Development of Automatic Chemical Reaction Mechanism Generation Software Using Object-Oriented Technology

Jing Song, Sumathy Raman, Joanna Yu, Catherina D Wijaya, George Stephanopoulos, and William H. Green

Department of Chemical Engineering,

Massachusetts Institute of Technology,

77 Massachusetts Avenue, Cambridge, MA 02139, USA

Introduction

Computer-aided reaction mechanism generation has been employed by many groups to model multi-component reacting mixtures, in processes such as pyrolysis and fuel combustion (Chinnick, et al., 1988; Hillewaert, et al. 1988; Chevalier, et al. 1990; Froment, 1991; DiMaio and Lignola, 1992; Quann and Jaffe, 1992; Broadbelt, et al., 1994; Blurock, 1995; Broadbelt, et al., 1995; Ranzi, et al., 1995; Broadbelt, et al., 1996; Prickett and Mavrovouniotis 1997a, 1997b, 1997c; Susnow, et al., 1997; Warth, et al., 2000). Good model-generation software is desired to quickly build more detailed and reasonably-structured reaction mechanisms with better kinetics parameters. With the efforts from those groups, people have successfully represented the chemical structures and properties of reaction systems and generated large reaction mechanisms. However, some common shortcomings in the existing software prevent their wide usage in chemical reaction simulations. For example, many fewer reaction patterns are considered in such software than should be; furthermore, the thermal and kinetics parameters are not specific enough to account for different types of reaction pathways. How to extend the model generation software to accommodate more types of chemical reactions and/or allow users to create their own desired reaction types easily and how to systematically integrate the available thermal and kinetics data from different sources remained unsolved before this work.

Many previous chemical software applications were developed in procedural languages, like Fortran and C. Although those languages are good for calculation efficiency, they make it difficult to satisfy the important requirements of ease of maintenance, reusability and extendibility, when we design and develop software applications for large complex systems. In recent years, object-oriented technology has been rapidly developed to satisfy those basic requirements of good software. Corresponding modeling languages, such as unified modeling language (UML), and programming languages, such as C++, Java, and C#, have been widely used in developing better-structured software. In this work, we made use of those advanced research fruits to develop a well-structured, reusable and extendable automatic reaction mechanism generation software, RMG.

Representing Chemical Structures by Graphs

Graphs, as fundamental data structures, and their associated algorithms have been widely used for uniquely representing chemical structures and estimating chemical properties in most chemical application software.

In RMG, we also used 2-dimensional graph data structures representing individual chemical species and chemical functional groups. The estimation of all the species features, such as symmetry number, resonance isomer structures, and cycle identification, etc. are implemented based on basic graph operations. The thermal properties are estimated by a group additivity method proposed by Benson (1968). Unlike chemical species with the unique graph representation, functional group describes a special group of chemical species with the same chemically functional centers. Introducing functional group object into reaction generation software allows one to easily define the reaction pattern later. Moreover, we developed a key matching algorithm to quickly identify the sub-graph relationship between species and functional groups and between two functional groups, which speed recognition of all the chemical reaction patterns existing in any species.

Defining Reaction Families

In most existing reaction model generation software, the reaction family definitions are hard-coded; this makes it very difficult for users to change the existing reaction families and to add any new ones. In RMG, we define the reactant families by drawing their functional groups and indicating the graph mutations happening to the reactants through the course of the reaction. Therefore, new reaction families can be easily input through a graphical interface, without the need for the user to modify the source code. This makes it much easier for normal users other than software developers to vary the present reaction family specification, to define new families of reactions, and to document the exact assumptions behind the models they construct. In this work, thirty-three primary reaction families, including eighteen forward families and fifteen reverse ones, are defined; this is the richest set of reaction families ever compiled. The summarization of all the reaction families is shown in table 1.

