

reaches a computational limit with regards to memory used. To manage this problem a methodology was employed in which a large unsolvable problem was split into smaller solvable problems. In terms of reaction networks this translates into splitting the reaction chemistries into separate INGen models reacting them separately and merging them into one reaction network. With regards to lignin, a reaction network was created for primary chemistries, and its molecular products fed to a second INGen model that performed secondary chemistries. The two reaction networks were then merged to create a unified representative reaction network. The workflow used for lignin reaction network merging is illustrated in Figure 41.

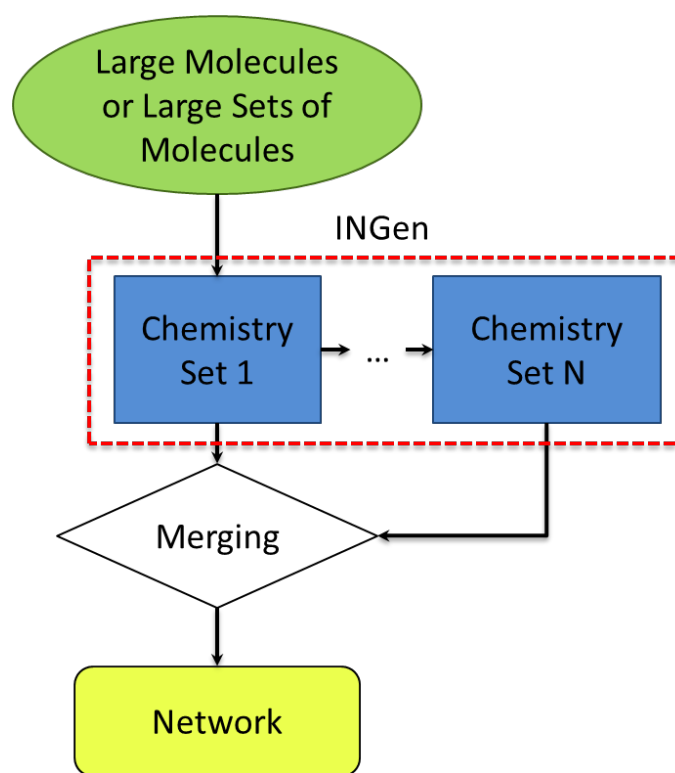


Figure 41: INGen Network Merging Approach

In order to merge reaction networks a united list of species has to be developed from the reaction networks being merged. Each reaction network in INGen has its species naming notation, typically “species” followed a numeric value. This species notation is utilized in the reaction network. This naming convention makes it difficult to discern the identity of the molecules in the reaction network but luckily INGen includes a unique molecule identifier called String Code. String Code was developed by Broadbelt et al. to identify molecular species within the reaction network generator (Broadbelt, Stark, & Klein, 1994). String Code is a list of characters describing atoms and their bonding. In reaction network merging, the String Code is employed to identify all the species in the reaction network and consolidate duplicates to create a merged species list. An example of String Code is shown for butane and isobutane in Figure 42.

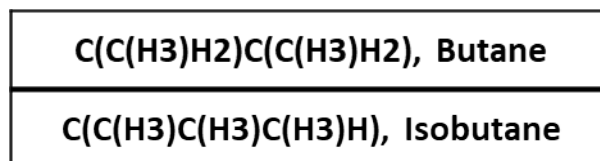


Figure 42: Butane and Isobutane String Code Example

Once a merged species list is created a combined reaction network must be established. To achieve a merged reaction network, duplicate reactions must be consolidated and the merged species list must be utilized to label the species in the new merged reaction network. Therefore, a String Code description of each reaction is utilized to identify duplicates and also recognize species from the merged species list. Figure 43 illustrates the application of String Code to describe a reaction for the isomerization of butane to isobutane. Once the merged species list and reaction network are created they are packaged as a new INGen model.

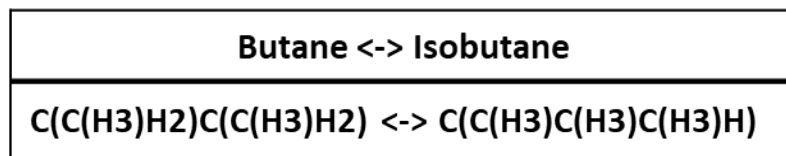


Figure 43: String Code Reaction Description

4.4.3 Initial Conditions and Reactor Conditions

Using a large molecular representation for the feed has an advantage in that the feed is easy to quantify. The only input into to the pyrolysis model is the representative lignin macromolecule. The modeling work presented here mimicked the experimental work by Zhang et al. The model and the experiment both used 5 mg of lignin in a pyroprobe (Zhang, Resende, & Moutsoglou, 2014). The pyroprobe was modeled as an isothermal batch reactor at 600 °C and a pressure of 1 atm. To calculate a concentration of lignin a density of 1.3 g/cc was utilized (Rahbar, 2015).

4.4.4 Model Equations and Kinetics

To model lignin pyrolysis, mole balances represented by ordinary differential equations were solved. The mole balances are a function of the reaction rate laws and the stoichiometry. The following batch reactor mole balance equation (4.4.4-60) was solved for every species i in the lignin pyrolysis system.

$$\frac{dC_i}{dt} = r_i = \sum_j^{rxns} v_{ij}r_j \quad (4.4.4-60)$$

Each rate law includes an Arrhenius rate constant with temperature independent pre-exponential and temperature dependent activation energy. To decrease kinetic parameter complexity, the Bell-Evans-Polanyi linear free energy relationship (LFER) was employed. The LFER concept utilizes the basis that a homologous series of reaction are only differentiated by substituents effects. The substituent effect in the

Bell-Evans-Polanyi LFER is captured by the enthalpy of reaction. A rate constant i can be described by homologous series or reaction family parameters denoted with a j as shown by equation (4.4.4-61). The reaction family includes a pre-exponential, activation energy, and alpha.

$$\begin{aligned} \ln k_{i,j} &= \ln A_j - \frac{(E_j + \alpha_j \Delta H_{rxn,i})}{RT} \\ &= \left(\ln A_j - \frac{E_j}{RT} \right) - \frac{\alpha_j \Delta H_{rxn,i}}{RT} \end{aligned} \quad (4.4.4-61)$$

All the rate laws were modeled as microkinetic rate laws shown in the equation (4.4.4-62) below:

$$r_i = k_{ij} \left(\prod_z^R C_z - \frac{\prod_z^P C_z}{K_{eq_i}} \right) \quad (4.4.4-62)$$

4.4.5 Kinetic Modeling Environment

The modeling of lignin pyrolysis was achieved via the Dynamic Model Builder (DMB). The DMB application is kinetic modeling environment designed to handle large systems. Due to large amount of species and reactions a CPU-GPU ode solver was utilized to model systems with over a 1000 species. The model that had product distribution that matched the GC chromatography work by Zhang et al. takes an average of ~5 seconds to simulate on Desktop PC with a CPU specification of 3.7 Ghz Intel i-7 8700k (“Coffee Lake”) CPU with 8 Gb of available ram (Zhang, Resende, & Moutsoglou, 2014). A high performance cluster (HPC) named Farber was utilized to perform kinetic parameter estimation. The Farber HPC specifications are 2.5 GHz Intel E5-2670 v2 (“Ivy Bridge”) CPU coupled two parallel NVidia K80 GPUs.

4.5 Results and Discussion

Reaction networks were created for the large Freudenberg adapted molecule as well as other modified versions. The use of the reaction network merging approaches allowed for the development of tractable reaction networks as well as pinpointing problematic reaction families. It was observed that the vinyl degradation and dehydrogenation pathways are the underlying culprits in the reaction network size explosion. The Table 35 illustrates a number of reaction networks created from the Freudenberg molecule via the reaction network merge approach. In the Table 35 the vinyl degradation was limited to occur only on reactions with a reactant carbon numbers less than 30. Dehydrogenation was varied to reactant carbons numbers ranging between 20, 25 and 30.

Table 35: Adapted Freudenberg Reaction Networks

Max Carbon Number	Species Number	Reactions Number
20	1048	5493
25	1816	9498
30	8170	53217

It was observed that exponential growth occurs between dehydrogenation reactant carbon numbers 25 and 30. The Freudenberg molecule representation product distribution did not match the gas chromatography work by Zhang et al. (Zhang, Resende, & Moutsoglou, 2014). This is expected because the Freudenberg lignin structure is a hardwood type and the data from experimental work performed by Zhang et al. is for a softwood Kraft lignin. The hardwood contains more synapyl units which once thermal degraded produce larger amounts of syringol and guaiacol

molecules. The gas chromatography by Zhang et al. illustrated a high phenolic content meaning the initial lignin structure that was pyrolyzed contained more coniferyl units. Therefore, modifications were made to the Freudenberg adapted lignin molecule to increase phenolic content while capturing the product distribution. The modified lignin molecule was a ten unit oligomer and is illustrated in Figure 44. Multiple modified lignin structures are illustrated in the appendix section; the one chosen for this work matched the experimental product output by Zhan et al.

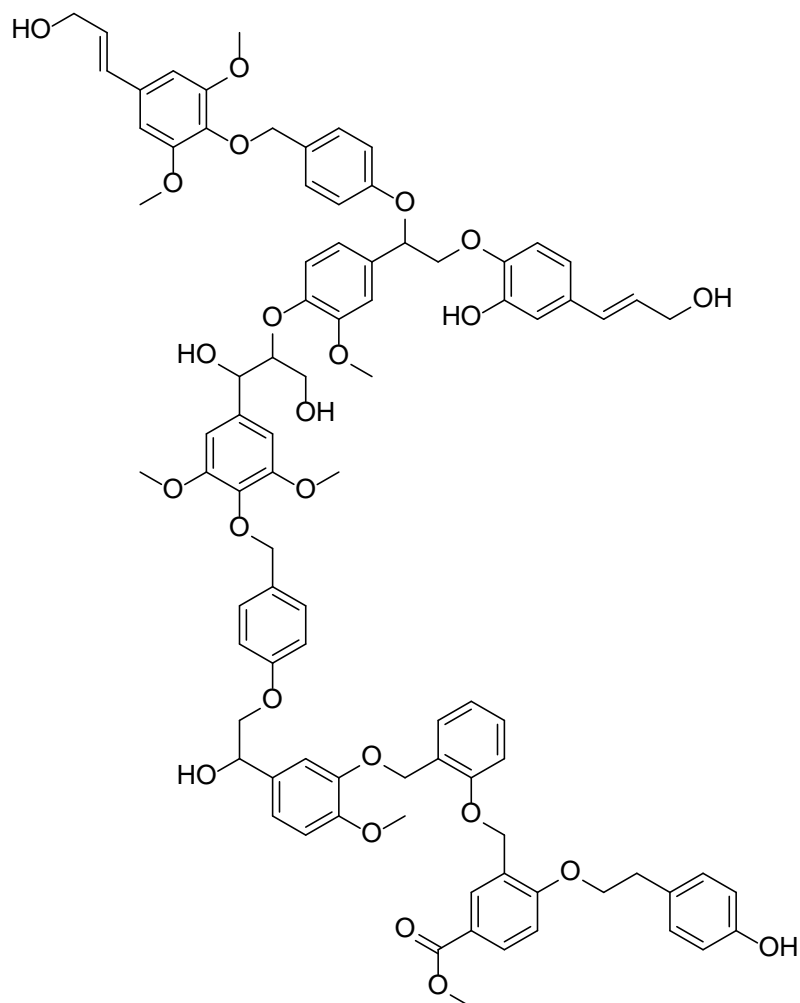


Figure 44: Modified Lignin Structure

The new modified adapted lignin structure was reacted with the same primary and secondary reaction chemistries applied to the Freudenberg lignin producing a reaction network of 628 species and 3329 reactions. The only difference is that the dehydrogenation reactions were excluded to keep the model simulation speed low. The new reaction network was simulated in a kinetic modeling environment and its kinetic parameter estimated to match the experimental results of Zhang et al. The results achieved are shown in Figure 45. It was observed that the lignin structure dictates the product distribution and model output. The simulation matches well with some important components like phenol and syringol but severely under produces vinylguaiacol. The under production of vinylguaiacol is directly tied to the production of guaiacol since vinyl degradation produces guaiacol in the model. Vinyl side chains must be present in the feed structure or the dehydrogenations reaction play a stronger role than expected. Table 36 illustrates the rate constants controlling the lignin pyrolysis model. The vinyl degradation secondary reactions have larger values rate constants than the primary reactions.

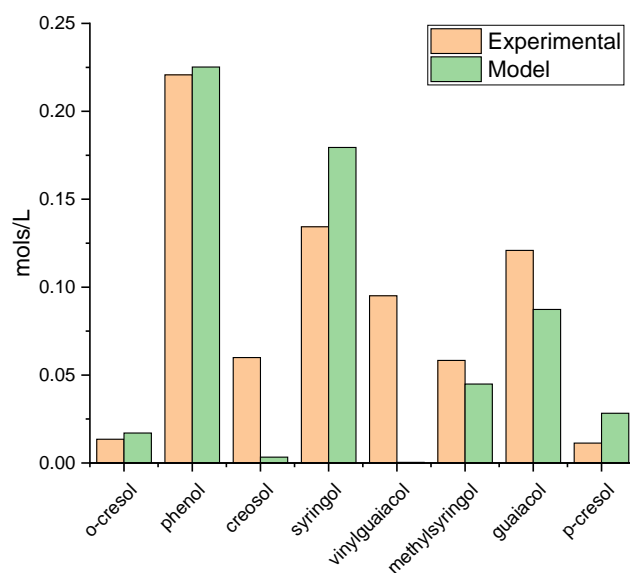


Figure 45: Experiment vs Model Important Species Results

Table 36: Lignin Pyrolysis Rate Constant Values

Reaction Families	$\log_{10} k$
AlphaEtherCracking-Oxygen	-1.41
BetaEtherCracking-Oxygen	-1.65
BetaReaction1-Oxygen	-1.61
BetaReaction2-Oxygen	-1.62
BetaReaction3-Oxygen	-1.63
Decarbonylation	0.01
EnolKetoTautomerization	0.1
KetoEnolTautomerization	0.05
StyreneDegrad1	-0.61
StyreneDegrad2	-0.35
StyreneDegrad3	-0.33

The rank of species in the reaction network is extremely important. If a molecule is produced directly from the lignin molecule or if it is produced by several reactions away it affects its concentration. In the modified lignin structure a higher amount of coniferyl units allows for the high production of phenol from the starting reactant and intermediate reactants. This leads to ample production of phenol that matches the experimental results by Zhang et al. This work illustrates the effect the starting molecule can have on the product distribution.

Through the findings in this work it is proposed that the reactant molecule must be approximated based on the model output. A new tool must be developed that constructs the lignin molecule by monomer units and linkages to form the reactant lignin molecule that matches the experimental output. Using well known chemistry and simulation output the starting component structure can be elucidated. This suggested approach is detailed in Figure 46, the work in this dissertation developed the reaction network and simulation phases.

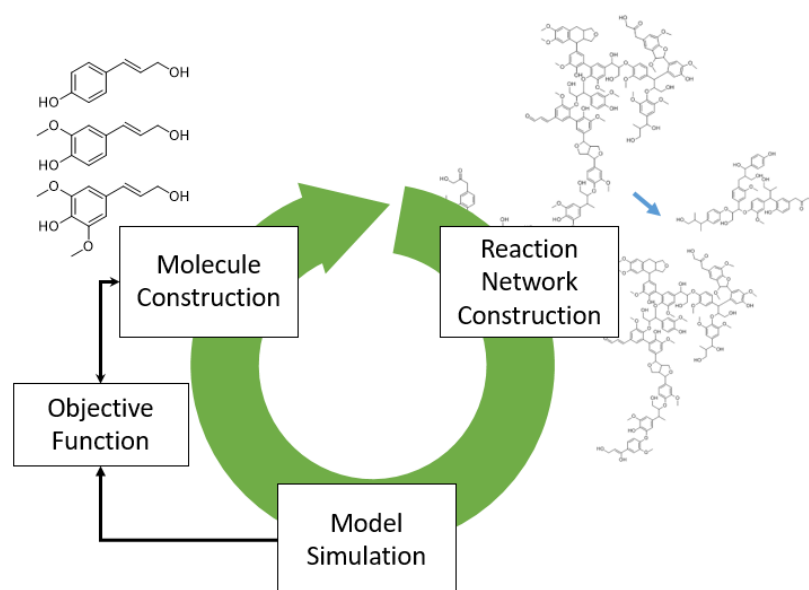


Figure 46: Lignin Simulation Approach

4.6 Summary and Conclusions

- An exhaustive reaction network merging approach was developed to produce tractable reaction networks for large molecule representations.
- Lignin reaction networks were simulated via a kinetic modeling environment named the Dynamic Model Builder.
- To decrease the time of simulation a CPU-GPU combination was used for systems over a 1000 species.
- Molecular-level models were created and simulated for Freudenberg lignin structures and adapted lignin structures.
 - GC data from lignin pyrolysis was utilized to obtain rate constants
 - Phenol is the largest product in the GC experimentation and is produced in the reaction networks from the dealkylation of ortho-cresol and para-cresol.

- Syringol is the second largest product is also produced from the dealkylation of methyl syringol.
- It was observed that the model output is dependent on the lignin structure the work here proposes that the starting material be tuned to obtain the correct product distribution and concentration.

Chapter 5

MOLECULAR-LEVEL SIMULATION OF THERMOGRAVIMETRIC ANALYSIS (TGA): A CELLULOSE PYROLYSIS EXAMPLE

5.1 Abstract

Thermogravimetric analysis (TGA) of cellulose pyrolysis was simulated using a molecular-level kinetic model. A temperature ramp between 373-1073 K was imposed in order to achieve the TGA simulation. A recursive optimization method was utilized to calculate the degree of polymerization of the starting cellulose molecule from literature bulk properties. The model's primary reaction pathways for cellulose decomposition allowed for degradation of the active cellulose molecule by hydrolysis and thermal degradation to create cellobiose, cellobiosan, and glucose. The mechanisms revealed by Agarwal et al. were used to degrade cellobiose (Agarwal, Dauenhauer, Huber, & Auerbach, 2012). An extension of the cellobiose mechanisms were applied to cellobiosan. The reactivity of glucose was captured by the reaction networks proposed in the work of Zhou et al. (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014). Monomer and dimer degradation pathways produced volatiles, furans, aldehydes, ketones, and char through a series of complex reactions. Linear free-energy relationships were applied to minimize the number of model kinetic parameters. Executing the cellulose pyrolysis model in TGA mode provided a prediction of the mass loss. The TGA simulation showed agreement with the experimental trends.

5.2 Introduction

The world's finite petroleum assets and the increasing fuel needs of emerging economies galvanize development of new fuel routes through renewable resources. The fast pyrolysis of lignocellulosic biomass produces bio oil that can be used for fuels and commodity chemicals. The heating rate, max reactor temperature, and residence time of the biomass pyrolysis processes dictate the product distribution. Governing these parameters allows molecular control of the bio-oil composition. Cellulose is the largest component in biomass therefore comprehending its pyrolysis is crucial to optimize the biomass pyrolysis process.

A number of reaction networks have been proposed for the thermal degradation of cellulose. The work by Broido and Nelson illustrated cellulose degradation producing char and volatile tars (Broido & Nelson, 1975). Naturally the modeling of cellulose pyrolysis progressed in complexity to capture more detail. The work by Diebold led to a more complicated scheme to capture formation of gases, volatiles, and tar illustrated in Figure 47 (Diebold, 1994). Inherently this has led to a multitude to reaction models that describe cellulose degradation including the works by Waterloo, Varhegyi-Antal, Wooten-Seeman-Hajaligol (Bradbury, Sakai, & Shafizadeh, 1979) (Antal & Varhegyi, 1995) (Wooten, Seeman, & Hajaligol, 2004). Typically, all the cellulose degradation reaction models involve the formation of active cellulose that further reacts into vapors, tar, and char.

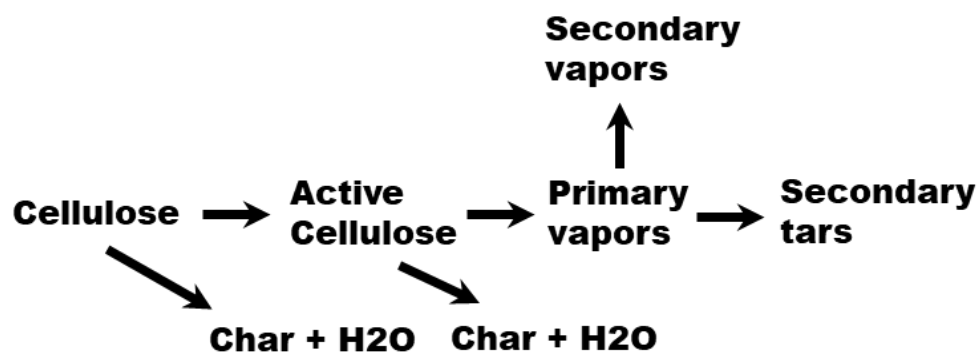


Figure 47: Diebold Global Cellulose Pyrolysis Model (Diebold, 1994)

The effect the heating rate has on product composition has been explored using the aforementioned reaction models. Recently the work by Lin et al. used the Diebold reaction model in Figure 47 to simulate cellulose thermogravimetric analysis (TGA) (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009). The TGA experiment measures the weight of sample while the surrounding temperature is increased and gasses are whisked away. The variation of heating rates during the temperature increase can lead to a quantitative and qualitative understanding of the activation energies. To adequately simulate TGA from a lumped scheme Lin et al. introduced a temperature lag effect. The lumped nature of the Diebold reaction model required this additional temperature lag variable. The work here focuses on the describing the TGA experiment through fundamental kinetic parameters in the Arrhenius equation.

A number of molecular-level models have been developed for the pyrolysis of cellulose. The work by Zhou et al. provided a detailed mechanistic model of cellulose pyrolysis illustrating the capability of molecular-level modeling (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014). The work presented here further builds upon the existing cellulose pyrolysis paradigm. A pathways molecular-level model for cellulose pyrolysis was developed and used to simulate TGA without a temperature lag effect.

A molecular-level resolution of the TGA details the underlying mechanisms and provides the insights to control the product distribution.

5.3 Computational Methods

To develop a molecular-level model of cellulose pyrolysis a molecular representation of cellulose was approximated. The molecular representation was reacted to produce a reaction network. The reaction network was utilized to model cellulose pyrolysis in a kinetic modeling environment. An approach was developed to simulate the TGA experiment from molecular-level information based on phase equilibria. The methods section will begin by detailing the approach used to approximate the cellulose molecule.