Table 1. Reaction Families Defined in RMG

Forward Reaction Family (Reverse Reaction Family)
(1) Inter-molecular Hydrogen Abstraction
(2) Radical Addition to Multiple Bond (Beta Scission)
(3) Radical Addition to CO (CO Elimination from Carbonyl)
(4) Radical Recombination (Bond Dissociation)
(5) Bi-radical Recombination to Form Cyclic Structure (Ring open)
(6) Disproportionation (Molecular Addition)
(7) 1,2 Insertion (1,1 Elimination)
(8) 1,3 Insertion (1,2 Elimination)
(9) 1+2 Cyclo-addition (Three-Ring Cleavage)
(10) 2+2 Cyclo-addition (Four-Ring Cleavage)
(11) Diels-Alder Addition (Retro Diels-Alder Addition)
(12) Keto-Enol Tautomerism (Enol-Keto Tautomerism)
(13) Intra-molecular Hydrogen Migration
(14) Intra-Molecular Addition across Multiple Bond to Form an Exo-Cyclic Radical (Ring Open for Exo-Cyclic Radical)
(15) Intra-Molecular Addition across Multiple Bond to Form an Endo-Cyclic Radical (Ring Open for Endo-Cyclic Radical)
(16) Cyclic Ether Formation from Alkyl-Hydroperoxy Radical (OH Addition to Cyclic Ether)
(17) Intra-Molecular Hydroxyl Migration
(18) HO_2 Elimination from Peroxy Radical

Constructing Thermal Groups and Reaction Kinetics database by Hierarchical Tree

With the technology described above, we are able to enumerate all the reactions occurring among any chemical species. However, a more important question still remains unanswered: how to get the best reaction kinetics parameter for a generated reaction?

Owing to the work of many experimental and computational chemical kinetics groups, now there are a large number of good-quality reaction kinetics data available. However, in much previous model generation software, only a small number of data are used to account for a huge group of reactions. Such approach is due in part to the high running time cost to construct and search through a huge kinetics database.

In this work, we construct systematically our reaction family kinetics database by a hierarchical tree structure, which can hold a large number of kinetics data for different subfamilies. Consequently, the cost for searching tree-structured database is reduced from O(N) to O(logN), where N is the overall number of kinetics data in one database. This dramatic reduction in data retrieving time makes it possible to break one big reaction family into numerous subfamilies with more precise parameters (e.g. accounting for nearest-neighbor effects on the rates); therefore, more precise kinetics data will be assigned to each individual reaction. So far, we have built and maintained the kinetics database into hierarchical-tree structure for all the reaction families. However, the database we constructed is not limited to its present status, and it is easy for anybody to extend the tree structure by adding new sub-nodes and their corresponding kinetics data. We also have filled in the kinetics database mainly with high level quantum chemistry calculation results from our group, (Sumathi, et al., 2001 and 2002; Wijaya, C.D., et al., 2003), and from the Livermore group (Curran, et al., 2002).

Similarly, the thermo-chemical group data are also stored and recalled for group additivity estimation of the thermal properties of each species. The thermal group data are originally from Benson's estimation, and the database could be extended in the same way as the reaction kinetics database.

Iterative Mechanism Generation Algorithm

This work adopted the rate-based reaction path screening method proposed by Susnow, et al. (1997) to identify the reactions and species required to satisfy the user-defined accuracy requirements. Since this reaction pathway screening algorithm greatly depends on the kinetics parameters of each competing reaction path, building a big and well-structured kinetics database seems to be more crucial in such circumstance. This, from another aspect, proves the importance of our hierarchical tree solution to building large and detailed kinetics databases. This work also implements the valid parameter range analysis algorithms, presented in Song et al. (2002), to construct robust chemical reaction mechanism under a wider range of reaction system conditions.

Applying Object-oriented Technology in RMG Design and Development

Object-oriented technology is the most advanced breakthrough in software engineering methodology in last decade, and it has been rapidly improved and broadly applied in most software applications. To assist in the development of good architecture for object-oriented software, unified modeling language, UML, has been broadly used.

Introduction to Unified Modeling Language (UML). UML is a modeling language to help build an unambiguous and well-structured blueprint of a software at the design stage. With a graphic visualization, UML not only specifies the static relations between objects inside system, but also defines the dynamic communications between different parts of software at different time. With the aid of UML, people in the different stages of software development, such as design, implementation, test, and maintenance, can talk to each other with the same design language for a better communication with much less misunderstanding. There are many commercial implementation tools of UML, for example, Rose from Rational and Rhapsody from I-Logix.