5.3.1 Cellulose Molecular Representation

Cellulose is a polymer composed of glucose monomers. This work approximates the cellulose polymer as a linear chain of monomers with a Flory linear polymer distribution shown in equation (5.3.1-63) below for n -mers (Flory, 1934)

$$x_n = (1 - p) * p^{n-1} \quad (5.3.1-63)$$

In the Flory polymer distribution equation above p is the extent of polymerization and is related to the average degree of polymerization (\bar{X}_n) by using the equation (5.3.1-64) below.

$$\bar{X}_n = 1 - \frac{1}{p} \quad (5.3.1-64)$$

An iterative process was used to approximate the degree of polymerization from elemental analysis using the same method found in the work by Horton et al. (Horton, Mohr, Zhang, Petrocelli, & Klein, 2016). The experimental elemental analysis information applied to approximate the cellulose molecule came from the work by Lin et al. and gave a degree of polymerization of three. Therefore the cellulose polymer can be approximated as a cellotriose based on experimental information, illustrated in the Figure 48. Cellotriose is a glucose trimer. It is hypothesized that cellulose extraction from biomass degrades the cellulose polymer composition decreasing its chain length to a trimer length.

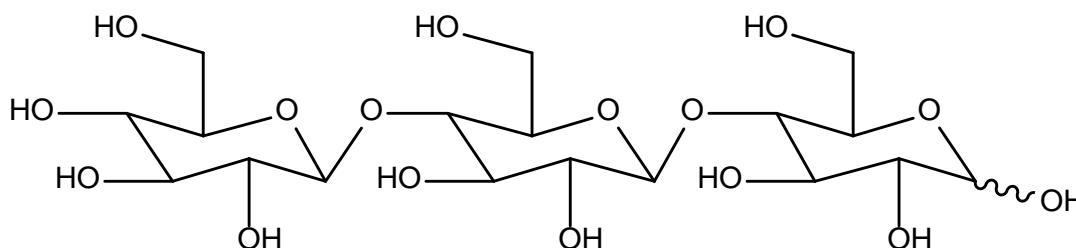


Figure 48: Cellotriose Molecule

5.3.2 Reaction Network Construction

The reaction network construction relied heavily on chemistries reported in literature. Primarily focusing on the chemistries supplied by Agarwal et al. and Zhou et al. for the decomposition of cellobiose and glucose (Agarwal, Dauenhauer, Huber, & Auerbach, 2012) (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014) and Bollas et al. and Norinaga et al. for char formation.

5.3.2.1 Cellotriose Thermal Degradation Pathways

In this pyrolysis model cellotriose is degraded via two pathways: thermohydrolysis and thermal degradation. Both of the aforementioned pathways form glucose monomer and dimer molecules. The dimers formed are cellobiose and cellobiosan. The two degradation routes for the pyrolysis of cellotriose are illustrated in Figure 49, where [1] depicts thermohydrolysis and [2] thermal degradation.

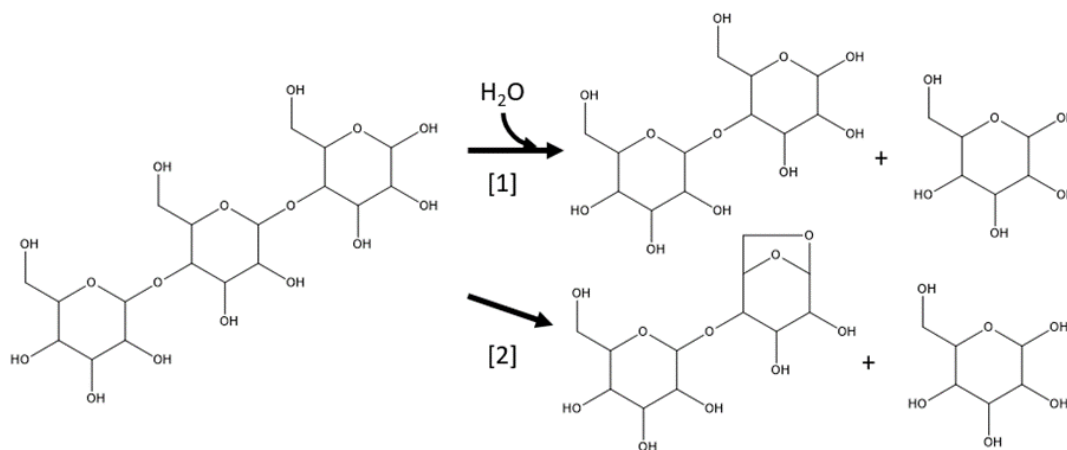


Figure 49: Cellotriose Degradation Pathways: 1. Thermohydrolysis 2. Thermal degradation

Cellobiose, cellobiosan, and glucose degrade into smaller molecules or polymerize to form larger char molecules. The following chemistry sections will focus on the degradation of the dimers followed by the glucose degradation chemistry.

5.3.2.2 Cellobiose and Cellobiosan Thermal Degradation Pathways

The cellobiose degradation chemistry was revealed by the molecular dynamics work of Agarwal et al. (Agarwal, Dauenhauer, Huber, & Auerbach, 2012). The work by Agarwal et al. focused primarily on cellobiose chemistry. The work presented here extends that chemistry to cellobiosan. The Figure 50, Figure 51, and Figure 52 show

the cellobiose and cellobiosan degradation pathways to form formic acid, enol fragments, hydroxymethylfurfural precursors, glucose and levoglucosan.

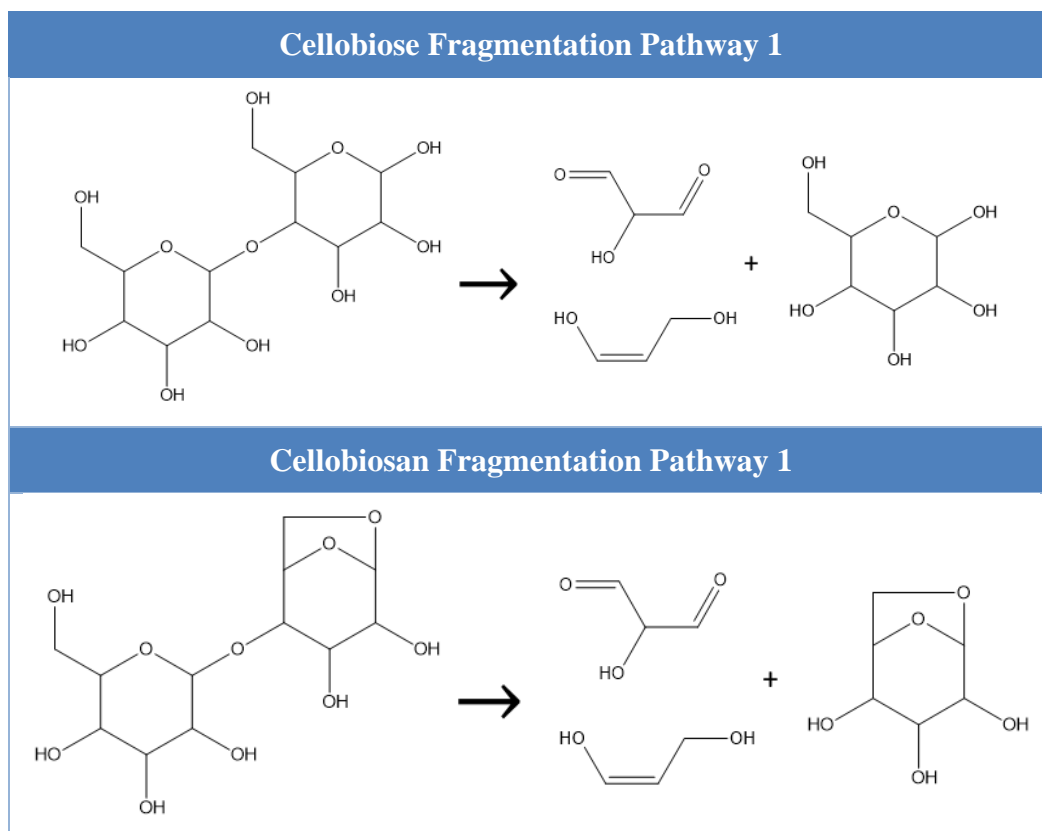


Figure 50: Cellobiose and Cellobiosan Fragmentation Pathway to form Formic acid

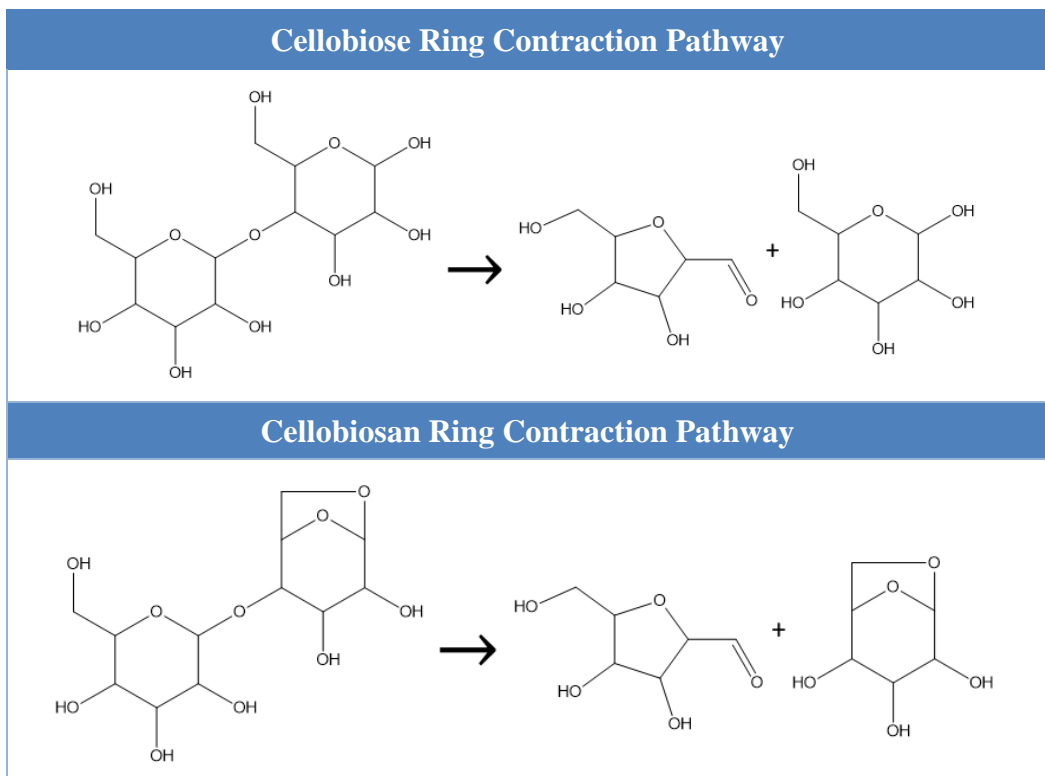


Figure 51: Cellulose and Cellobiosan Ring Contraction to form HMF precursor

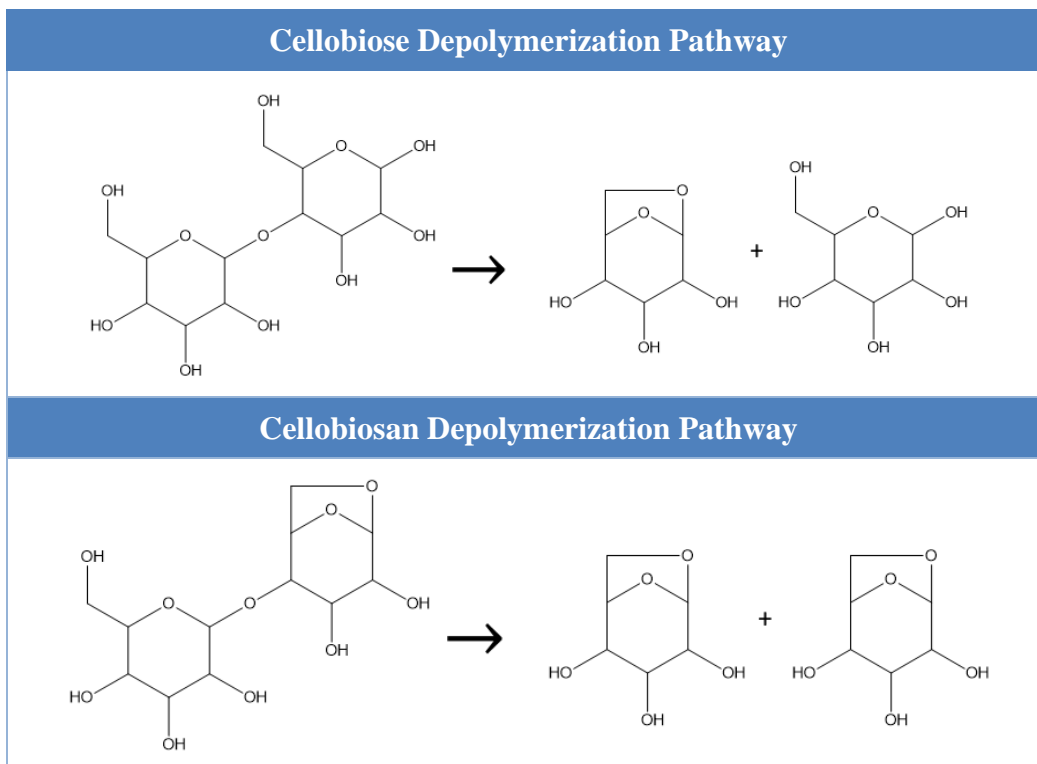


Figure 52: Cellobiose and Cellobiosan Depolymerization Pathway

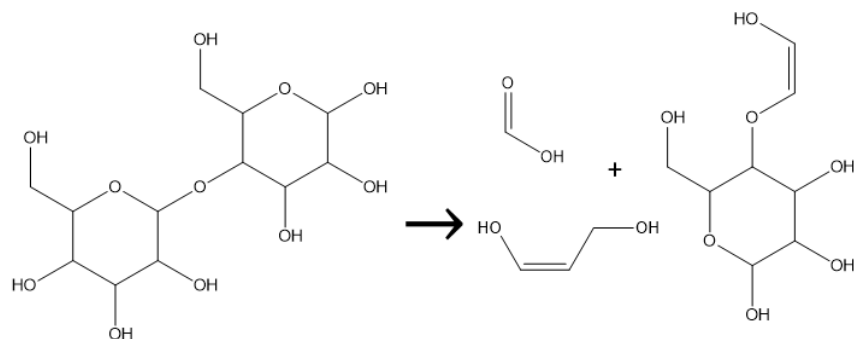
Notice that smaller molecules are produced from the degradation of the glucose dimer molecules. This is extremely important because experimental gas chromatography shows significant quantities of HMF and formic acid (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009). Their larger production may signify an early rank in the reaction network. Tuning of the model on experimental results can help determine which is more likely: production from large molecules or higher rank intermediates. Including these pathways allows for exploitation of experimental information to determine the correct mechanisms.

5.3.2.3 Cellobiose and Cellobiosan Char Formation Pathways

So far we have discussed the degradation of cellobiose and cellobiosan but char pathways from these molecules are also implemented in the reaction network.

Based on the work by Agarwal et al. the char precursor molecules can be formed from cellobiose and cellobiosan. The char precursor pathways are shown in Figure 54. In the reaction network the char precursor molecules are formed and further reacted with other like char precursor molecules to form higher molecular weight char molecules. A lumped effect is introduced in order to decrease the need for subsequent polymerizations reactions that form larger and larger char molecules. Up to this point the degradation and polymerization chemistries of cellobiose and cellobiosan has been covered, the upcoming section will illustrate the degradation pathways of glucose based on the work by Zhou et al.

Cellobiose Ring Fragmentation Pathway



Cellobiosan Ring Fragmentation Pathway

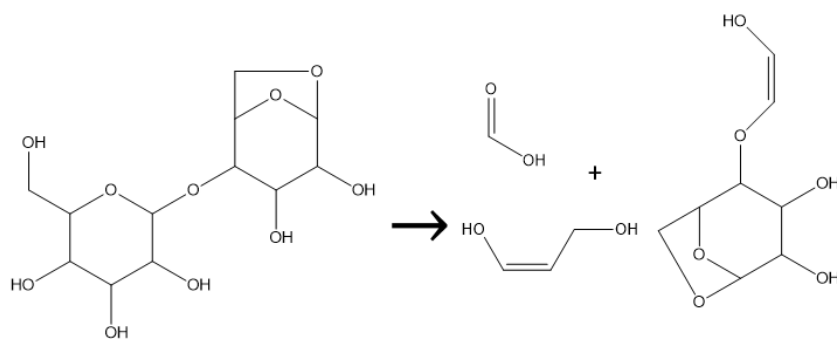


Figure 53: Cellobiose and Cellobiosan Ring Fragmentation Pathway to form Formic Acid

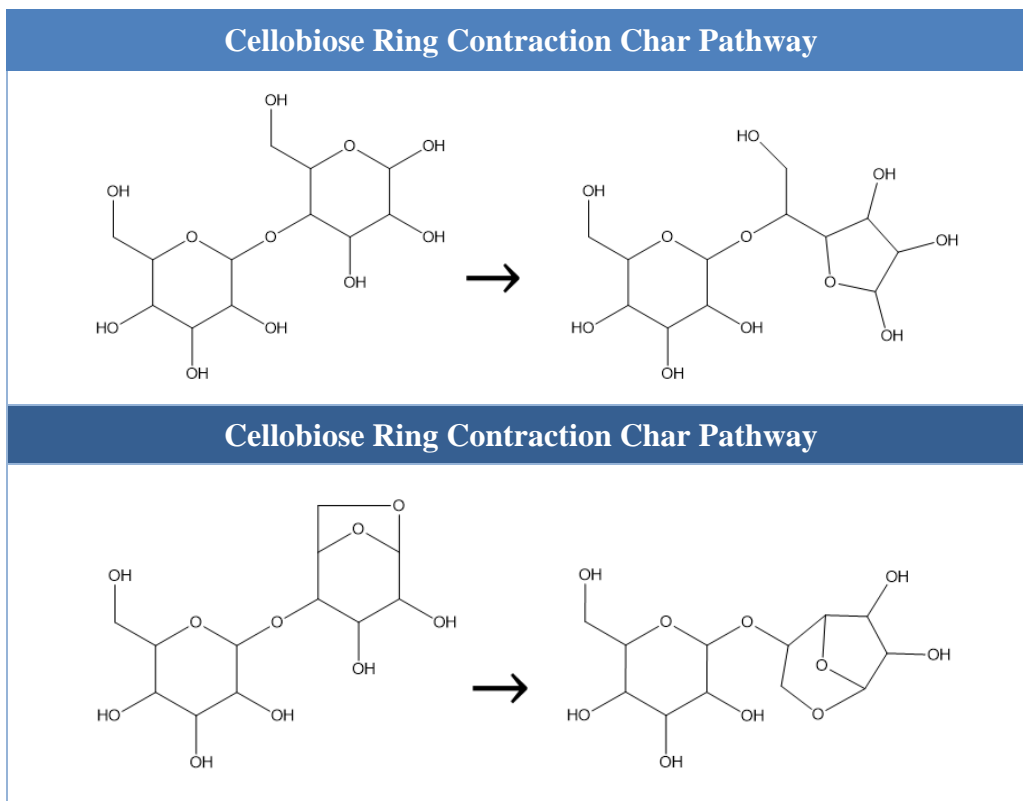


Figure 54: Char Precursor Formation from Cellobiose and Cellobiosan

5.3.2.4 Glucose Thermal Degradation Pathways

The chemistry employed to model the thermal degradation of cellulose came from the work by Zhou et al. An adaption of their reaction network is illustrated by Figure 55, Figure 56, and Figure 57. The glucose molecules can thermally degrade via two pathways ring opening and dehydration. Ring opening produces a D-glucose which has many degradation pathways illustrated in Figure 56 and Figure 57. The dehydration of glucose produces levoglucosan (LGA) and a dehydrated glucose molecule. In this model LGA is a terminal product due its high experimental yield (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009). At higher temperatures the LGA molecule can also further degrade but those pathways were not included in this

work (Xiaolei Zhang, 2012). The linear D-glucose and dehydrated glucose can experience a series of reactions to further degrade into smaller molecules. The types of reactions that can occur are dehydration, retro Diels-Alder, retro aldol condensation, enol-keto tautomerization, isomerization, Grob fragmentation and decarbonylation. The dehydration reaction removes water from a molecule and produces double carbon to carbon bond. The retro Diels-Alder reaction forms a diene and a dienophile through concerted mechanisms. In this reaction network the retro Diels-Alder reaction breaks rings open. The retro aldol reaction produces a ketone enolate and aldehyde group from an aldol group. An aldol group involves a ketone/aldehyde and alcohol group two carbon atoms away. The Grob fragmentation breaks aliphatic chains and rings to fragments via elimination reactions (Paine III, Pithawala, & Naworal, 2008). The decarbonylation reactions remove aldehyde and carboxylic acid groups in the form of carbon monoxide.

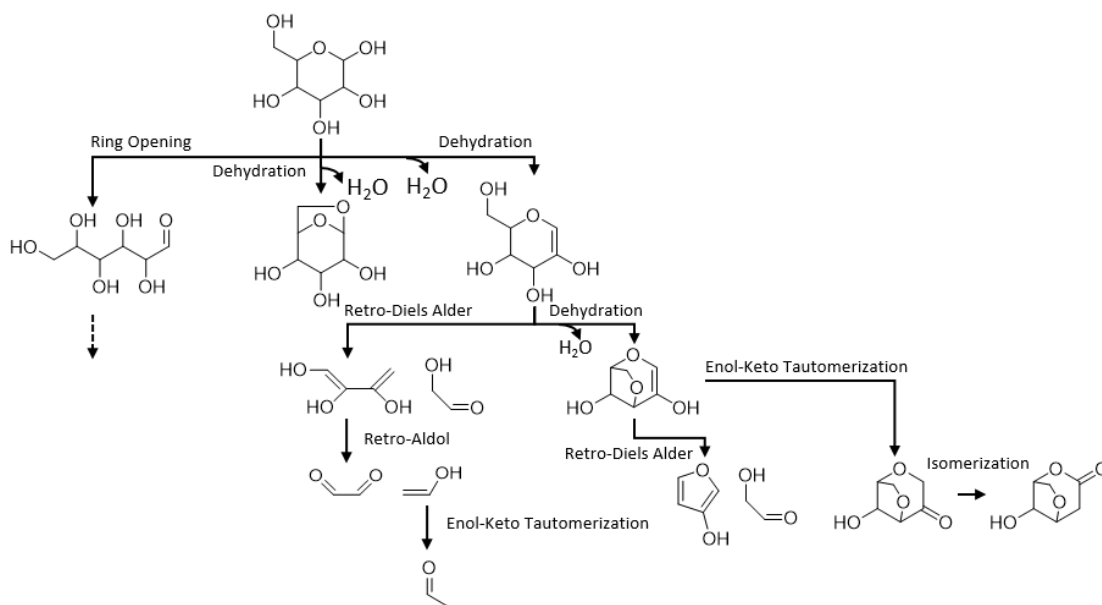


Figure 55: Glucose Degradation Pathways (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014))

Bollas is propadiene. The work presented here utilizes the pathways defined in the work by Paine et al. to form ketene from D-glucose (Paine III, Pithawala, & Naworal, 2008). The ketene molecule can thermally decompose to produce propadiene and ethylene (Jackson, 1968). Propadiene can form propyne through an isomerization reaction. Propyne and ethylene are extremely reactive and react with a host of small molecules to form aromatics. Ethylene can also dehydrogenate to form acetylene which forms large char precursors like cyclopentadiene when reacted with propyne (Norinaga, et al., 2014). The work by Bollas et al. requires the precursor propene which is a product of the hydrogenation of propyne. Propene can react with furfural to form methylbenzaldehyde. Methylbenzaldehyde can experience a series of aldol condensation reactions to become a large char molecule. Bollas et al. also introduced a char pathway that utilizes toluene the product of cyclopentadiene and acetylene. The work by Norinaga et al. also described the formation of naphthalene from bimolecular reaction of 2 cyclopentadiene's. The toluene molecule can also be produced from Diels-Alder reactions with furan groups. Toluene experiences self-alkylation reactions to produce larger char aromatic compounds as shown in Figure 58.

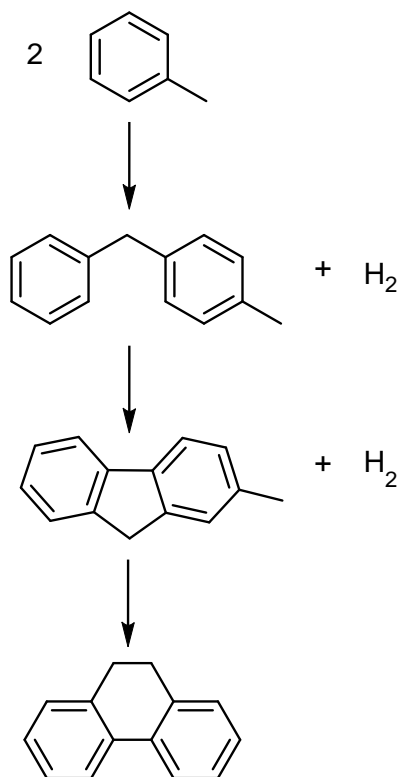


Figure 58: Toluene Self-Alkylation Char Pathway

5.3.3 Model Equations and Kinetics

The molecular-level model of cellulose pyrolysis includes mole balances in the form of ordinary differential equations. The mole balances are a function of reactions and stoichiometry. The mole balance utilized to model cellulose pyrolysis was batch reactor shown equation (5.3.3-65) below.

$$\frac{dC_i}{dt} = r_i = \sum_j^{rxns} v_{ij} r_j \quad (5.3.3-65)$$

The reaction rate laws are of the microkinetic nature shown by equation (5.3.3-66) below.

$$r_i = k_{ij} \left(\prod_z^R C_z - \frac{\prod_z^P C_z}{K_{eq_i}} \right) \quad (5.3.3-66)$$

The rate constant in the rate law are of the Arrhenius form, which include a pre-exponential and activation energy values. Bell-Evans-Polanyi linear free energy relationships (LFER), equation (5.3.3-67), were utilized to reduce the parameter complexity (Bell, 1936) (Evans & Polanyi, 1935).

$$\ln k_{i,j} = \ln A_j - \frac{(E_j + \alpha_j \Delta H_{rxn,i})}{RT} = \left(\ln A_j - \frac{E_j}{RT} \right) - \frac{\alpha_j \Delta H_{rxn,i}}{RT} \quad (5.3.3-67)$$

The LFER concept reduces parameters by describing a homologous series of reactions as reaction families. LFER describes like reactions which make up one reaction family by a set of three parameters: pre-exponential, activation energy, and alpha. The substituent effects are controlled by the enthalpy of reaction.

5.3.4 Thermogravimetric Analysis (TGA) Simulation Approach

Previous methods for simulating TGA have used modifiers on the rate constants to capture the loss of mass caused by phase changes (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009). The rate constant modifiers are necessary when using lumped components. This study utilizes molecular mechanisms therefore exploits molecular properties to describe phase equilibria. In this section an iterative molecular approach to simulating TGA will be outlined building upon the approach of Horton et al. (Horton, et al., 2016). In the TGA experiment a sample is weighed as temperature is ramped up. As the temperature increases reactions begin to occur, producing smaller molecules while simultaneously phase changes transpire. The sample weight decreases due to the reactions and phase changes which generate volatiles that leave the system

as air is flowed over the sample. To mimic this complexity there is a need to simulate phase change in the system. Computationally this can be achieved by using the equilibrium flash evaporation calculations. Flash evaporation for multi component systems can be calculated by the Rachford-Rice equation (5.3.1-68) shown below. In the Rachford-Rice equation a flash calculation must be achieved for each molecular species i in the feed. The equation involves the liquid fraction x_i of the feed and equilibrium constants K_i .

$$\sum_i \frac{x_i(K_i - 1)}{1 + v_f(K_i - 1)} = 0 \quad (5.3.1-68)$$

The equilibrium constants can be approximated by Rault's Law which involve vapor pressures for each component i . To calculate the vapor pressures of each species the Lee-Kessler method (Shown below in equations below) was applied with molecular information from group contribution methods. For each molecule group contribution methods were used to calculate the critical temperature (T_c), critical pressure (P_c), and the acentric factor (ω). Newton-Raphson bisection method was utilized to approximate the vapor fraction v_f by finding the root of the Rachford-Rice equation (5.3.1-68) above.

$$\ln \frac{P_{sat}}{P_c} = f^0 + \omega f^1 \quad (5.3.1-69)$$

$$f^0 = 5.92714 + \frac{6.09648 * T_c}{T} - 1.28862 * \ln \frac{T}{T_c} + 0.169347 * \left(\frac{T}{T_c}\right)^6 \quad (5.3.1-70)$$

$$f^1 = 15.2518 + \frac{15.6875 * T_c}{T} - 13.4721 * \ln \frac{T}{T_c} + 0.43577 * \left(\frac{T}{T_c}\right)^6 \quad (5.3.1-71)$$

Once the equilibrium flash was implemented into the kinetic modeling environment a workflow was developed to simulate TGA. Figure 59 illustrates the TGA approach which utilizes the equilibrium flash to calculate gas and liquid concentrations as function of pressure and temperature. In the approach a feed concentration is fed to the pathways cellulose pyrolysis model which runs at temperature T, after the reactor there is a flash which separates the components into gas and liquid phases. The gas phase is discarded while the mass of the liquid phase is calculated and saved. The temperature is then increases based on the heating rate. The cellulose pyrolysis model then runs at a new temperature and the iterative process proceeds until the final temperature is achieved.

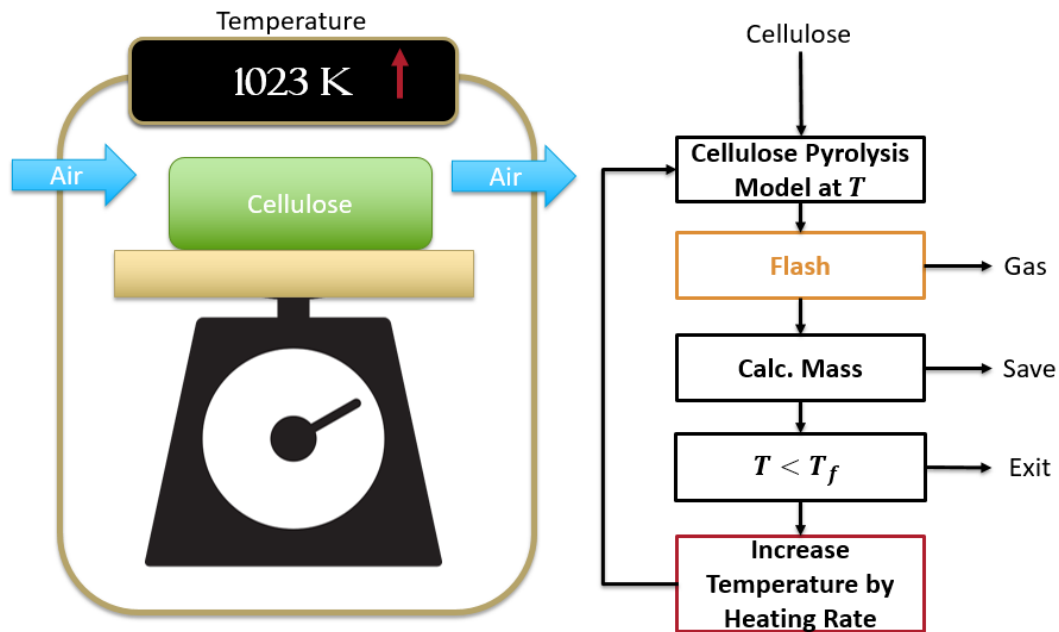


Figure 59: Thermogravimetric Analysis (TGA) Simulation Approach

5.3.5 Kinetic Modeling Environment

The pyrolysis of cellulose was modeled in an object oriented kinetic modeling environment named Dynamic Model Builder (DMB). The object oriented nature allows for simple model development and additions. The equilibrium flash necessary to simulate TGA is an object of the framework. This object can be accessed by other models to perform flash operations or be used in the ordinary differential equation numerical integration for multi-phase systems. In this case the flash was added after the molecular-level kinetic model and used in a recursive method to simulate TGA.

The equilibrium flash calculations are designated for two phase systems that involve liquid and gas, to account for solid molecules in the equilibrium flash calculations the char molecule vapor pressures were decreased to values close to zero. Decreasing the vapor pressures assured that the char molecules did not reenter the gas phase.

5.4 Results and Discussion

A pathways based molecular-level kinetic model of cellulose pyrolysis was developed with 123 reactions and 113 species. This translates to 113 ordinary differential equations that describe the mass balances of the system. 63 reaction families were used describe the kinetic parameters in cellulose pyrolysis. A total of 126 independent parameters are utilized in the model. The values for pre-exponentials and activation energies were based on the work by Zhou et al. (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014). The alpha values for the reaction families were set to zero. Simulation of TGA showed agreement with the work by Lin et al. without a need for extraneous temperature lag effect as shown in Figure 60. The TGA simulation captured the cellulose degradation as temperature increased and the subsequent char formation at higher temperatures. As the heating rate was increased there was marked changes in the amount of char formed, decreasing with increased heating rates. The cellulose hydrolysis and thermal degradation activation energies were 34 kcal/mol to achieve results matching the work by Lin et al. the model required char activation energies averaging ~21 kcal/mol. This differentiates from the work by Zhou et al. who presented activation energies for char formation of 40 kcal/mol. The utilization of the equilibrium flash allowed for a progression of vapor fraction to be tracked as the heating rate is increased, illustrated in Figure 61. As the heating rate is increased there is a shift in vapor fraction which is analogous to the shift seen in TGA. This illustrates the strong contribution phase equilibrium has on the TGA experiment.

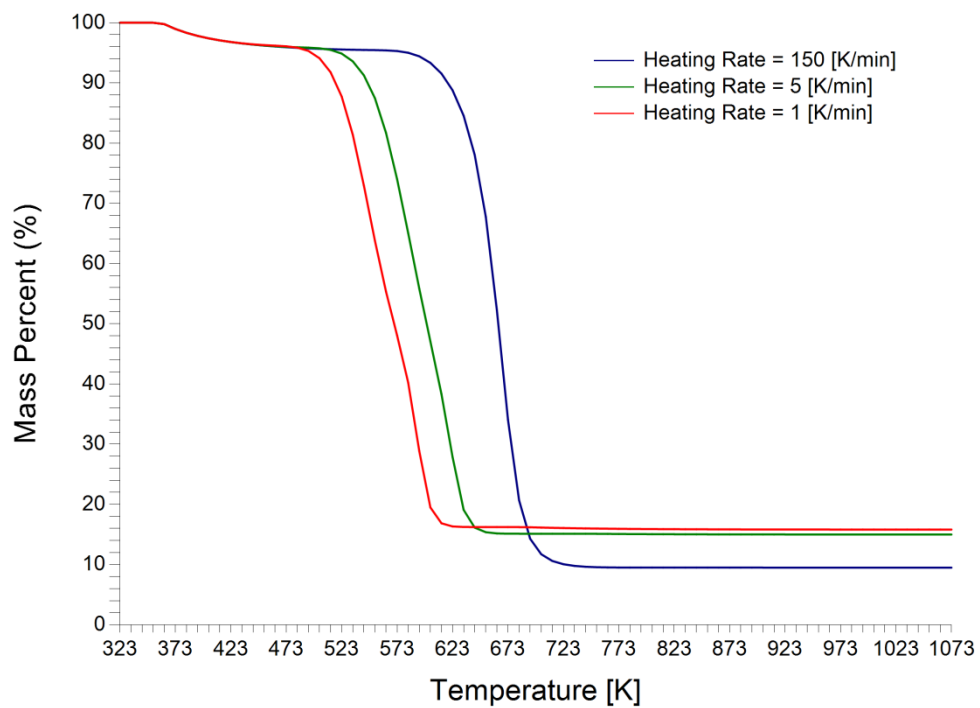


Figure 60: Thermogravimetric Analysis (TGA) Simulation Result

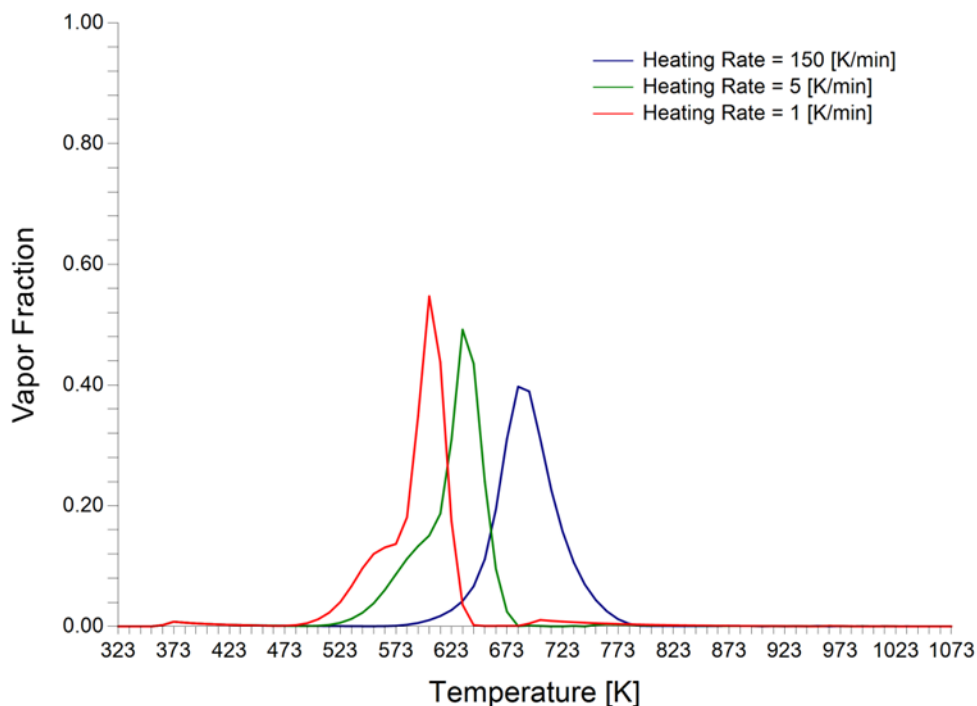


Figure 61: Vapor Fraction vs Temperature Result

It was observed that cellotriose degradation and char formation reaction families control the structure of the TGA curve. This explains why others have been able to simulate TGA using lumped models (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009) (Bellan & Miller, 2010) (Antal & Varhegyi, 1995). As the heating rate increases rapidly the lower activation energies determine the kinetics of the system. The dominance cellotriose degradation pathways and char pathways is illustrated by their smaller activation energies when compared to secondary reaction chemistries that degrade cellobiose and cellobiosan as illustrated in Table 37. The system can therefore be simply depicted as cellulose going to volatiles and char from a TGA viewpoint. The cellulose pyrolysis kinetic parameters in Table 37 were compiled by Zhou et al. from experimental and quantum mechanics work (Zhou, Nolte, Mayes, Shanks, &

Broadbelt, 2014). The work presented here utilized the kinetic parameters by Zhou et al. for the decomposition of cellulose. To achieve the curvature of the TGA curve char forming reactions were altered to have lower activation energy barriers.

Table 37: Cellulose Pyrolysis Kinetic Parameters (Adapted from (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014))

Reaction	Citation Label	log10A	E [kcal/mol]
Hydrolysis	Not Present	14	34
Retro-Diels–Alder	5, 9, 61	15.02938	55
Dehydration	1	12.8451	48.2
	4	15.69897	60
	8	15.30103	54
	14	14.69897	50
	22	15.17609	49.7
	23, 31, 37, 52	12.8893	38.9
	24	15.69897	56.5
	32, 38	15.69897	57.5
	41	9	29
	45	9	27.6
	53	15.69897	61
	55, 63, 64	15.30103	50.5
	58	13.47712	47.6
	60	12.69897	47.6
Ring-opening/closing	2	13.59106	47
	3	10.91908	34
	12	10.87506	32.7
	13	13.63347	42
	20	11.77815	37.6
	21	13.20412	42.1
	28	11.82282	31.7
	29	14.82282	41.7
	35, 50	12.10721	37.6
	36, 51	13.52763	42.1
Retro-aldol	56	13.33244	42
	57	13.97497	46
	6, 62	12.75128	39.2
	16	12.35984	36.5

	18	12.57287	32.4
	44, 66, 69	12	39.7
	48	12	39.2
Enol-keto tautomerization	10, 17, 34, 54, 59, 65, 68	13.37658	0
	7, 42, 43, 46, 49	15.69897	57
Isomerization	11, 27, 40	13.37658	46.8
	19	10.63347	33.4
Cyclic/Grob fragmentation	15, 30	14.90309	52.5
	33, 39	15.17609	52.5
Decarbonylation	25, 26, 47, 67	15.69897	62
Char formation	Not Present	10.81291	21

The molecular nature of the TGA simulation allowed for the composition of the melt to be analyzed. The melt is the reducing mass during pyrolysis. The molecular interpretation of the melt is shown in Figure 62. To achieve accurate dynamics due to heating rates this work proposes that the gasses which are removed from the system be analyzed through gas chromatography. This can help obtain a grasp on the apparent effect the heating rate has on the melt.

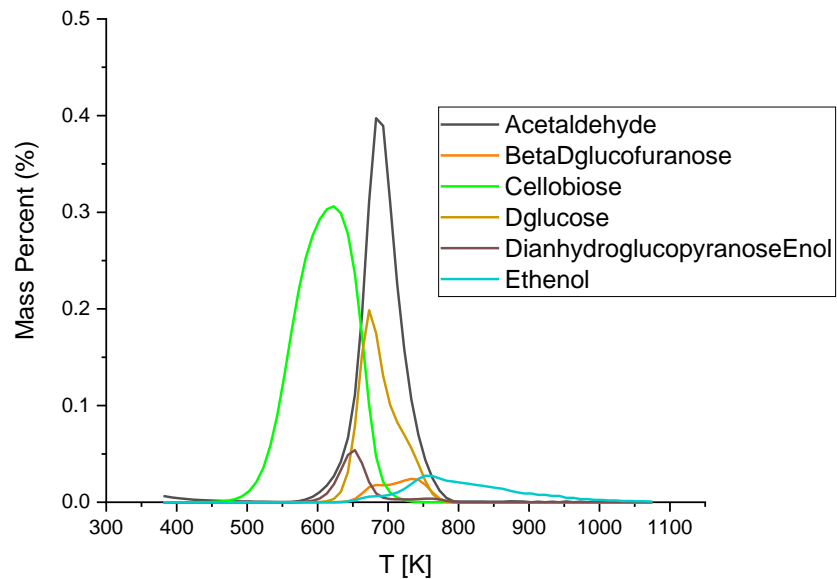


Figure 62: Melt Dynamics as a Function of Temperature

5.5 Summary and Conclusions

- A pathways molecular-level model for cellulose pyrolysis was developed.
- An approach was established to simulate TGA which captures chemical kinetics and phase equilibria simultaneously.
- The kinetic parameters found in literature by Zhou et al. were used to simulate cellulose pyrolysis.
- TGA simulation showed agreement with the experimental work by Lin et al. based only on activation energies and no temperature lag.
- The average char activation energy of ~ 21 kcal/mol was found to be smaller than that found in literature at 40 kcal/mol.

Chapter 6

DEVELOPMENT OF PARALLEL KINETIC PARAMETER ESTIMATION STRATEGIES FOR COMPLEX MOLECULAR-LEVEL SIMULATIONS

Juan C. Lucio-Vega and Michael T. Klein

Department of Chemical and Biomolecular Engineering, University of Delaware

6.1 Abstract

A molecular-level kinetic model has been developed for gas phase diesel hydrocracking on a bifunctional metal acid catalyst. The diesel feed was modeled as a complex mixture of paraffins, multi-branched isoparaffins, naphthenics, and aromatics. Linear free energy relationships in the Bell-Evans-Polyani form were applied to minimize the number of model kinetic parameters (Bell, 1936) (Evans & Polanyi, 1935). The generated model was utilized to develop parallel tuning methodology for robust kinetic parameter estimation.

6.2 Introduction

The increasing utilization of heavier crude oils in refineries marks the hydrocracking process as essential in the production of middle and to light distillates. Typically, hydrocrackers are utilized for feed upgrading, removing impurities and cracking molecules to produce valuable fractions. The hydrocracking unit can be utilized to process a variety of feeds like gas oil, residues, lube oils, middle distillates and light gases. The feed variability of the hydrocracking unit places it strongly in an economically driven refinery atmosphere.

Ideal hydrocracking chemistry involves a bifunctional catalyst comprising of a noble metal and a Brønsted acid (Weitkamp, 2012). The metal functionality dehydrogenates saturated compounds to olefins which diffuse to nearby acid sites to isomerize or experience carbon-carbon bond scission via carbocation reactive intermediates. Due to the hydrocracking unit's widespread usage and importance, a fundamental understanding is crucial to optimize the process. The kinetic modeling of petroleum processes can lead to a comprehensive understanding of the underlying chemical mechanisms which are necessary for process design.