Design and Implementation of RMG in UML. The automatic reaction mechanism generation software developed by our group, RMG, is fully designed and implemented in UML. Rhapsody in J is the UML tool we adopted in this work. There are four major packages in RMG, chemUtil, chem, rxn, and rxnSystem. ChemUtil defines the basic data structure objects, Graph and Tree, used in RMG; Chem defines the fundamental chemical components such as Atom, Bond, Electron, etc.; Rxn describes the common structure of Reactions in RMG, as well as the reaction generator from defined reaction families; RxnSystem performs the iterative reaction mechanism generation algorithm and the simulation of a homogeneous reaction system. Since RMG is designed and implemented in an object-oriented manner, it is easily extended and re-factored further. For instance, although the present species model is a 2-dimensional graph, it can be later extended into a 3-dimensional model by simply adding coordinates of graph nodes. Beside designed in UML, all the packages are written in Java and also documented in the standard Java Doc format.

Applications

Applications of RMG for automatically generating chemical reaction mechanisms for a complex liquid-fuel combustion process are studied.

Conclusion

In this work, we successfully designed and developed a new chemical reaction mechanism generation Java software, RMG, using object-oriented technology. Two advanced technologies, graph representation of reaction families and a hierarchy tree-structured database for retrieving thermal and kinetics parameters, have been proposed and implemented.

Acknowledgement. We acknowledge financial support from Alstom Power, the National Science Foundation, and US Department of Energy.

References

(1) Blurock, E. S. J. Chem. Inf. Comput. Sci., **1995**, 35, 607.
(2) Benson, S.W., Thermochemical kinetics, **1968**, John Wiley & Sons, Inc.
(3) Broadbelt, L. J.; Stark, S. M.; Klein, M. T. *Ind. Eng. Chem. Res.* **1994**, *33*, 790.
(4) Broadbelt, L. J.; Stark, S. M.; Klein, M. T. *Ind. Eng. Chem. Res.* **1995**, *34*, 2566.
(5) Broadbelt, L. J.; Stark, S. M.; Klein, M. T. *Computers Chem. Engng.* **1996**, *20*, 113.
(6) Chevalier, C.; Warnatz, J.; Melenk, H. *Ber. Bunsenges. Phys. Chem.* **1990**, *94*, 1362.
(7) Chinnick, S. J.; Baulch, D. L.; Ayscough, P. B. *Chemometrics and Intelligent Laboratory Systems*. **1988**, 5, 39.
(8) Curran, H.J.; Gaffuri, P.; Pitz, W.J.; Westbrook, C.K.. *Combustion and Flame*, **2002**, 129, 253.
(9) DiMaio, F. P.; Lignola, P. G. *Chem. Eng. Sci.*, **1992**, 47, 2713.
(10) Froment, G. F. *Chemical Reactions in complex systems: the Mobil Workshop*, **1991,** A. V. Sapre and F. J. Krambeck. New York, Van Nostrand Reinhold.
(11) Hillewaert, L. P.; Diericks, J. L.; Froment, G. F. *AIChE J.*, **1988**, 34, 17.
(12) Prickett, S. E.; Mavrovouniotis, M. L. *Comp. Chem. Engn*, **1997a**, 21, 1219.
(13) Prickett, S. E.; Mavrovouniotis, M. L. *Comp. Chem. Engn*, **1997b**, 21, 1237.
(14) Prickett, S. E.; Mavrovouniotis, M. L. *Comp. Chem. Engn*, **1997a**, 21, 1325.
(15) Quann, R. J.; Jaffe, S. B. *Ind. Eng. Chem. Res.* **1992**, 31, 2483.
(16) Song, J.; Stephanopoulos G.; Green, W.H., *Chem. Eng. Sci.*, **2002**, 57, 4475.
(17) Susnow, R. G., Dean, A. M., Green, W. H., Peczak, P. and Broadbelt, L. J. *J. Phys. Chem. A*, **1997**, 101, 3731.
(18) Warth, V., Battin-Leclerc, F., Fournet, P. A., Glaude, P. A., Come, G. M. and Scacchi, G. *Computers and Chemistry*, **2000**, 24, 541.
(19) Sumathi, R.; Carstensen, H.-H.; Green, W. H. *J. Phys. Chem. A*, **2001**, 105, 6910.
(20) Sumathi, R.; Carstensen, H.-H.; Green, W. H. *J. Phys. Chem. A*, **2001**, 105, 8969.
(21) Sumathi, R.; Carstensen, H.-H.; Green, W. H. *J. Phys. Chem. A*, **2002**, 106, 5474.
(22) Sumathi, R.; Green, W. H. *Theor. Chem. Acc.*, **2002**, 108, 187-213.
(23) Wijaya, C.D.; Sumathi, R.; Green Jr., W. H. *J. Phys. Chem. A*, **2003**, (in press).