The kinetic modeling of hydrocracking units has progressively moved towards molecular resolutions. One of the primary hydrocracking feeds, vacuum gas oil (VGO), is difficult to quantify experimentally due to its large molecular nature. Nonetheless, this has not impeded the development of molecular-level kinetic models for the process (Kumar, 2007) (Becker, Serrand, Celse, Guillaume, & Dulot, 2017). For a heavy distillate like VGO, the feed must be reconstructed from a limited amount of experiments to achieve molecular-level resolution. An accurate molecular description of the feed and products however can reduce the uncertainty in the model. Diesel fractions can be heavily quantified with molecular detail via modern analytical techniques like 2D gas chromatography. Modeling diesel and extending the knowledge gained on the general underlying chemical mechanism of diesel hydrocracking can provide insights into larger feeds.

In the past, kinetic models of diesel hydrocracking have focused on specific components in a diesel feed tailoring to specific chemistries to decrease model complexity (Froment, Depauw, & Vanrysselberghe, 1994) (Quann & Jaffe, 1996) (Martens, Marina, Martens, Jacobs, & Baron, 2000) (Froment J. M., 2006). The work

presented here details the development of a pathways molecular-level model for diesel hydrocracking. A focus was placed on developing parallel tuning methodology.

6.3 Computational Methods

To develop a pathways based molecular-level kinetic model, a reaction network was constructed based on the species found in a diesel feed. The feed composition was reconstructed utilizing probability density functions (PDFs) that describe paraffin, isoparaffin, naphthenic and aromatic content through molecular information from the total species list. The reaction network and feed compositions were employed in a kinetic modeling environment to simulate and tune diesel hydrocracking model.

6.3.1 Reaction Network Construction

The reaction network utilized in this model was created via an in-house reaction network generation software tool named the Interactive Network Generator (INGen) (Bennett, 2009) (Wei, et al., 2008). INGen produces a reaction network by applying a set of user specified reaction rules to a molecular species. It relies on the basis that a molecule can be represented computationally by a bond-electron matrix. The reactive moiety of the molecule can be described as a reduced reactant matrix only including the atoms and bonds participating in the reaction. The bond breaking and forming algorithm can be described by a reaction matrix. Simple matrix algebra between the reduced reactant matrix and the reaction matrix produces a reduced product matrix. Through this iterative approach a reaction network can be developed. The reaction matrix and reactive subgroup apply to multiple molecules and are independent of overall structure

Bifunctional hydrocracking chemistry was employed for both acid and metal catalyzed chemistries. The product composition is primarily determined by the acidic chemistry. Acid catalyzed chemistries in the model include dealkylation, isomerization, β -scission, side chain cracking, ring opening and ring isomerization. Metal catalyzed chemistries focused on the hydrogenation of aromatic content. The diesel hydrocracking reaction network presented here was constructed by making two independent reaction networks: one for ringed components and one for acyclic saturated compounds. After each reaction network was built autonomously, they were merged into one reaction network via reaction network merging approaches. Reaction network merging approaches were employed to minimize the aliphatic saturated compound reaction network. Isomerization of paraffins to mono, di and tri branched isoparaffin can cause a high number of reactions and species numbers. To keep the isoparaffin species manageable, on a carbon number basis, three species represented mono, di and tri branched configurations for one carbon number. At each carbon number, one molecule represented all mono-branched configurations. To capture reactivity of all mono-branched molecules a product distribution was created that matched Type C β -scission from varying branch locations. This approach was completed from carbon numbers 5 to 30 and extended to di and tri branched isoparaffins including Type A and Type B β -scission. The minimization technique decreased the number of isoparaffins to 90 while capturing the acid cracking reactivity. Consequently, two reaction networks were created independently due to the varying approaches, in the following section the chemistry of the acyclic saturated compounds will be discussed.

Paraffin and isoparaffin acid catalyzed chemistries composed the acyclic saturated compounds reaction network. Normal paraffins up to carbon number 30 were isomerized to mono, di, and tri branched isoparaffins. The branched isoparaffins experienced Type A, Type B, and Type C β -scission chemistries thermally cracking to normal paraffin and isoparaffin products. In the reaction network, Type A β -scission thermally cracked a tri-branched isoparaffin to produce two mono-branched isoparaffins. The Type B β -scission thermally cracks di-branched isoparaffins to an isoparaffin and paraffin product. Type C β -scission thermally cracks mono-branched isoparaffins to two normal paraffins. The normal paraffin and isoparaffin chemistries are illustrated in Figure 63, Figure 64, Figure 65, and Figure 66. In the figures containing β -scission reactions, the mechanisms are shown above the pathway.

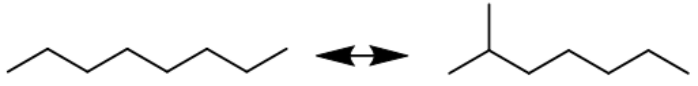
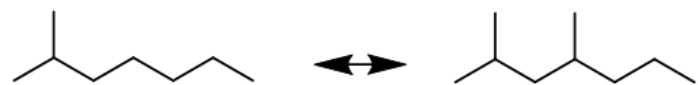

Reaction Type		Reaction Matrix			
	\rightleftharpoons				
	\rightleftharpoons				
	\rightleftharpoons				
		C_1	C_2	C_3	H_1
C_1		0	-1	1	0
C_2		-1	0	0	1
C_3		1	0	0	-1
H_1		0	1	-1	0

Figure 63: Paraffin Isomerization and Reaction Matrix

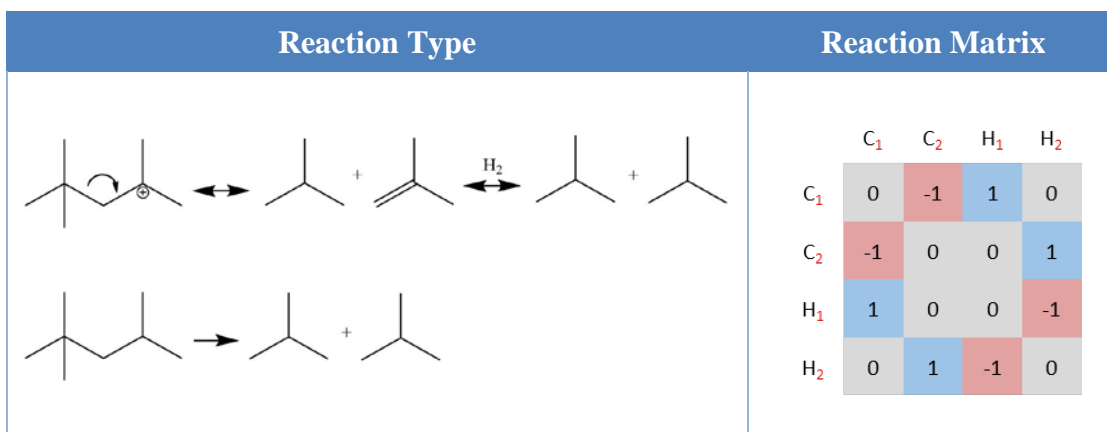


Figure 64: Type A β -scission Pathway and Reaction Matrix

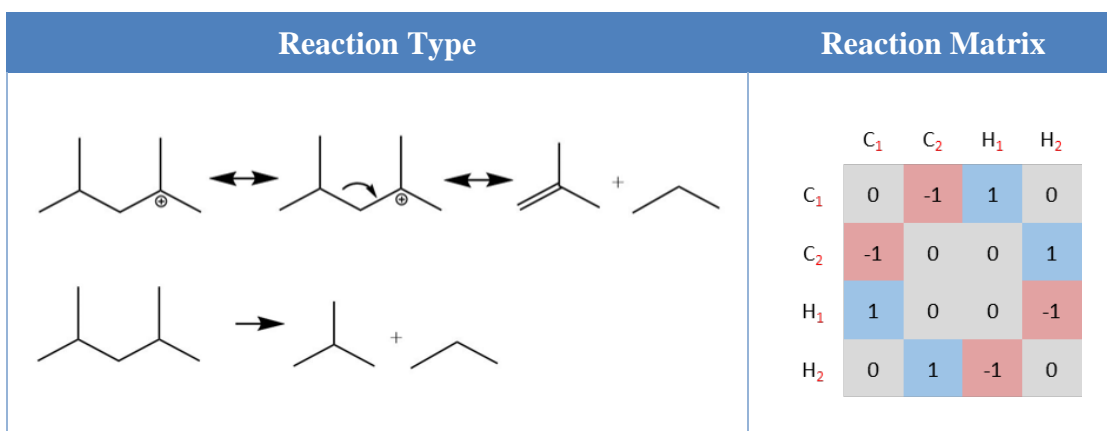


Figure 65: Type B β -scission Pathway and Reaction Matrix

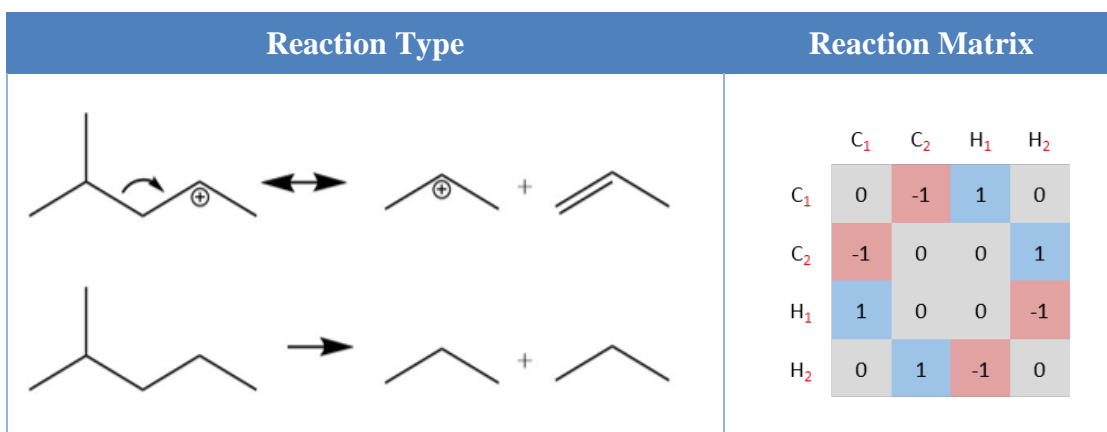


Figure 66: Type C β -scission Pathway and Reaction Matrix

The ringed component reaction network was developed by seeding an alkylated three ring aromatic molecular species in INGen. Saturation, ring opening, ring isomerization, dealkylation, and side chain cracking reactions were allowed to occur on the seeded molecules to produce the ring component reaction network. The saturation reactions represent the conversion of aromatic rings into their saturated counterparts. Figure 67 illustrates saturation reaction types that add 6 hydrogens or 4 hydrogens to an aromatic molecule. The reaction matrix illustrated in Figure 67 is for a 4 hydrogen atom saturation reaction.

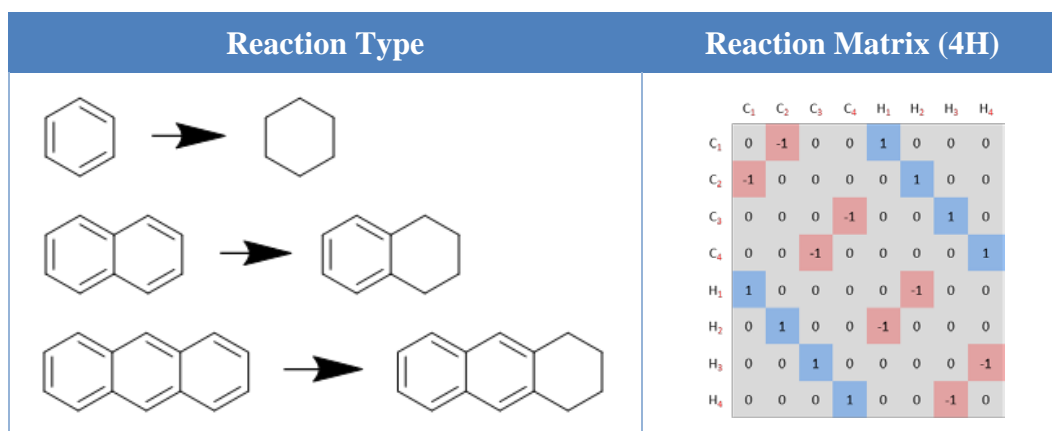


Figure 67: Saturation Pathway and 4H Saturation Reaction Matrix

The ring isomerization is a reversible reaction that converts a 6 membered saturated ring to 5 membered methylated saturated rings, as shown in Figure 68. Methylated saturated rings on multi-ring component are allowed to ring open as shown in Figure 69. Ring opening of single ring components was prohibited in the reaction network. The work by Korre et al. showed that the ring opening rate on methylated indane compounds is greater than that of its aliphatic equivalent (Korre, Quann, & Klein, Hydrocracking of Polynuclear Aromatic Hydrocarbons.

Development of Rate Laws through Inhibition Studies, 1997). In this work all aliphatic ring equivalents have their own reaction families.

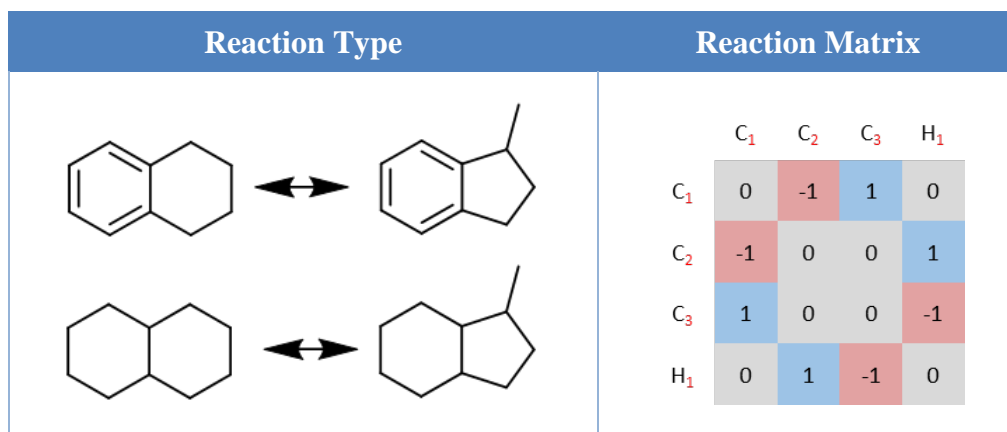


Figure 68: Ring Isomerization Pathway and Reaction Matrix

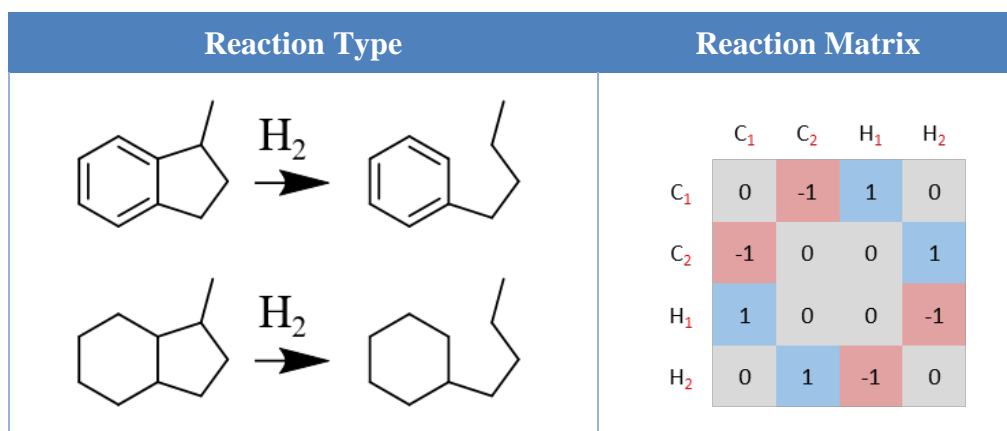


Figure 69: Ring Opening Pathway and Reaction Matrix

The dealkylation reactions occur on mono- and multi-aromatic compounds as well as their naphthenic counterparts. The dealkylation reaction cleaves the alkyl group on the molecule. The dealkylation pathways produce normal paraffins and ring components as well as isoparaffins and ring components. The dealkylation reaction

that produces isoparaffins includes an implicit isomerization and can be thought of as Type B β -scission. Figure 70 and Figure 71 illustrate the dealkylation reactions.

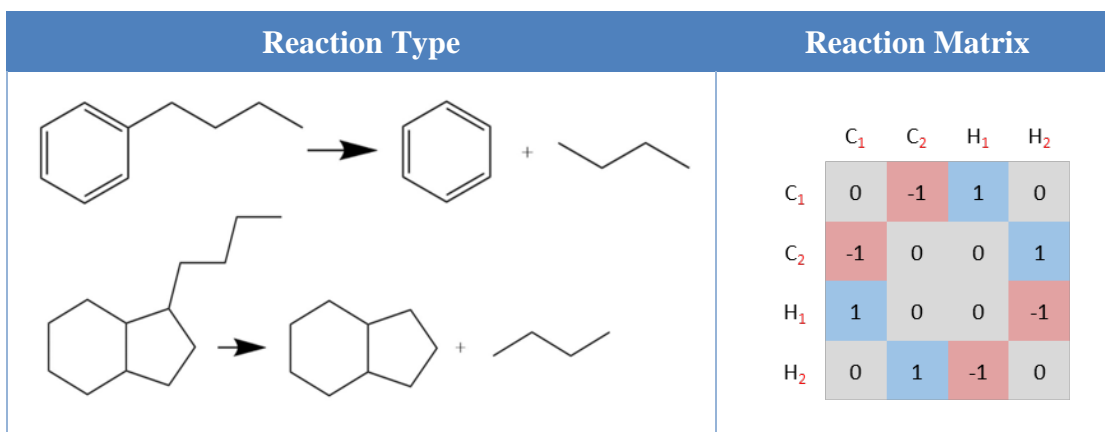


Figure 70: Dealkylation Pathway Normal Paraffin Product

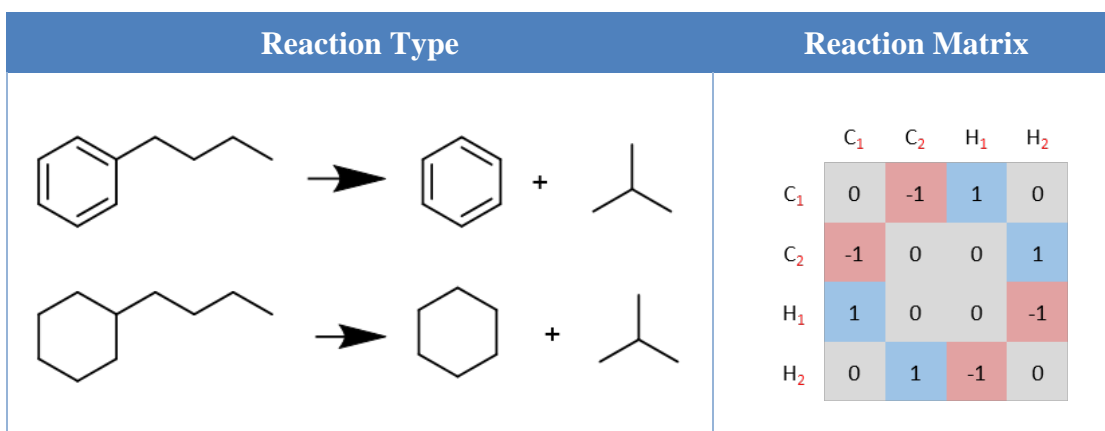


Figure 71: Dealkylation Pathway Isoparaffin Product

The side chain cracking reactions are Type B and Type C β -scission pathways occurring on a side chain. The side chain cracking pathway includes an implicit isomerization of the of the side chain to achieve Type B and Type C β -scission. In order to lower model complexity, side chains were not allowed to have branching.

Once the two reaction networks were built independently, they were merged into one complete diesel hydrocracking reaction network.

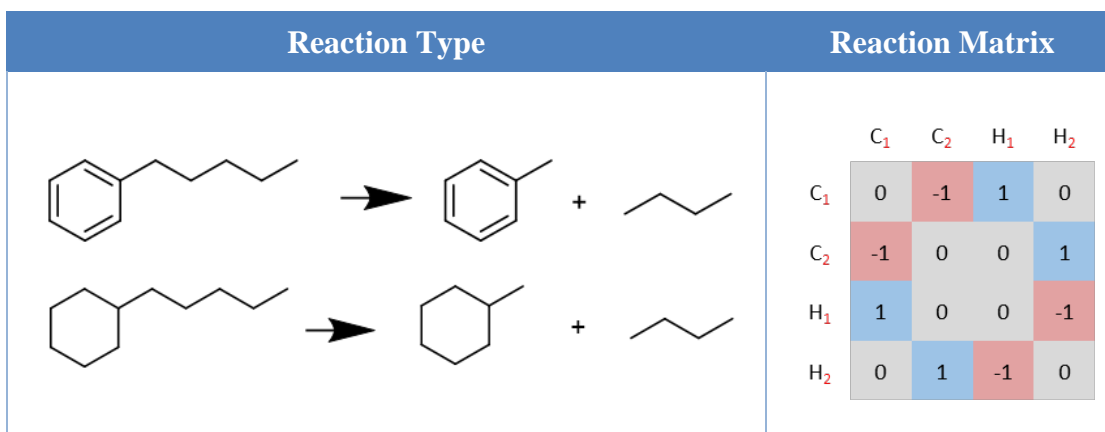


Figure 72: Side Chain Cracking Type C Pathway and Reaction Matrix

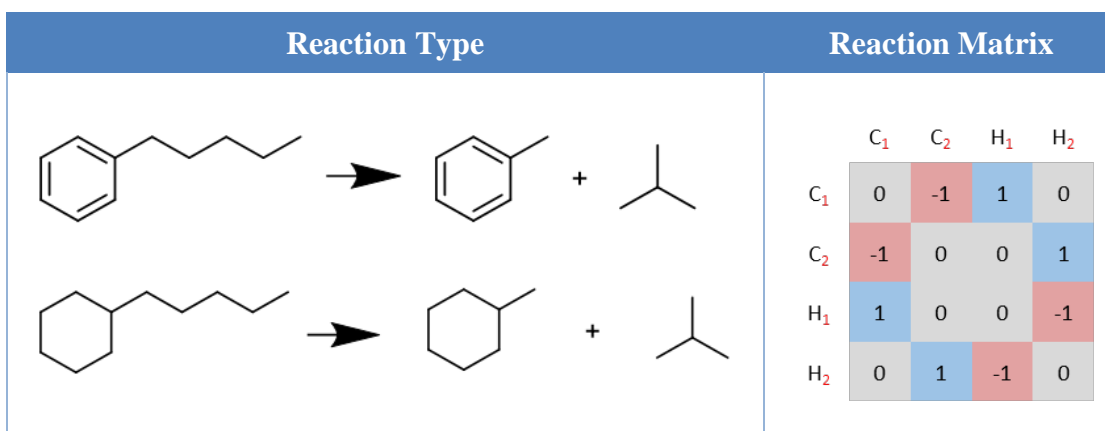


Figure 73: Side Chain Cracking Type B Pathway and Reaction Matrix

6.3.2 Molecule Property Database

Once the reaction network was constructed, properties for each molecule were retrieved from an in-house molecule property database. The database stores properties calculated by Benson and Gani group contribution as well as literature values provided by the users (Benson & Cohen, 1993) (Gani, 1994). The database uses an in-house

tool to calculate properties for each molecule not available in the database known as PropGen (Wei W. , et al., 2008). PropGen is fed molecular structure information and performs the necessary group contribution calculations to determine molecular properties for molecules. The information from these properties can be used to relate molecular and bulk information through bulk property correlations.

6.3.3 Initial Condition Generation

Diesel is a heavily quantified feed due to its heavy utilization in industry. This characterization has illustrated that diesel feed can be split into discrete compound types known as compound classes. The compound classes that make up diesel are well known: paraffins, isoparaffins, naphthenics and aromatics. These compound classes can be described by a probability density functions (PDFs) or a histogram (Daniel M. Trauth, 1994) (Pyl, et al., 2011) (Wei W. , Bennett, Tanaka, Hou, & Klein, 2008). A PDF can be utilized over a histogram when the behavior of the molecular distribution is unknown. In the case of diesel, many molecular distributions can be identified from gas chromatography experiments. The composition model created here for a diesel feed only needed PDF parameters to describe large aromatic species which are not easily quantified via experiments. This composition model was developed as an initial condition generator for the large diesel model, dummy experimental data was utilized to create initial conditions as parallel tuning approaches is the goal of this work.

6.3.4 Model Equations and Kinetics

The kinetic model involves a set of ordinary differential equations (ODEs) that describe mass balances of components in the system. Each mass balance accounts for

the reactions that generate and consume species. The mass balance ODE used in this model was for a plug-flow reactor shown in equation (6.3.4-72) below.

$$\frac{dF_i}{dz} = A_c r_i = A_c \sum_j^{rxns} v_{ij} r_j \quad (6.3.4-72)$$

Each reaction within the model includes a rate constant of the Arrhenius form with a pre-exponential and an activation energy. To reduce parameter complexity, the Bell-Evans-Polanyi linear free-energy relationship (LFER) is used. The concept introduces the idea of reaction families which represent homologous series of reactions that react in the same way and only differ in reactivity by substituent effects. In the equation (6.3.4-73) below, the rate constant for each reaction i can be described by reaction family j parameters: pre-exponential, activation and alpha (Bell, 1936) (Evans & Polanyi, 1935). The substituent effects are captured by the enthalpy of reaction for each reaction i . A carbon number dependence was introduced to the enthalpy of reaction term due to the limited variation of enthalpy between reactions in a reaction family. This was observed by Martinis and Froment and is due to the group contribution calculation of enthalpies (Froment J. M., 2006). Group contribution methods utilize atomic groups to calculate properties. A set of molecules in a reaction family stereotypically have the same atomic groups that react in the same way; therefore, there is a low quantitative variance in enthalpies of reaction.

$$\ln k_{i,j} = \ln A_j - \frac{(E_j + \alpha_j \Delta H_{rxn,i})}{RT} = \left(\ln A_j - \frac{E_j}{RT} \right) - \frac{\alpha_j \Delta H_{rxn,i}}{RT} \quad (6.3.4-73)$$

To describe the catalytic chemistry, LHHW formalisms were applied to microkinetic and power law rate laws. A general surface rate controlled LHHW rate law expression is shown in the equation (6.3.4-74) below for a gas phase system:

$$r_i = \frac{k_{sr_{i,j}} C_T \prod_z^R K_{ads,z} \left(\prod_z^R P_z - \frac{\prod_z^P P_z}{K_{eq_i}} \right)}{\left(1 + \sum_z^N P_z K_{ads,z} \right)^\alpha} \quad (6.3.4-74)$$

The equilibrium constant was formulated by using the thermodynamic definition of K_{eq} calculated from the Gibbs free energy of reaction at temperature T . The Gibbs free energy of reaction is calculated through group contribution methods. Corrections to the equilibrium constant at T were made through tunable parameters a and b in the van't Hoff equation. The van't Hoff equation (6.3.4-75) is used to calculate K_{eq} at a temperature other than the reference temperature (S.C. Korre, 1996).

$$\ln K_{eq}(T) = a * \ln K_{eq}(T_0) + b * \int_{T_0}^T \frac{\Delta H_{rxn}}{RT^2} dT \quad (6.3.4-75)$$

The adsorption of molecules to metal or acid sites is controlled by the coefficient K_{ad} which was calculated in a similar fashion to the work by Korre et al. (S. C. Korre, 1997). The equation (6.3.4-76) below describes the form of the adsorption constants for each species i . The linear equation below is function of temperature and quantitative structure-property correlations (QSPC). The single (a) term is independent of temperature. The temperature dependent terms are multiplied by a QSPC describing aromatic ring number, naphthenic ring number, and saturated carbon number.

$$\ln K_{ads,i} = a + \frac{b * N_{Arom Rings,i}}{RT} + \frac{c * N_{Naph Rings,i}}{RT} + \frac{b * N_{Sat Carbons,i}}{RT} \quad (6.3.4-76)$$

6.3.5 Kinetic Modeling Environment and High Performance Computing

The diesel hydrocracking model simulated in this work had 400 species and 5000 reactions. The model has 78 tunable parameters associated with LFER concepts and 8 parameters for the control of molecular adsorption to the metal and acid sites. The model was simulated in a next generation kinetic modeling environment named Dynamic Model Builder (DMB). Model simulation was ~5 seconds in the DMB framework on Windows PC with a 3.70 Ghz Intel i7-8700k (“Coffee Lake”) CPU with 8 Gb of DDR4 memory. DMB functions on both a Windows PC and a high performance cluster (HPC) allowing for a variable simulation workflow. Model simulation on the University of Delaware’s Farber HPC was less than 3 seconds utilizing the Intel compiler on a 2.5 GHz Intel E5-2670 v2 (“Ivy Bridge”) CPU. A large amount of simulation time was spent on an HPC for kinetic parameter estimation due to the ample resources available and low simulation time. Through the HPC 20 nodes can be accessed that represent 20 independent CPUs. This allows for parallel kinetic parameter estimation runs. A robust metaheuristic method, adapted simulated annealing, was employed for kinetic parameter estimation (Ingber, 1995). The following equation (6.3.4-77), a chi-square function, is minimized in the objective function via ASA.

$$\chi^2 = \sum_{i=0} \left(\frac{Observed - Expected}{\alpha * Observed} \right)^2 \quad (6.3.4-77)$$

A large number of datasets can translate to high number of simulations in the ASA objective function. Sequential processing of the simulations can cause a large amount of time spent in the objective function for each perturbation parameter set. This may place a strain on kinetic parameter estimation metaheuristic to achieve a solution in a timely manner. To alleviate this issue, the simulations in the objective function have been parallelized on the CPU using Open Multi Processing (OpenMP) and multiple threads. The HPC cluster CPU has 20 threads available allowing up to 20 datasets to be tuned in the objective function at once.

6.4 Results and Discussion

The kinetic parameters of the diesel model presented here were not tuned on experimental data. Instead dummy values were chosen to produce a model that could be utilized to test parallel tuning methodologies. To illustrate the capabilities of the model, simulations of 2D gas chromatography (GC) experiments were produced. The dummy model predictions to those experiments are illustrated by Figure 74, Figure 75, Figure 76, and Figure 77. The model can predict isoparaffins carbon number trends as illustrated in Figure 75. The mono-naphthenic distribution is also simulated over a large carbon range and shows dependency on Type B and Type C side chain cracking.

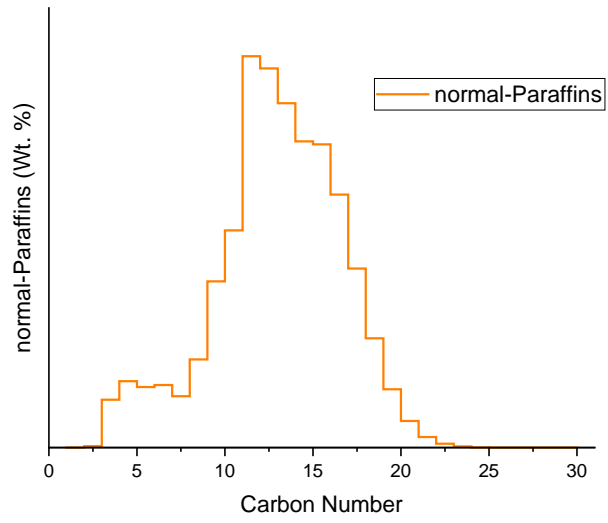


Figure 74: normal-Paraffin Model Prediction based on Carbon Number Distribution



Figure 75: Isoparaffin Model Prediction based on Carbon Number Distribution

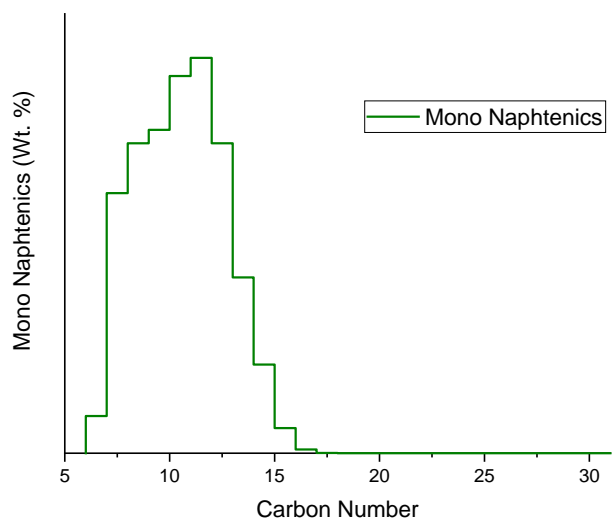


Figure 76: Mono-Naphtenic Model Prediction based on Carbon Number Distribution

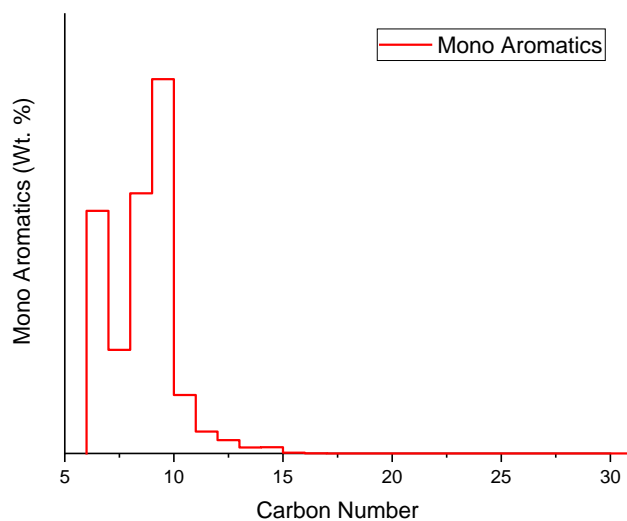


Figure 77: Mono-Aromatic Model Prediction based on Carbon Number Distribution

As model sizes increase by either species numbers and or reactions so does the simulation time. This increase in time severely cripples kinetic parameter estimation. Therefore, this work focused on implementing parallel tuning approaches in the

objective function mentioned in Section 6.3.5. Figure 78 illustrates the advantage of parallelization in the objective function. The x -axis of Figure 78 illustrates the number of datasets run in the objective function. Notice a linear increase for sequential computing shown by the black line. For parallel computing, which is the forest green line, there is a relatively small increase due to thread and memory creation. This overhead increase is relatively unimportant when compared to the advantage of parallel vs. serial computing. The forest green line illustrated by Figure 78 shows the results for utilizing the same number of processors as datasets. In the parallel computing case simulation time remained constant as datasets numbers increased and showed a marked advantage when compared to the serial computing strategy.

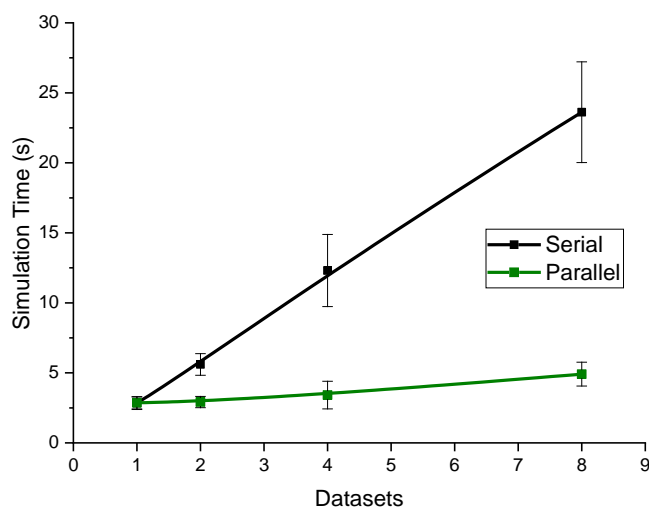


Figure 78: Simulation Time in the Objective Function as Function of Number of Datasets

6.5 Summary and Conclusions

- A pathways based molecular-level model for diesel hydrocracking was developed.
- Model simulation can predict the following experiments:

- PIONA experiment
- Yields experiment
- 2D Gas Chromatography experiments
- The model simulation time was optimized through software profiling tools and it was found that 72% of simulation time was spent in the ODE solver routine.
- Simulation time within the objective function has been decreased by parallelization of dataset simulations on the CPU through OpenMP

Chapter 7

APPLICATION DEVELOPMENT

7.1 Creating One Framework

The current form of the Kinetic Modeler's Toolkit (KMT) is four separate software programs. Communication between the four software tools is necessary on a daily basis for a user is but is not streamlined. Development of each of the software tools in the Klein Research Group occurred separately with many graduate student contributions. Data transfer between applications in KMT has been notoriously achieved through various files and can become confusing and clunky. A new application was created that linked the Dynamic Model Builder (DMB), Property Database (PD) and Interactive Network Generator (INGen) on a software level. Streamline development cycles are necessary to achieve molecular-level modeling in a timely manner. This chapter focuses on describing the modernized communication between the Dynamic Model Builder graphical user interface (GUI) and KMT software tools.

DMB, PD, and INGen can communicate with each other via a dynamic linking library (.dll). The .dll stores information from each software tool in computer memory and makes them accessible to all software tools. In the typical DMB GUI workflow, the GUI calls the library to read all files from an INGen model and stores them in memory. Once the INGen model is available memory, the information from it can be accessed by the DMB GUI. Through the .dll the DMB GUI can also access the Property Database. Unique molecule identifiers, String Code, created in an INGen

model can be used to search the Property database. Properties for each molecule in an INGen model can be extracted from memory and utilized to develop a DMB model. This accessibility is crucial to rid the system of extraneous file IO, instead the system communicates pragmatically. Communication of this type between all the tools in the KMT framework simplifies and modernizes development.

7.2 Dynamic Model Builder Editor GUI

The Dynamic Model Builder is accompanied by a graphical user interface (GUI) named the Dynamic Model Builder Editor. The DMB GUI was developed in the C# programming language and acts as front end to the DMB framework. Its main functionality is to create the necessary file system for the DMB framework. The GUI makes model building a straightforward process. It offers the tools necessary to build a model and displays quantitative information on rate constants, enthalpy of reaction, Gibbs free energy of reaction, equilibrium constants, and adsorptions constants. The new GUI also does not face the same issues as the past interface for KME which had cell number limitations, which limited large model development.

The DMB GUI connects the DMB application to other software tools in the Kinetic Modeler's Toolkit. The Figure 79 and Figure 80 illustrate the DMB Editor GUI during the model building process. Figure 79 depicts the GUI displaying important reaction information. Figure 80 displays more information important to a molecular-level modeler like reactor type, chemistry, initial conditions, Arrhenius parameters, enthalpy of reaction, Gibbs free energy of reaction and equilibrium constant. In past framework none of these quantities were available to the user. This abstracted the model parameters from the developer and made it difficult to diagnose issues in the model. The model developer had to have programming knowledge to

access the inner workings of the software tool and find out these values. Now the GUI displays all the model values and the user can change reactor conditions and see variable effects. An important one is the temperature dependence of the equilibrium constant. The equilibrium constant controls whether the forward rate or backwards rate dominates. The user can change the temperature in the GUI and understand how the equilibrium constant changes as a function of temperature. An added analysis needs to be done via extent of reaction to determine the effect of equilibrium but the DMB framework allows users to have the information for that type of analysis. Another important parameter to understand quantitatively is the enthalpy of reaction which contributes to the rate constant through the LFER concepts. The GUI allows the user to vary the reaction family parameters alpha and see direct changes to rate constants in reaction families. This type of control and knowledge allows the user to determine the usage of model parameters.

Model development has been streamlined from the previous frameworks GUI which was Excel based. Excel has limitations based on row length within an Excel sheet. As model sizes began to get larger and larger the Excel would stop functioning and crash. This led to model development barriers that were just solely related to model building. These issues are additional to the ones created by the use of Perl and VBA. The DMB C# GUI does not have row limitations. The C# programming language also performs operations faster than VBA since it is full-fledged compiled programming language. Development was also streamlined by tracking changes to the model by the user by the Model Check area. As the user progresses in model development the red boxes that say reaction network, reactor type, chemistry, initial conditions, properties, Arrhenius parameters and equilibrium constants turn green. If

values that cause model issues like dividing by zeros, the box will turn red. This allows the user to track model development and is illustrated in Figure 80. The user can also see model logistics information such as number of reactions, number of species, and number of reaction families.

Dynamic Model Builder

File Import Phase Chemistry Options About

Model: Reactor Setup

Reaction Network

Sort by Reaction Families

Index	Reactions	Reaction Families
0	species1 -> species3 + species4	BetaEtherCracking-Oxygen
1	species1 -> species5 + species6	BetaEtherCracking-Oxygen
2	species1 -> species7 + species8	BetaEtherCracking-Oxygen
3	species1 -> species9 + species10	BetaEtherCracking-Oxygen
4	species1 -> species3 + species11	BetaReaction1-Oxygen
5	species1 -> species5 + species12	BetaReaction1-Oxygen
6	species1 -> species7 + species13	BetaReaction1-Oxygen
7	species1 -> species14 + species15	BetaReaction2-Oxygen
8	species1 -> species15 + species16	BetaReaction2-Oxygen
9	species1 -> species15 + species17	BetaReaction2-Oxygen
10	species1 + species2 -> species18 + species19	AlphaEtherCracking-Oxygen
11	species1 + species2 -> species20 + species21	AlphaEtherCracking-Oxygen
12	species1 + species2 -> species22 + species23	AlphaEtherCracking-Oxygen
13	species1 + species2 -> species24 + species25	AlphaEtherCracking-Oxygen
14	species1 + species2 -> species26 + species27	AlphaEtherCracking-Oxygen
15	species3 -> species7 + species28	BetaEtherCracking-Oxygen
16	species3 -> species9 + species29	BetaEtherCracking-Oxygen
17	species3 -> species7 + species30	BetaReaction1-Oxygen
18	species3 -> species15 + species31	BetaReaction2-Oxygen
19	species2 + species3 -> species22 + species32	AlphaEtherCracking-Oxygen
20	species2 + species3 -> species24 + species33	AlphaEtherCracking-Oxygen
21	species2 + species3 -> species26 + species34	AlphaEtherCracking-Oxygen
22	species4 <-> species11	EnolKetoTautomerization-Oxyg...
23	species4 -> species6 + species35	BetaEtherCracking-Oxygen
24	species4 -> species12 + species35	BetaReaction1-Oxygen
25	species4 -> species15 + species36	BetaReaction2-Oxygen
26	species2 + species4 -> species21 + species37	AlphaEtherCracking-Oxygen
27	species2 + species4 -> species19 + species38	AlphaEtherCracking-Oxygen
28	species2 + species4 -> species39	StyreneDegrad1-Oxygen
29	species2 + species4 -> species40 + species41	StyreneDegrad2-Oxygen
30	species5 -> species3 + species35	BetaEtherCracking-Oxygen
31	species5 -> species7 + species42	BetaEtherCracking-Oxygen
32	species5 -> species9 + species43	BetaEtherCracking-Oxygen
33	species5 -> species3 + species44	BetaReaction1-Oxygen
34	species5 -> species7 + species45	BetaReaction1-Oxygen
35	species5 -> species15 + species46	BetaReaction2-Oxygen

Figure 79: DMB GUI Reactions

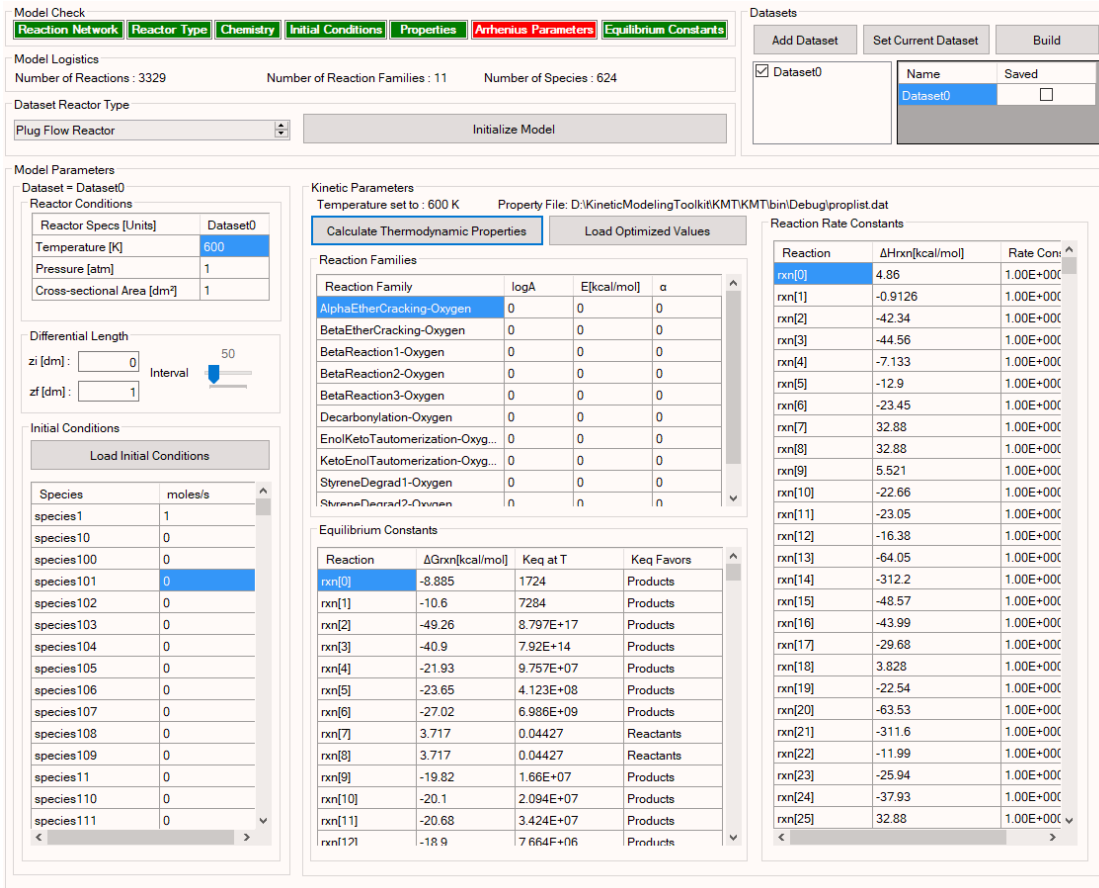


Figure 80: DMB GUI Model Information

7.3 Summary

Streamline model development is extremely important for novice users. The development of a robust GUI allows future users accessibility to develop strong models. Problems areas can be determined quantitatively by displaying all the model parameters in the GUI. The development of GUI is crucial as the DMB modeling environments takes its place as the next generation modeling environment.

Chapter 8

KINETIC PARAMETER ESTIMATION METHODOLOGY

8.1 Motivation

Kinetic parameter estimation is an integral step in the model development process. It can be time consuming and may cripple model development if the necessary performance criteria are not met. Typically, these performance criteria are based on the percent error found in experimentation. This thesis placed a focus on developing a robust framework that could speed up kinetic parameter through advances in optimization algorithms and visualizations. Utilizing the information produced from these advances allowed for a methodology to be developed.

8.2 Kinetic Parameter Estimation Methodologies

Methodologies to perform kinetic parameter estimation are generally not presented due to the traditional brute force approaches utilized. Parameter estimation can be considered an art or a stroke of luck due to its stochastic nature. The following section will describe a methodology to streamline and maneuver through this stochastic environment. Initial conditions, lower/upper bounds, and reaction rate fluxes are several variables that are utilized in the manual and computer aided methodology presented here.

8.2.1 Step 1: Determine Initial Conditions and Bounds

During kinetic parameter estimation, heuristics are utilized to find an optimum set of kinetic parameters that match a criterion. The initial guesses of kinetic parameters that are given to an optimization algorithm are critical in finding the optimum set. If the initial condition is in the vicinity of the final answer it will be easier to find. If the initial guess is too far from the solution, then local optimum may be found that does not match the criteria set and may cause the optimizer to get confused and stuck. Therefore, having good initial guesses for the optimization problem is critical to find the global optimum. Defining these initial guesses is difficult, in this work a strategy is defined to help find the initial guesses. Especially when the system complexity is high it may be difficult to determine an initial guess and the lower/upper bounds. Understanding the relative magnitudes of kinetics parameters based on chemical intuition is very useful. It can help to get in the ballpark but mathematical constraints may convolute the expected values; this is where an optimizer can help. The first part of the method recommended here, is to use a global optimization algorithm like ASA to find the initial kinetic parameter space. Begin by entering information from literature as initial guesses and let a global optimizer perturb the parameters to find a kinetic parameter space that fits the criteria. In the work performed here, the initial search for initial conditions placed all the variables at the same number and placed upper and lower bounds 30% from the guess. This strategy becomes an iterative approach decreasing the bounds after optimization have completed or stalled.

8.2.2 Step 2: Determine Issues through Visualization

The ideal situation is that a global optimization algorithm like ASA can determine the initial conditions and perturb around them to find the correct solution space. This is not always the case, since the metaheuristic may get lost due ill-defined initial conditions or inaccurate chemistry in the model. Therefore, it is recommended to utilize visualization of the rate law fluxes to determine the flow of mass in the model. The 3D surface plots approach presented in the Section 3.10 can illustrate where mass is going in the model while illustrating the effect of equilibrium in a set of reactions that generate and produce a molecule of interest.

8.2.3 Summary

A 2-step method for kinetic parameter estimation has been presented here where initial conditions were determined stochastically and rate law fluxes are visualized. This methodology helps direct kinetic parameter estimation so that it could be achieved in a quicker way. This 2-step method can be further enhanced by utilizing a local optimizer to search local solution subspaces requiring less computational time. This methodology suggests that kinetic parameter estimation is an iterative approach in which model design is altered based on the criterion set by the user. Reactions or species can be added based on the information gained through visualizations to help achieve kinetic parameter estimation.

Chapter 9

DISSERTATION SUMMARY AND CONCLUSIONS

9.1 Summary and Conclusions

In summary, this dissertation presented two parallel research objectives: the development of model building tools and the development of reactor models. Molecular-level reactor models were developed for lignin pyrolysis, cellulose pyrolysis, and diesel hydrocracking. The development of each reactor model molded the parallel development of a new kinetic modeling environment and model building tools. A highlighting achievement was the development of a kinetic modeling environment that does not contain computational barriers when scaling to large molecular systems. The new software tools were utilized to model and analyze the abovementioned reactor models. All the chemical processes modeled in this work share two commonalities: valuable molecular products and the utilization of the next generation kinetic modeling environment.

Lignin pyrolysis was the first reactor model to be created in this dissertation. Reactivity limitations in the development of reaction networks from large molecular species required reaction network merging approaches to achieve solution. After tractable reaction networks were generated from large molecular species, the previous kinetic modeling environment presented issues during compilation. These ranged from compilation failures because of large files to excessive time requirements for parameter estimation. A new approach was developed to model large systems compelling the creation of a new kinetic modeling environment, the Dynamic Model

Builder (DMB). The new approach created models in memory and reconstructed the rate laws and mole balances during run time. In order to increase the simulation speed performance, the DMB application was utilized in the UD GPU Hackathon 2016. This experience allowed for a deeper understanding of the compilation environments, software profiling tools and CPU-GPU applications. Once large models solution times were decreased through a GPU enabled ODE solver the simulation of larger lignin models was no longer impeding. Reaction network analysis demonstrated that the reaction network adequately described a large amount of the chemical kinetics occurring during lignin pyrolysis. It was observed that the vinyl degradation reaction family leads to the formation of high value products like phenol, xylenes, guaiacol derivatives. The primary goal of this work was to develop reaction networks from large molecular species and construct models for them.

It was concluded that the lignin structure is variable and its properties are affected by the monolignol monomers that compose it. The structure variability also affects the pyrolysis product composition. Therefore, one structure cannot adequately define all lignin structures during pyrolysis. Further experimental analysis is required to fully characterize the different possible lignin structures that are needed to accurately predict the products of pyrolysis. This dissertation presented the methods to develop a pathways based molecular-level model from a large structure that retains molecular integrity. The model results agree well with the experimental data utilized in this study based on a generated lignin structure. Following the simulation of lignin pyrolysis, another biomass constituent, cellulose, was simulated.

The simulation of cellulose thermogravimetric analysis (TGA) was the first introduction of multi-phase approaches into the DMB framework. It required the

integration of flash calculations to formulate an overall vapor fraction. Utilizing reaction chemistry and kinetic parameters from the work by Zhou et al. and Agarwal et al., a pathways-based cellulose model was developed (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014) (Agarwal, Dauenhauer, Huber, & Auerbach, 2012). Contrary to the work by Lin et al., cellulose TGA was simulated using activation energies and phase equilibria without the introduction of a temperature lag variable and a coinciding ordinary differential equation (Lin, Cho, Tompsett, Westmoreland, & Huber, 2009). The activation energy temperature dependence managed the effects of heating rates alone. It was also observed that the TGA curves are strongly affected by the initial cellulose degradation and the char formation mechanisms. This is due to the high activation energies of cellulose and cellobiosan which create a barrier to smaller intermediary products. The char activation energy was found to be much lower when compared with the work of Zhou et al. (Zhou, Nolte, Mayes, Shanks, & Broadbelt, 2014). After modeling two biomass constituents, the hydrocracking of a petroleum middle distillate was simulated.

Diesel hydrocracking was the final reactor model developed in this dissertation. The diesel hydrocracking project utilized both the reaction network merging approaches and the next generation kinetic modeling environment. The project contributed heavily to the development of the kinetic parameter estimation portion of the DMB framework. It involved a large number of datasets that varied in feed, residence time, temperature and conversion. In order to capture all these effects, kinetic parameter estimation became a focal point. All the dataset simulations in the objective function were parallelized to decrease the objective function calculation time. It was found that the parallelization of the dataset simulations was more

important than the optimization of single simulations. Parallelization made kinetic parameter estimation of large systems with large data tractable and achievable. The kinetic parameter estimation workflow was accomplished on a high performance cluster (HPC). A HPC workflow allowed for fast simulations due to advanced compilers. The HPC workflow allowed for multiple separate kinetic parameter estimation runs to be performed utilizing up to 20 CPU threads at a time with varying initial conditions and bounds. The usage of an HPC accelerated the kinetic parameter estimation phase and allowed for a streamlined model development. The calculated kinetic parameters represented trends of variable feed, residence time, and molecular information. All the reactor models created in this dissertation contributed in the development of a robust kinetic modeling environment.

The parallel development of models and software tools allowed the reactor models to inform the development of tools. This dissertation illustrated the strength of this type of development and the need for advanced technologies in the simulation of mega models. The main contribution of this dissertation is the next generation kinetic modeling environment (DMB) which outperforms the previous framework. The new framework presents new parallel strategies to accelerate model simulation and kinetic parameter estimation. With these advances, deterministic molecular-level kinetic modeling of large and small systems can be streamlined to ease development. No longer are computational barriers a problem, the obstacles are now mathematical.

9.2 Recommendations for Future Work

The dissertation work presented here focused on developing models for large reaction networks. The reactor model focus was on three pathways-based molecular-level reactor models for lignin pyrolysis, cellulose pyrolysis and diesel hydrocracking.

The problems large systems pose make molecular-level kinetic modeling a challenging task. The computational research presented bridged the difficulty. Further work still remains to streamline large molecular processes model development. The following section will outline the future updates on the reactor models as well as upgrades to the kinetic modeling framework.

9.2.1 Continuation of the Lignin Pyrolysis Work

The first molecular-level model developed in this dissertation was for lignin pyrolysis. The scope of work on lignin pyrolysis focused primarily on the development of reaction networks and models from large molecules. It was observed that as the lignin structure is altered, the product distribution varies. It was also observed that the Freudenberg structure does not adequately represent all product distributions. The feed structure must be informed by experimental data, reaction chemistry, and reactor output results. The work from this dissertation proposes an approach to determine the initial lignin structure based on an iterative method. In the method first lignin monomers are linked to create a lignin structure whose properties are compared to experimental bulk feed properties. Secondly the proposed lignin structure is reacted according to the INGen chemistry defined in this thesis. Finally, the INGen reaction network is then utilized to produce a model for which the output will be compared to experimental work. If this process were streamlined, an objective function can be placed at every comparison portion mentioned to determine the lignin structure. This approach can be thought as tuning the feed structure based on the experimental output. Through the use of the tools outlined in this dissertation this approach can be created. Figure 81 illustrates this approach in detail.

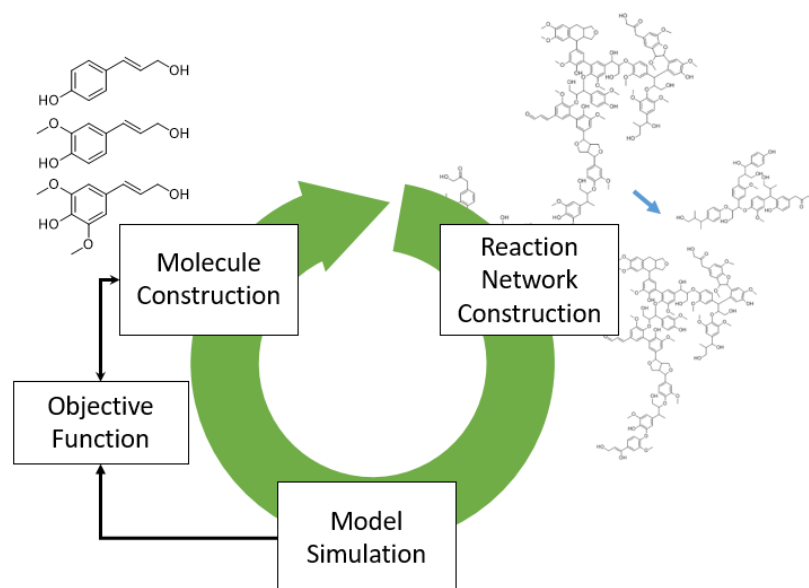


Figure 81: Lignin Feed Tuning Approach

9.2.2 Addition of a Differential Algebraic Equation Solver

The current state of the DMB framework employs three types of reactors a batch reactor, a non-steady state continuous stirred tank reactor (CSTR), and plugged flow reactor (PFR). All these reactors have one commonality; they all require a solution to an ordinary differential equation. The steady state configuration of a CSTR reactor does not require ODEs but is instead a system of equations. A generic form of the steady-state CSTR is illustrated in equation (9.2.2-78). To handle both configurations of CSTR a new type of solver must be utilized.

$$V = \frac{F_{io} - F_i}{r_i} \quad (9.2.2-78)$$

Instead of utilizing the CVODEs solver, a differential algebraic solver must be used from the SUNDIALS software suite. The DAE solver is called Implicit

Differentiation-Algebraic (IDA) solver. The IDA solver can be employed to solve both numerical integration and algebraic equations.

9.2.3 Parallelization of Kinetic Parameter Estimation

Kinetic parameter estimation is a difficult task for larger systems. This topic has a wide audience affecting industry and academics alike. There is no single methodology for kinetic parameter estimation, and it is considered by many to be almost a form of art informed by intuition. In terms of the language of this dissertation, kinetic parameters estimation is heavily affected by simulation speed, number of simulations, tunable parameters, and the metaheuristic method. The bulk of the work presented in this dissertation focused on speeding up single simulations and parallelizing objective function simulations. In this section, two areas for improvement will be discussed: the numerical solution technique for the Jacobian and the complete parallelization of kinetic parameter estimation.

The simulations discussed to this point were performed using implicit numerical integration methods. The Jacobian matrix in this work is approximated by finite differences which produces a dense matrix. The CVODEs library allows the user to define the Jacobian matrix. In the case of a microkinetic system, the Jacobian matrix can be defined as a sparse matrix. A system of equations involving a sparse Jacobian matrix can be solved by generalized minimal residual methods (GMRES). GMRES are iterative methods that can decrease numerical solution time. The MAGMA GPU library also offers a GMRES solver. In the future, a sparse matrix can be defined in the DMB framework and a GMRES based solver applied to it. This will increase simulation performance by decreasing solution time. The Jacobian for a

microkinetic system can be sparse, but for a catalytic system, that may not be the case since the adsorption group denominator includes every species in the system.

Therefore, the derivative of every rate law will produce a dense Jacobian matrix. The solution to this may be to precondition the matrix with approximated adsorption isotherm values.

The second area for improvement in the DMB framework was partially pursued in this dissertation. It deals with the parallelization of the metaheuristic method. Currently, the DMB framework utilizes a serial adaptive simulated annealing (ASA) metaheuristic for kinetic parameter estimation. The workflow for kinetic parameter estimation currently exists on a high performance cluster (HPC). The HPC utilization allows for the parallelization of the ODE solver and the objective function calculations. CPU threads parallelize simulations in the objective function while GPU processors parallelize the LU decomposition routine in the ODE solver. However, the HPC workflow allows for one more type of parallelization to be implemented. This parallelization is of the metaheuristic method and it can be achieved via node parallelization. A typical HPC is composed of many nodes where each node can be thought of as a computer. The Farber HPC contains 190 compute nodes. Each node is a core of processors with 20 CPU threads. A genetic algorithm (GA) is metaheuristic method that lends itself well to parallelization. A genetic algorithm package, written by Matthew Wall at the Massachusetts Institute of Technology, has been examined that uses parallelization of multiple nodes on an HPC via Message Passing Interfaces (MPI). GA methods can search larger kinetic parameter solution spaces than that of ASA (Ingber, 1995). The ultimate dream would be to parallelize every portion of the kinetic parameter estimation. The metaheuristic would function via node

parallelization of dataset simulations using MPI where each node would calculate an objective function. Each objective function would employ multi-threading to parallelize simulations. Each simulation would use ODE solver parallelization via the GPU. Figure 82 illustrates the proposed parallelized kinetic parameter estimation workflow. The green boxes illustrate the areas of parallelization: genetic algorithm, objective function, and ODE solver. The orange boxes illustrate the parallelization hardware: nodes, CPU threads and GPU compute units. Especially as system sizes increase, kinetic parameters estimation needs to be further accelerated through parallelization to make it achievable.

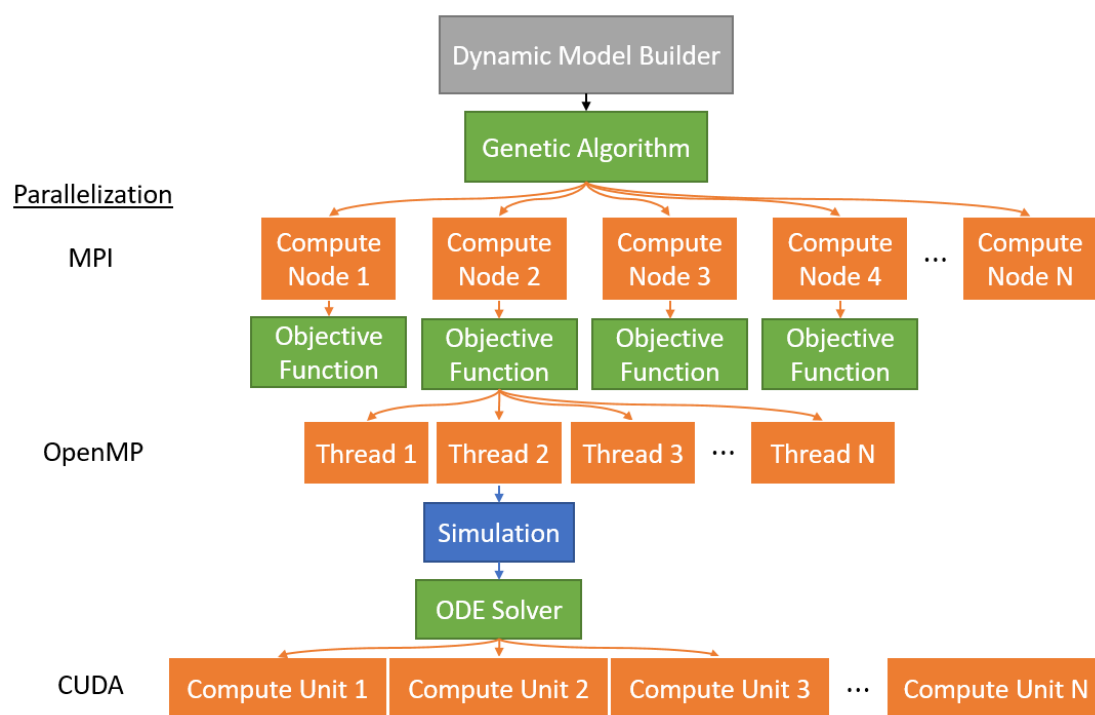


Figure 82: Parallel Kinetic Parameter Estimation

9.2.4 Large Model Explicit Ordinary Differential Equations

As mentioned, the DMB framework creates rate laws and mole balances during run time from a series of nested loops. This bypasses the need for large ODE files which are not compiler friendly. As system sizes increase, there might be a need to use explicit ODEs again depending on performance needs. It has become clear that the problem during compilation is object code creation. The compiler creates object codes which are statements and instruction in machine language. Typically, each file represents one object module. The compiler links the object modules together into one executable during compilation. The problem with the previous framework is that the object size limit was reached with large models. All the code for a model was written to one object. A new approach would be to explicitly write the rate laws and mole balances to different functions in separate files. Inherently, this separation of code discretizes the code so that it can be compiled. The separate files would produce separate object modules and can be created separately by the compiler. Each object would have a function that only needs to be called by the original ODE file. This decreases the amount of code in one file and makes it so that the objects can be linked together into one executable. Figure 83 illustrates the separation of code into multiple files to achieve compilation.

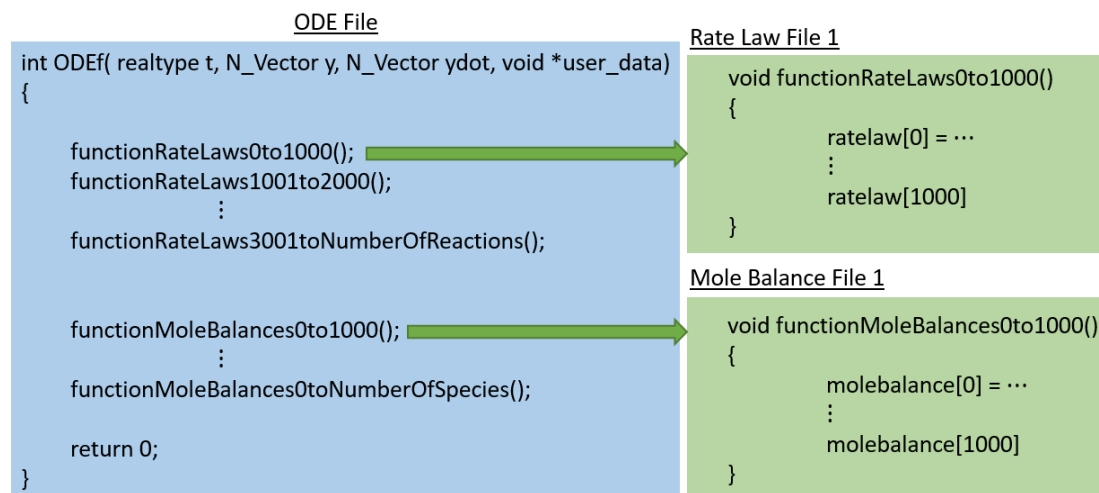


Figure 83: Explicit ODE File Separation

9.3 Closing Remarks

The molecular-level modeling of the three reactor systems detailed in this dissertation was achieved through various software methods and approaches. In the case of biomass pyrolysis there are many process parameters that need to be optimized to make the process practical. As of 2016 biomass contributes to only 5% of the fuels market, of that 5%, 48% comes from ethanol and 41% from wood derived sources (Biomass Explained, 2018). To increase the usage of biomass pyrolysis processes further optimization needs to occur. Molecular-level models are software tools that can be utilized to find these optimal trajectories. They can also be used to predict the integration of these reactor units in existing platforms to determine their benefits. Molecular-level modeling is becoming more and more necessary as feed complexity increases and molecular requirements become the norm.

As the natural deterministic molecular-level modeling progression continues to attempt larger and larger systems the coupling of technological tools and algorithms is becoming more important. This dissertation laid the ground work for kinetic modeling

of large systems at molecular resolutions through the utilization multiple advanced technologies. Through the work presented here it has become clear that unconventional hardware is necessary in the modeling of real systems. This has not remained unnoticed as many molecular-level modelers adopt new workflows on high performance clusters (HPC) (Cochegrue, et al., 2011) (Becker, Celse, Guillaume, Dulot, & Costa, 2015) (Becker, Serrand, Celse, Guillaume, & Dulot, 2017). The contributions in this dissertation enable the development of large systems and evolve KMT software suite into a new burgeoning era of advanced technological utilization.

REFERENCES

- Agarwal, V., Dauenhauer, P. J., Huber, G. W., & Auerbach, S. M. (2012). Ab Initio Dynamics of Cellulose Pyrolysis: Nascent Decomposition Pathways at 327 and 600 °C. *Journal of the American Chemical Society*, *134*, 14958–14972.
- Antal, M. J., & Varhegyi, G. (1995). Cellulose Pyrolysis Kinetics: The Current State of Knowledge. *Ind. Eng. Chem. Res.*, *34*, 703.
- Aspen Assay Management. (2018, May 5). Retrieved from Aspen Assay Management: <https://www.aspentech.com/en/products/pages/aspen-assay-management>
- Baboulin, S. T. (2010). Towards dense linear algebra for hybrid GPU accelerated manycore systems. *Parallel Matrix Algorithms and Applications*, *36*(5-6), 232-240.
- Becker, P. J., Celse, B., Guillaume, D., Dulot, H., & Costa, V. (2015). Hydrotreatment modeling for a variety of VGO feedstocks: A continuous lumping approach. *Fuel*, 133-143.
- Becker, P. J., Serrand, N., Celse, B., Guillaume, D., & Dulot, H. (2017). A single events microkinetic model for hydrocracking of vacuum gas oil. *Computers and Chemical Engineering*, *98*, 70-79.
- Bell, R. P. (1936). The Theory of Reactions Involving Proton Transfers. *Proc. R. Soc. London, Ser. A*, *414-429*, 154.
- Bellan, J., & Miller, R. S. (2010). A Generalized Biomass Pyrolysis Model Based on Superimposed Cellulose, Hemicellulose and Lignin Kinetics. *Combustion Science and Technology*, 97-137.
- Bennett, C. A. (2009). User-Controlled Kinetic Network Generation with INGen. *Ph.D. Thesis, Rutgers, The State University of New Jersey, New Brunswick, NJ*.
- Benson, S. W., & Cohen, N. (1993). Estimation of Heats of Formation of Organic Compounds by Additivity Methods. *Chem. Rev*, *93*, 2419-2438.
- Billa, T., Horton, S. R., Sahasrabudhe, M., Saravanan, C., Hou, Z., Agarwal, P., . . . Klein, M. T. (2017). Enhancing the value of detailed kinetic models through the development of interrogative software applications. *Computers & Chemical Engineering*, *106*, 512-528.
- Biomass Explained. (2018, May 3). Retrieved from US Energy Information Administration: https://www.eia.gov/energyexplained/index.cfm?page=biomass_home
- Bradbury, A. G., Sakai, Y., & Shafizadeh, F. (1979). A kinetic model for pyrolysis of cellulose. *J. Appl. Polym. Sci.*, *23*, 3271-80.

- Broadbelt, L. J., Stark, S. M., & Klein, M. T. (1994). Computer Generated Pyrolysis Modeling: On-the-Fly Generation of Species, Reactions, and Rates. *Ind. Eng. Chem. Res.*, 790-799.
- Broido, A., & Nelson, M. A. (1975). Char Yields on Pyrolysis of Cellulose. *Combust. Flame*, 24, 263-268.
- California Diesel Fuel Program. (2018, April 24). Retrieved from California Air Resources Board: <https://www.arb.ca.gov/fuels/diesel/diesel.htm>
- Cohegrue, H., Gauthier, P., Verstraete, J. J., Surla, K., Guillaume, D., Galtier, P., & Barbier, J. (2011). Reduction of Single Event Kinetic Models by Rigorous Relumping: Application to Catalytic Reforming. *Oil Gas Sci. Technol.*, 367-397.
- Dellon, L. D., Yanez, A. J., Li, W., Mabon, R., & Broadbelt, L. J. (2017). Computational Generation of Lignin Libraries from Diverse Biomass Sources. *Energy Fuels*, 8263–8274.
- Diebold, J. P. (1994). A Unified, Global Model for the Pyrolysis of Cellulose. *Biomass and Bioenergy*, 7, 75-85.
- Diego Iribarren, J. F. (2012). Life cycle assessment of transportation fuels from biomass pyrolysis. *Fuel*, 812-821.
- Diesel Fuel Standards and Rulemakings. (2018, April 24). Retrieved from United States Environmental Protection Agency: <https://www.epa.gov/diesel-fuel-standards/diesel-fuel-standards-and-rulemakings>
- Du, S., Valla, J. A., & Bollas, G. M. (2013). Characteristics and origin of char and coke from fast and slow, catalytic and thermal pyrolysis of biomass and relevant model compounds. *Green Chem.*, 3214-3229.
- Evans, M. G., & Polanyi, M. (1935). Some Applications of the Transition State Method to the Calculation of Reaction Velocities, Especially in Solution. *Trans Faraday Soc.*, 35, 875.
- Faravellia, T., Frassoldatia, A., Migliavacca, G., & Ranzia, E. (2010). Detailed kinetic modeling of the thermal degradation of lignins. *Biomass and Bioenergy*, 290-301.
- Flory, P. J. (1934). Molecular Size Distribution in Linear Condensation Polymers. *Journal of the American Chemical Society*, 58, 1877-1885.
- Fogler, H. S. (2006). *Elements of Chemical Reaction Engineering*. Boston: Pearson Education.
- Freudenberg, K. (1959). Biosynthesis and Constitution of Lignin. *Nature*, 1152–1155.
- Froment, G. F., Bischoff, K. B., & de Wilde, J. (2011). *Chemical Reactor Analysis and Design*. Danvers MA: John Wiley & Sons, Inc.
- Froment, G. F., Depauw, G. A., & Vanrysselberghe, V. (1994). Kinetic Modeling and Reactor Simulation of Oil Fractions. *Ind. Eng. Chem. Res.*, 33, 2975-2988.
- Froment, J. M. (2006). Alkylation on Solids Acids. Part 2. Single-Event Kinetic Modeling. *Ind. Eng. Chem. Res.*, 45, 954-967.
- Gani, L. C. (1994). New Group Contribution Method for Estimating Properties of Pure Compounds. *AIChE Journal*, 40(10), 1697-1710.

- Harrod, W. (2012). *A Journey to Exascale Computing*. Salt Lake City, Utah: SC 12.
- Hernández Pérez, F. E., Mukhadiyev, N., Xu, X., Sow, A., Lee, B. J., Sankaran, R., & Im, H. G. (2018). Direct numerical simulations of reacting flows with detailed chemistry using many-core/GPU acceleration. *Computers and Fluids*, *3*, 1-7.
- Hindmarsh, A. C. (2005). SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, *31*(3), 363-396.
- Horton, S. R. (2016). *Modeling Municipal Solid Waste Gasification: Molecular-level Kinetics and Software Tools*. Newark.
- Horton, S. R., Mohr, R. J., Zhang, Y., Petrocelli, F. P., & Klein, M. T. (2016). Molecular-Level Kinetic Modeling of Biomass Gasification. *Energy and Fuels*, *30*, 1647-1661.
- Horton, S. R., Woeckener, J., Mohr, R., Zhang, Y., Petrocelli, F., & Klein, M. T. (2016). Molecular-Level Kinetic Modeling of the Gasification of Common Plastics. *Energy Fuels*, 1662–1674.
- Horton, S. R., Zhang, L., Hou, Z., Bennet, C. A., Klein, M. T., & Zhao, S. (2015). Molecular-Level Kinetic Modeling of Resid Pyrolysis. *Ind. Eng. Chem. Res.*, *54*.
- Hou, Z. (2011). *Software Tools for Molecule-based Kinetic Modeling of Complex Systems*. Brunswick, New Jersey: ProQuest, LLC.
- Hou, Z., Bennett, C. A., Klein, M. T., & Virk, P. S. (2010). Approaches and Software Tools for Modeling Lignin Pyrolysis. *Energy Fuels*, 58-67.
- Hughey, C. A., Rodgers, R. P., & Marshall, A. G. (2002). Resolution of 11 000 Compositionally Distinct Components in a Single Electrospray Ionization Fourier Transform Ion Cyclotron Resonance Mass Spectrum of Crude Oil. *Anal. Chem.*, 4145–4149.
- Ingber, L. (1995). Adaptive simulated annealing (ASA): Lessons learned. *Control and Cybernetics*, *25*, 33-54.
- Jackson, P. G. (1968). The Thermal Decomposition of Acetic Acid. *Phys. Org.*, 1153-115.
- Jaffe, P. G. (2006). Detailed Composition-Based Model for Predicting the Cetane Number of Diesel Fuels. *Ind. Eng. Chem. Res.*, 346–351.
- Jaffe, R. J. (1992). Structure-oriented lumping: describing the chemistry of complex hydrocarbon mixtures. *Ind. Eng. Chem. Res.*, 2483–2497.
- Klein, M. T. (2006). *Molecular Modeling in Heavy Hydrocarbon Conversions*. Taylor & Francis.
- Korre, S. C., & Klein, M. T. (1996). Development of temperature-independent quantitative structure/reactivity relationships for metal- and acid-catalyzed reactions. *Catalysis Today*, *31*, 79-91.
- Korre, S. C., Klein, M. T., & Quann, R. J. (1997). Hydrocracking of Polynuclear Aromatic Hydrocarbons. Development of Rate Laws through Inhibition Studies. *Ind. Eng. Chem. Res.*, *36*, 2041-2050.

- Korre, S. C., Quann, R., & Klein, M. T. (1997). Hydrocracking of Polynuclear Aromatic Hydrocarbons. Development of Rate Laws through Inhibition Studies. *Ind. Eng. Chem. Res.*, 36(6), 2041–2050.
- Kumar, G. F. (2007). Mechanistic Kinetic Modeling of the Hydrocracking of Complex Feedstocks, such as Vacuum Gas Oil. *Ind. Eng. Chem. Res.*, 46, 5881–5897.
- Lewis, L. B. (2000). Dirigent Proteins and Dirigent Sites Explain the Mystery of Specificity of Radical Precursor Coupling in Lignan and Lignin Biosynthesis. *Plant Physiology*, 453–461.
- Lin, Y.-C., Cho, J., Tompsett, G. A., Westmoreland, P. R., & Huber, G. W. (2009). Kinetics and Mechanism of Cellulose Pyrolysis. *Journal of Physical Chemistry*, 13, 20097–20107.
- Martens, G. G., Marina, G. B., Martens, J. A., Jacobs, P. A., & Baron, G. V. (2000). A Fundamental Kinetic Model for Hydrocracking of C8 to C12 Alkanes on Pt/US–Y Zeolites. *Journal of Catalysis*, 253–267.
- Matyas Kosa, H. B. (2011). Pyrolysis oils from CO₂ precipitated Kraft Lignin. *Green Chemistry*, 3196–3202.
- Moreno, B. M., Li, N., Lee, J., Huber, G. W., & Klein, M. T. (2013). Modeling aqueous-phase hydrodeoxygenation of sorbitol over Pt/SiO₂–Al₂O₃. *Royal Society of Chemistry*, 23769–23784.
- Norinaga, K., Yang, H., Tanaka, R., Appari, S., Iwanaga, K., Takashima, Y., . . . Hayashi, J.-i. (2014). A mechanistic study on the reaction pathways leading to benzene and naphthalene in cellulose vapor phase cracking. *Biomass and Bioenergy*, 144–154.
- NVidia. (2018). *AI Revolution Moores Law*. Retrieved April 04/20/2018, 2018, from <https://blogs.nvidia.com/wp-content/uploads/2018/05/AI-Revolution-Moores-Law.png>
- Paine III, J. B., Pithawala, Y. B., & Naworal, J. D. (2008). Carbohydrate pyrolysis mechanisms from isotopic labeling: Part 2. The pyrolysis of d-glucose: General disconnective analysis and the formation of C1 and C2 carbonyl compounds by electrocyclic fragmentation mechanisms. *Journal of Analytical and Applied Pyrolysis*, 10–41.
- Phillips, Z. L. (1991). Comprehensive Two-Dimensional Gas Chromatography using an On-Column Thermal Modulator Interface. *Journal of Chromatographic Science*, 227–231.
- Pyl, S. P., Hou, Z., Van Geem, K. M., Reyniers, M.-F., Marin, G. B., & Klein, M. T. (2011). Modeling the Composition of Crude Oil Fractions Using Constrained Homologous Series. *Ind. Eng. Chem. Res.*, 10850–10858.
- Quann, R. J., & Jaffe, S. B. (1996). Building useful models of complex reaction systems in petroleum refining. *Chemical Engineering Science*, 1615–1631.
- Rahbar, S. Y. (2015). Molecular Origin of Strength and Stiffness in Bamboo Fibrils. *Scientific Reports*, 5, 1–13.

- Speeches*. (2018, May 4). Retrieved from Exxon Mobil:
<http://corporate.exxonmobil.com/en/company/news-and-updates/speeches/full-integration-a-recipe-for-success-in-chemical>
- Vermerris, W., Thompson, K. J., & McIntyre, L. M. (2002). The maize Brown midrib1 locus affects cell wall composition and plant development in a dose-dependent manner. *Heredity*, 450–457.
- Virk, P., & Klein, M. T. (1983). Model pathways in lignin thermolysis. 1. Phenethyl phenyl ether. *Ind. & Eng. Chem.*, 35-45.
- Wei, W., Bennett, C. A., Tanaka, R., Hou, G., & Klein, M. T. (2008). Detailed kinetic models for catalytic reforming. *Fuel Processing Technology*, 344-349.
- Wei, W., Bennett, C. A., Tanaka, R., Hou, G., Klein, M. T., & Klein, M. T. (2008). Computer Aided Kinetic Modeling with KMT and KME. *Fuel Process. Technol.*, 89, 350.
- Weitkamp, J. (2012). Catalytic Hydrocracking - Mechanisms and Versatility of the Process. *ChemCatChem*, 4, 292-306.
- Wooten, J. B., Seeman, J. I., & Hajaligol, M. R. (2004). Observation and Characterization of Cellulose Pyrolysis Intermediates by ¹³C CPMAS NMR. A New Mechanistic Model. *Energy Fuels*, 18, 1-15.
- Xiaolei Zhang, W. Y. (2012). Thermal decomposition mechanism of levoglucosan during cellulose pyrolysis. *Journal of Analytical and Applied Pyrolysis*, 110-119.
- Yang, K. a. (1950). Determination of Mechanism of Catlyzed Gaseus Reactions. *Chem. Eng. Prog.*, 146-157.
- Zhang, M., Resende, F. L., & Moutsoglou, A. (2014). Catalytic fast pyrolysis of aspen lignin via Py-GC/MS. *Fuels*, 358-369.
- Zhou, X., Nolte, M. W., Mayes, H. B., Shanks, B. H., & Broadbelt, L. J. (2014). Experimental and Mechanistic Modeling of Fast Pyrolysis of Neat Glucose-Based Carbohydrates. 1. Experiments and Development of a Detailed Mechanistic Model. *Industrial & Engineering Chemical Research*, 53, 13274-13289.

Appendix A

ADJACENCY LIST, DMB OPTIONS, AND LIGNIN MOLECULES

A.1 Adjacency List Example

The Figure below illustrates the isohexane molecule discussed in the Ch2. The adjacency list only contains non-zero information.

```
//2methylhexane CRLE
23 CRLE
0 0 H : {3, -1}; CRLE
1 0 H : {3, -1}; CRLE
2 0 H : {3, -1}; CRLE
3 0 C : {0, -1}, {1, -1}, {2, -1}, {9, -1}; CRLE
4 0 C : {8, -1}, {5, -1}, {6, -1}, {9, -1}; CRLE
5 0 H : {4, -1}; CRLE
6 0 H : {4, -1}; CRLE
7 0 H : {9, -1}; CRLE
8 0 H : {4, -1}; CRLE
9 0 C : {4, -1}, {7, -1}, {3, -1}, {10, -1}; CRLE
10 0 C : {9, -1}, {11, -1}, {12, -1}, {15, -1}; CRLE
11 0 H : {10, -1}; CRLE
12 0 H : {10, -1}; CRLE
13 0 H : {15, -1}; CRLE
14 0 H : {15, -1}; CRLE
17 0 H : {16, -1}; CRLE
16 0 C : {22, -1}, {17, -1}, {18, -1}, {15, -1}; CRLE
18 0 H : {16, -1}; CRLE
19 0 H : {22, -1}; CRLE
15 0 C : {13, -1}, {14, -1}, {16, -1}, {10, -1}; CRLE
20 0 H : {22, -1}; CRLE
21 0 H : {22, -1}; CRLE
22 0 C : {19, -1}, {20, -1}, {16, -1}, {21, -1}; CRLE
```

Figure 84: 2-methylhexane Adjacency List

A.2 DMB Framework

Table 38: List of DMB Defined Mixture Properties

Mixture Property	Fraction
Density(gg/cc)	None
MW(g/mol)	None

PONA_ModifiedP_WT	Mole/Mass/Volume
PONA_ModifiedI_WT	Mole/Mass/Volume
PONA_O_WT	Mole/Mass/Volume
PONA_ModifiedN_WT	Mole/Mass/Volume
PONA_A_WT	Mole/Mass/Volume
C5C6Unspeciated_WT	Mole/Mass/Volume
Modified_Monocycloparaffins_WT	Mole/Mass/Volume
Othercycloparaffins_WT	Mole/Mass/Volume
AromRing1_WT	Mole/Mass/Volume
AromRing2_WT	Mole/Mass/Volume
AromRing3_WT	Mole/Mass/Volume
AromRing4Plus_WT	Mole/Mass/Volume
Sulfur_WT	Mole/Mass/Volume
Hydrogen_WT	Mole/Mass/Volume
Nitrogen_WT	Mole/Mass/Volume
NapthaFrac(C7-350F)	Mole/Mass/Volume
UCOFrac(350F+)	Mole/Mass/Volume
LNfrac(C5-C6)	Mole/Mass/Volume
LEFrac(C1-C4)	Mole/Mass/Volume
VIBP(K)	Mole/Mass/Volume
V5BP(K)	Matches VIBP(K) Basis
V10BP(K)	Matches VIBP(K) Basis
V20BP(K)	Matches VIBP(K) Basis
V30BP(K)	Matches VIBP(K) Basis
V40BP(K)	Matches VIBP(K) Basis
V50BP(K)	Matches VIBP(K) Basis
V60BP(K)	Matches VIBP(K) Basis
V70BP(K)	Matches VIBP(K) Basis
V80BP(K)	Matches VIBP(K) Basis
V90BP(K)	Matches VIBP(K) Basis
V95BP(K)	Matches VIBP(K) Basis
VEBP(K)	Matches VIBP(K) Basis
nP1_WT	Mole/Mass/Volume
nP2_WT	Mole/Mass/Volume
nP3_WT	Mole/Mass/Volume
nP4_WT	Mole/Mass/Volume
nP5_WT	Mole/Mass/Volume
nP6_WT	Mole/Mass/Volume
nP7_WT	Mole/Mass/Volume

nP8_WT	Mole/Mass/Volume
nP9_WT	Mole/Mass/Volume
nP10_WT	Mole/Mass/Volume
nP11_WT	Mole/Mass/Volume
nP12_WT	Mole/Mass/Volume
nP13_WT	Mole/Mass/Volume
nP14_WT	Mole/Mass/Volume
nP15_WT	Mole/Mass/Volume
nP16_WT	Mole/Mass/Volume
nP17_WT	Mole/Mass/Volume
nP18_WT	Mole/Mass/Volume
nP19_WT	Mole/Mass/Volume
nP20_WT	Mole/Mass/Volume
nP21_WT	Mole/Mass/Volume
nP22_WT	Mole/Mass/Volume
nP23_WT	Mole/Mass/Volume
nP24_WT	Mole/Mass/Volume
nP25_WT	Mole/Mass/Volume
nP26_WT	Mole/Mass/Volume
nP27_WT	Mole/Mass/Volume
nP28_WT	Mole/Mass/Volume
nP29_WT	Mole/Mass/Volume
nP30_WT	Mole/Mass/Volume
iP4_WT	Mole/Mass/Volume
iP5_WT	Mole/Mass/Volume
isopar_nC5_nC6_WT	Mole/Mass/Volume
isopar_nC6_nC7_WT	Mole/Mass/Volume
isopar_nC7_nC8_WT	Mole/Mass/Volume
isopar_nC8_nC9_WT	Mole/Mass/Volume
isopar_nC9_nC10_WT	Mole/Mass/Volume
isopar_nC10_nC11_WT	Mole/Mass/Volume
isopar_nC11_nC12_WT	Mole/Mass/Volume
isopar_nC12_nC13_WT	Mole/Mass/Volume
isopar_nC13_nC14_WT	Mole/Mass/Volume
isopar_nC14_nC15_WT	Mole/Mass/Volume
isopar_nC15_nC16_WT	Mole/Mass/Volume
isopar_nC16_nC17_WT	Mole/Mass/Volume
isopar_nC17_nC18_WT	Mole/Mass/Volume
isopar_nC18_nC19_WT	Mole/Mass/Volume

isopar_nC19_nC20_WT	Mole/Mass/Volume
isopar_nC20_nC21_WT	Mole/Mass/Volume
isopar_nC21_nC22_WT	Mole/Mass/Volume
isopar_nC22_nC23_WT	Mole/Mass/Volume
isopar_nC23_nC24_WT	Mole/Mass/Volume
isopar_nC24_nC25_WT	Mole/Mass/Volume
isopar_nC25_nC26_WT	Mole/Mass/Volume
isopar_nC26_nC27_WT	Mole/Mass/Volume
isopar_nC27_nC28_WT	Mole/Mass/Volume
isopar_nC28_nC29_WT	Mole/Mass/Volume
isopar_nC29_nC30_WT	Mole/Mass/Volume
mononaf_nC5_nC6_WT	Mole/Mass/Volume
mononaf_nC6_nC7_WT	Mole/Mass/Volume
mononaf_nC7_nC8_WT	Mole/Mass/Volume
mononaf_nC8_nC9_WT	Mole/Mass/Volume
mononaf_nC9_nC10_WT	Mole/Mass/Volume
mononaf_nC10_nC11_WT	Mole/Mass/Volume
mononaf_nC11_nC12_WT	Mole/Mass/Volume
mononaf_nC12_nC13_WT	Mole/Mass/Volume
mononaf_nC13_nC14_WT	Mole/Mass/Volume
mononaf_nC14_nC15_WT	Mole/Mass/Volume
mononaf_nC15_nC16_WT	Mole/Mass/Volume
mononaf_nC16_nC17_WT	Mole/Mass/Volume
mononaf_nC17_nC18_WT	Mole/Mass/Volume
mononaf_nC18_nC19_WT	Mole/Mass/Volume
mononaf_nC19_nC20_WT	Mole/Mass/Volume
mononaf_nC20_nC21_WT	Mole/Mass/Volume
mononaf_nC21_nC22_WT	Mole/Mass/Volume
mononaf_nC22_nC23_WT	Mole/Mass/Volume
mononaf_nC23_nC24_WT	Mole/Mass/Volume
mononaf_nC24_nC25_WT	Mole/Mass/Volume
mononaf_nC25_nC26_WT	Mole/Mass/Volume
mononaf_nC26_nC27_WT	Mole/Mass/Volume
mononaf_nC27_nC28_WT	Mole/Mass/Volume
mononaf_nC28_nC29_WT	Mole/Mass/Volume
mononaf_nC29_nC30_WT	Mole/Mass/Volume
mononaf_nC30_nC31_WT	Mole/Mass/Volume
othernaf_nC6_nC7_WT	Mole/Mass/Volume
othernaf_nC7_nC8_WT	Mole/Mass/Volume

othernaf_nC8_nC9_WT	Mole/Mass/Volume
othernaf_nC9_nC10_WT	Mole/Mass/Volume
othernaf_nC10_nC11_WT	Mole/Mass/Volume
othernaf_nC11_nC12_WT	Mole/Mass/Volume
othernaf_nC12_nC13_WT	Mole/Mass/Volume
othernaf_nC13_nC14_WT	Mole/Mass/Volume
othernaf_nC14_nC15_WT	Mole/Mass/Volume
othernaf_nC15_nC16_WT	Mole/Mass/Volume
othernaf_nC16_nC17_WT	Mole/Mass/Volume
othernaf_nC17_nC18_WT	Mole/Mass/Volume
othernaf_nC18_nC19_WT	Mole/Mass/Volume
othernaf_nC19_nC20_WT	Mole/Mass/Volume
othernaf_nC20_nC21_WT	Mole/Mass/Volume
othernaf_nC21_nC22_WT	Mole/Mass/Volume
othernaf_nC22_nC23_WT	Mole/Mass/Volume
othernaf_nC23_nC24_WT	Mole/Mass/Volume
othernaf_nC24_nC25_WT	Mole/Mass/Volume
othernaf_nC25_nC26_WT	Mole/Mass/Volume
othernaf_nC26_nC27_WT	Mole/Mass/Volume
othernaf_nC27_nC28_WT	Mole/Mass/Volume
othernaf_nC28_nC29_WT	Mole/Mass/Volume
othernaf_nC29_nC30_WT	Mole/Mass/Volume
othernaf_nC30_nC31_WT	Mole/Mass/Volume
monoarom_nC6_nC7_WT	Mole/Mass/Volume
monoarom_nC7_nC8_WT	Mole/Mass/Volume
monoarom_nC8_nC9_WT	Mole/Mass/Volume
monoarom_nC9_nC10_WT	Mole/Mass/Volume
monoarom_nC10_nC11_WT	Mole/Mass/Volume
monoarom_nC11_nC12_WT	Mole/Mass/Volume
monoarom_nC12_nC13_WT	Mole/Mass/Volume
monoarom_nC13_nC14_WT	Mole/Mass/Volume
monoarom_nC14_nC15_WT	Mole/Mass/Volume
monoarom_nC15_nC16_WT	Mole/Mass/Volume
monoarom_nC16_nC17_WT	Mole/Mass/Volume
monoarom_nC17_nC18_WT	Mole/Mass/Volume
monoarom_nC18_nC19_WT	Mole/Mass/Volume
monoarom_nC19_nC20_WT	Mole/Mass/Volume
monoarom_nC20_nC21_WT	Mole/Mass/Volume
monoarom_nC21_nC22_WT	Mole/Mass/Volume

monoarom_nC22_nC23_WT	Mole/Mass/Volume
monoarom_nC23_nC24_WT	Mole/Mass/Volume
monoarom_nC24_nC25_WT	Mole/Mass/Volume
monoarom_nC25_nC26_WT	Mole/Mass/Volume
monoarom_nC26_nC27_WT	Mole/Mass/Volume
monoarom_nC27_nC28_WT	Mole/Mass/Volume
monoarom_nC28_nC29_WT	Mole/Mass/Volume
monoarom_nC29_nC30_WT	Mole/Mass/Volume
monoarom_nC30_nC31_WT	Mole/Mass/Volume
diarom_nC11_nC12_WT	Mole/Mass/Volume
diarom_nC12_nC13_WT	Mole/Mass/Volume
diarom_nC13_nC14_WT	Mole/Mass/Volume
diarom_nC14_nC15_WT	Mole/Mass/Volume
diarom_nC15_nC16_WT	Mole/Mass/Volume
diarom_nC16_nC17_WT	Mole/Mass/Volume
diarom_nC17_nC18_WT	Mole/Mass/Volume
diarom_nC18_nC19_WT	Mole/Mass/Volume
diarom_nC19_nC20_WT	Mole/Mass/Volume
diarom_nC20_nC21_WT	Mole/Mass/Volume
diarom_nC21_nC22_WT	Mole/Mass/Volume
diarom_nC22_nC23_WT	Mole/Mass/Volume
diarom_nC23_nC24_WT	Mole/Mass/Volume
diarom_nC24_nC25_WT	Mole/Mass/Volume
diarom_nC25_nC26_WT	Mole/Mass/Volume
diarom_nC26_nC27_WT	Mole/Mass/Volume
diarom_nC27_nC28_WT	Mole/Mass/Volume
diarom_nC28_nC29_WT	Mole/Mass/Volume
diarom_nC29_nC30_WT	Mole/Mass/Volume
diarom_nC30_nC31_WT	Mole/Mass/Volume
diarom_nC31_nC32_WT	Mole/Mass/Volume
triarom_nC17_nC18_WT	Mole/Mass/Volume
triarom_nC18_nC19_WT	Mole/Mass/Volume
triarom_nC19_nC20_WT	Mole/Mass/Volume
triarom_nC20_nC21_WT	Mole/Mass/Volume
triarom_nC21_nC22_WT	Mole/Mass/Volume
triarom_nC22_nC23_WT	Mole/Mass/Volume
triarom_nC23_nC24_WT	Mole/Mass/Volume
triarom_nC24_nC25_WT	Mole/Mass/Volume
triarom_nC25_nC26_WT	Mole/Mass/Volume

triarom_nC26_nC27_WT	Mole/Mass/Volume
triarom_nC27_nC28_WT	Mole/Mass/Volume
triarom_nC28_nC29_WT	Mole/Mass/Volume
triarom_nC29_nC30_WT	Mole/Mass/Volume
triarom_nC30_nC31_WT	Mole/Mass/Volume
triarom_nC31_nC32_WT	Mole/Mass/Volume
triarom_nC32_nC33_WT	Mole/Mass/Volume
triarom_nC33_nC34_WT	Mole/Mass/Volume
fourplusarom_nC20_nC21_WT	Mole/Mass/Volume
fourplusarom_nC21_nC22_WT	Mole/Mass/Volume
fourplusarom_nC22_nC23_WT	Mole/Mass/Volume
fourplusarom_nC23_nC24_WT	Mole/Mass/Volume
fourplusarom_nC24_nC25_WT	Mole/Mass/Volume
fourplusarom_nC25_nC26_WT	Mole/Mass/Volume
fourplusarom_nC26_nC27_WT	Mole/Mass/Volume
fourplusarom_nC27_nC28_WT	Mole/Mass/Volume
fourplusarom_nC28_nC29_WT	Mole/Mass/Volume
fourplusarom_nC29_nC30_WT	Mole/Mass/Volume
fourplusarom_nC30_nC31_WT	Mole/Mass/Volume
fourplusarom_nC31_nC32_WT	Mole/Mass/Volume
fourplusarom_nC32_nC33_WT	Mole/Mass/Volume
fourplusarom_nC33_nC34_WT	Mole/Mass/Volume
fourplusarom_nC34_nC35_WT	Mole/Mass/Volume
benzene	Mole/Mass/Volume
toluene	Mole/Mass/Volume
C2_alkylbenzenes	Mole/Mass/Volume
C3_alkylbenzenes	Mole/Mass/Volume
C4_alkylbenzenes	Mole/Mass/Volume
C5_alkylbenzenes	Mole/Mass/Volume
C6_alkylbenzenes	Mole/Mass/Volume
C7_alkylbenzenes	Mole/Mass/Volume
C8_alkylbenzenes	Mole/Mass/Volume
C9_alkylbenzenes	Mole/Mass/Volume
C10_alkylbenzenes	Mole/Mass/Volume
C11_alkylbenzenes	Mole/Mass/Volume
C12_alkylbenzenes	Mole/Mass/Volume
C13_alkylbenzenes	Mole/Mass/Volume
C14_alkylbenzenes	Mole/Mass/Volume
C15_alkylbenzenes	Mole/Mass/Volume

C16_plus_alkylbenz	Mole/Mass/Volume
indane	Mole/Mass/Volume
methylindans	Mole/Mass/Volume
tetralin	Mole/Mass/Volume
C2_indans_and_methyl_tetralins	Mole/Mass/Volume
C3_indans_and_C2_tetralins	Mole/Mass/Volume
C4_indans_and_C3_tetralins	Mole/Mass/Volume
C5_indans_and_C4_tetralins	Mole/Mass/Volume
other_indans_tetralins_and_polynap	Mole/Mass/Volume
naphthalene	Mole/Mass/Volume
methyl_naphthalenes	Mole/Mass/Volume
C2_naphthalenes	Mole/Mass/Volume
C3_naphthalenes	Mole/Mass/Volume
C4_naphthalenes	Mole/Mass/Volume
C5_naphthalenes	Mole/Mass/Volume
C6_naphthalenes	Mole/Mass/Volume
C7_and_higher_naphthalenes	Mole/Mass/Volume
biphenyl	Mole/Mass/Volume
methyl_biphenyls	Mole/Mass/Volume
acenaphthene	Mole/Mass/Volume
fluorene	Mole/Mass/Volume
methyl_fluorenes	Mole/Mass/Volume
anthracene_and_phenanthrene	Mole/Mass/Volume
C1_tri aromatics	Mole/Mass/Volume
C2_tri aromatics	Mole/Mass/Volume
C3_tri aromatics	Mole/Mass/Volume
C4_tri aromatics	Mole/Mass/Volume
C5_and_higher_tri aromatics	Mole/Mass/Volume
fluoranthene	Mole/Mass/Volume
pyrene	Mole/Mass/Volume
C1_4ring_ aromatics	Mole/Mass/Volume
C2_4ring_ aromatics	Mole/Mass/Volume
C3_plus_4ring_ aromatics	Mole/Mass/Volume
Hydrogen Gas(mols/s)	None
Hydrogen Gas(g/s)	None
Hydrogen Gas(molefrac)	None
Hydrogen Gas(Wtfrac)	None
CetaneNumber	None
CloudPoint(K)	None

Carbazole_WT	Mole/Mass/Volume
Ammonia_WT	Mole/Mass/Volume

Table 39: Species Properties in Property Database

Species Property	Index
Tref	0
Tc	1
Pc	2
Vc	3
Tb	4
Tm	5
Hform	6
Gform	7
CPa	8
CPb	9
CPc	10
CPd	11
Hfusion	12
Hvap	13
LogKw	14
Fp	15
Hvb	16
Svb	17
SigmaD	18
SigmaP	19
SigmaH	20
Sigma	21
Omega	22
Vm	23
Visa	24
Visb	25
CP	26
VIS	27
TotalCarbonNum	28
AromCNum	29
NaphCNum	30
AromRingNum	31
NaphRingNum	32
MW	33
Density	34

TotalHydrogenNum	35
Nap5RingNum	36
Nap6RingNum	37
SideChains	38
ThphRingNum	39
TotalSulfurNum	40
TotalNitrogenNum	41
Quan_HF	42
AromHydrogenNum	43
AlphaHydrogenNum	44
ThioSulfurNum	45
SulfideSulfurNum	46
MercaptansSulfurNum	47
PyridineNitrogenNum	48
PyrroleNitrogenNum	49
TotalOxygenNum	50
ZNum	51
DBE	52
AlkylaAroCNum	53
PhenolAroCNum	54
FusedAroCNum	55
AmineNum	56
AmideNum	57
AlcoholNum	58
EtherNum	59
CarboxylicAcidNum	60
EsterNum	61
CarbonylNum	62
AldehydeNum	63
MethylNum	64
SaturatedCNum	65
UserClass1st	66
UserClass2nd	67

A.3 Modified Lignin Structures

Lignin pyrolysis required an approximation of the initial lignin structure.

Different lignin structures were developed and are illustrated below.

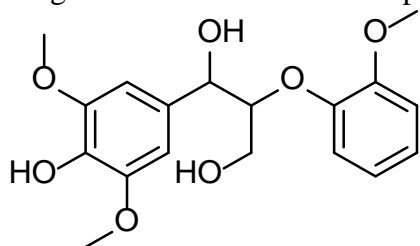


Figure 85: Lignin dimer β -O-4 linkage

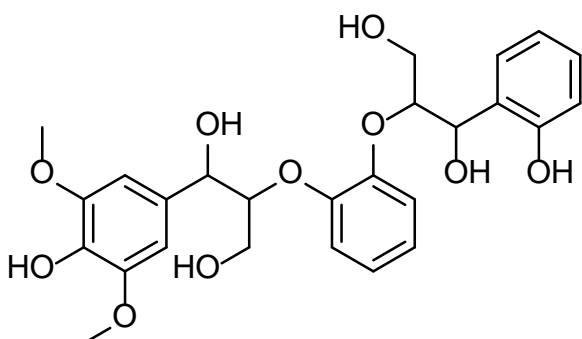


Figure 86: Lignin trimer, 2 β -O-4 linkage

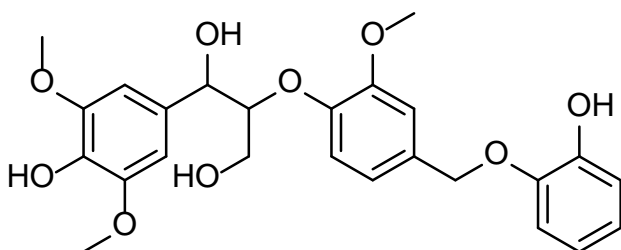


Figure 87: Lignin trimer, 1 β -O-4 linkage & 1 α -O-4 linkage

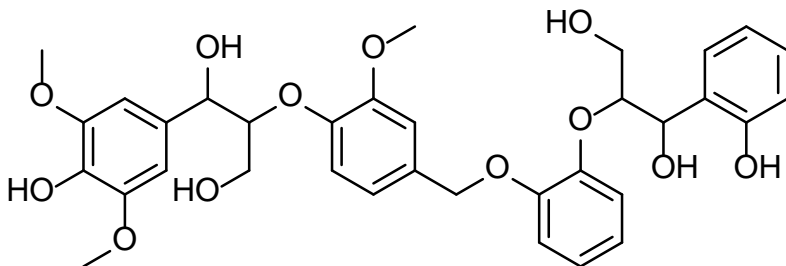


Figure 88: Lignin tetramer, 2 β -O-4 linkage & 1 α -O-4 linkage

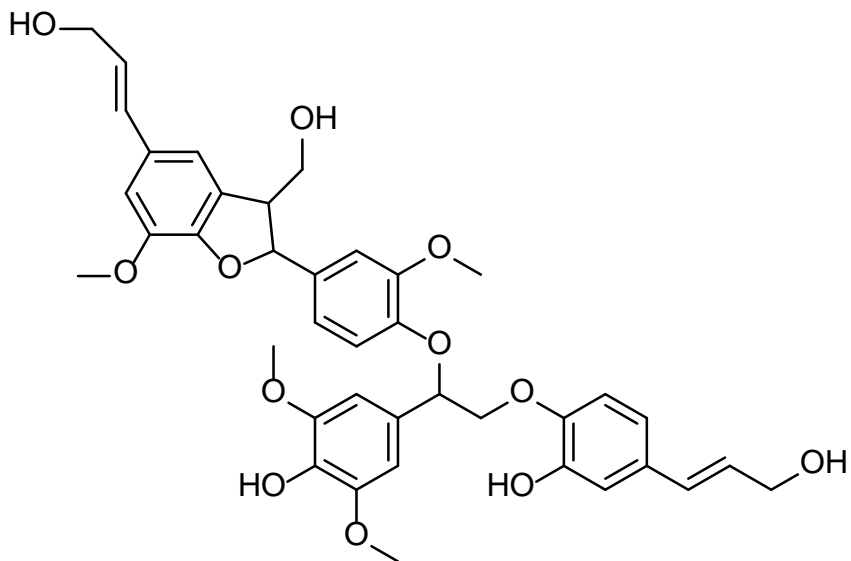


Figure 89: Lignin tetramer, 1 β -O-4 linkage & 1 α -O-4 linkage

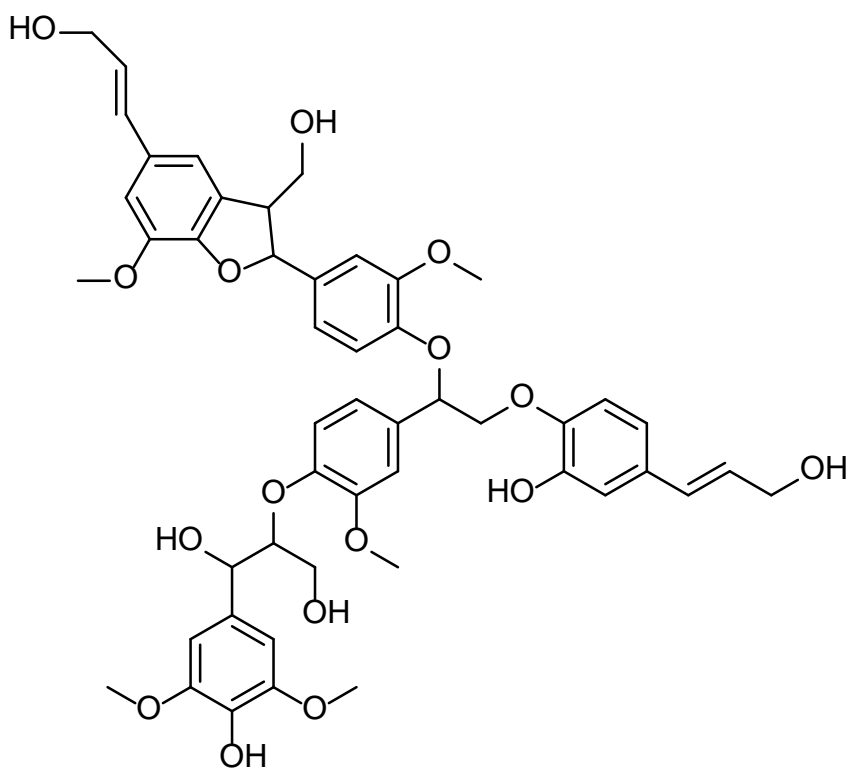


Figure 90: Lignin pentalignol, 3 β -O-4 linkage & 1 α -O-4 linkage

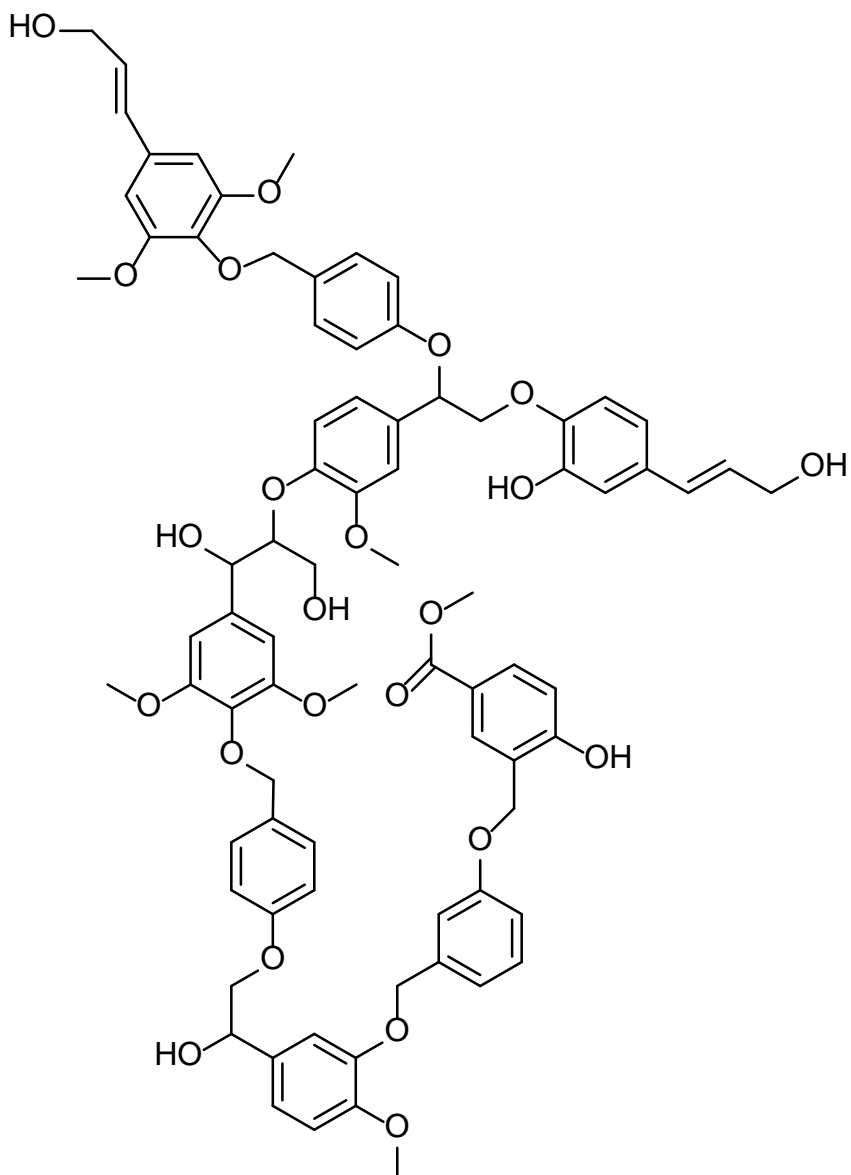


Figure 92: Lignin Nonalignol, 5 β -O-4 linkage & 4 α -O-4 linkage