KUCRS – Detailed Kinetic Mechanism Generator for Versatile Fuel Components and Mixtures

*Akira Miyoshi
Department of Chemical Systems Engineering, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

Key Words: Autoignition, Kinetic Mechanism, Automatic Generation, Versatile Fuel

ABSTRACT

A software framework for the construction of detailed kinetic mechanisms of combustion for versatile fuel components and mixtures named KUCRS (knowledge-basing utilities for complex reaction systems) has been developed and its structure and essential features are described. Its major purpose is to automate the routine but complex tasks for the construction, improvement and maintenance of large chemical kinetic mechanisms for hydrocarbon combustion by knowledge- and rule-based methods. It is comprised of "chemgen" (reaction macro processor), "thermgen" and "trangen" (thermodynamic and transport data generators), "combust" (main mechanism generator) and interface tools. The system features classes (program units) for the identification of chemical species by unique internal representation which takes into account the electronic structures of resonance stabilized radicals, for the administration of names and status of chemical species, as well as for the detection of duplicated reactions and dead-end species. The rate rules are defined in external text files and can be revised by mechanism developers without modifying the source code. It generates kinetic mechanism, thermodynamic and transport data compliant with Chemkin-II software and it can import/export chemical species in SMILES representation. Current version of the software can generate oxidation mechanisms for non-cyclic and mono-cyclic alkanes and non-cyclic alkenes (mono-ens) and alcohols, as well as for any mixtures consisting of any number of supported components. For multiple component fuels, it also generates cross reactions important in the low temperature oxidation.

INTRODUCTION

Due to the demand for carbon neutral biofuels to reduce the carbon dioxide emission, changes of the fossil energy sources including the oil sands and shale gas, and increasing regulations for the automotive exhaust, more and more knowledge on the chemical kinetics of the combustion of diversified fuel compounds is needed. However, the kinetic mechanisms of the oxidation of hydrocarbons often contain thousands of elementary chemical reactions [1,2], especially those including the low-temperature oxidation mechanisms inevitable for the reproduction and understanding of the autoignition phenomena utilized in the next-generation technologies such as homogeneous-charge compression-ignition (HCCI). Although the elementary reactions in the H_2-O_2 system and small hydrocarbon-O_2 systems have been extensively studied and fairly well established, only a limited number of reactions have been investigated in detail for large hydrocarbons, and the rate parameters for most of the thousands of reactions have been estimated by analogy to those of small hydrocarbons [1,2]. Such estimates often involve crude assumptions and the rate parameters and the mechanism need to be revised in the near future. For example, the reaction routes of hydroperoxyalkylperoxy radicals and their rate parameters are subject to change when new experimental or theoretical investigation becomes available and may be "tuned" within the estimated error limits to reproduce the observed overall phenomena. In order to assure the validity of the rule-based method and to keep the consistency among the oxidation mechanisms for various, but the same class of, fuels (e.g., alkanes), the same rule should be used throughout the mechanisms for the same class of fuels. However, the manual procedure for improvement and maintenance of the rules is becoming more and more impractical since the number of reactions increases with the size and variety of the fuel molecules.

Clearly, one promising solution to this problem is to automate the mechanism generation procedures by computer software. For more than three decades, many research groups have elaborated to develop software for this purpose [3–14]. However, only a few of them seem to be efficiently and continuously used for later improvement and maintenance. The software framework, named KUCRS (knowledge-basing utilities for complex reaction systems), described in this paper not only possesses some new features, but is also intended to be a robust and easy-to-use workbench for various stages of the development and maintenance of detailed kinetic mechanisms, and to reduce the time and effort required for the routine and complex tasks. The modular structure of the programs enables the developers to use a part of the programs separately, for example, for the generation of thermodynamic data from SMILES [15] inputs, or for maintaining the rule-based reaction rate constants by keeping them as user-defined macros.

SYSTEM STRUCTURE AND COMPOMENTS

The major part of the system is written in ANSI C++ language. Although the C++ codes may look complex and hard to read for those unfamiliar with the object-oriented languages, it has been chosen since it significantly reduces the effort in maintaining or customizing the system. For example, to generate a product radical formed by the hydrogen abstraction reaction from a fuel in the system, one only needs to add following two lines in the source code.

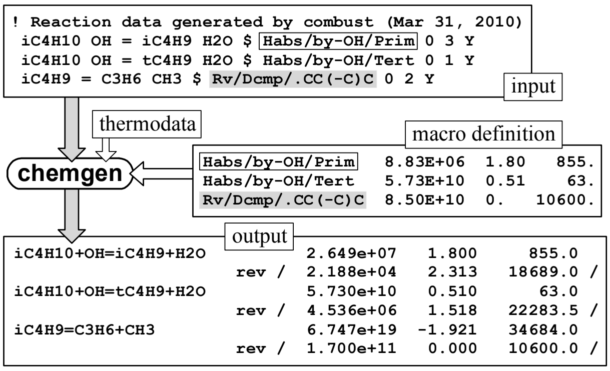
```
product = fuel;
product.beRadicalAt(position);
```

The KUCRS system consists of several program modules that can be used separately. They will be described below, from downstream to upstream, since it will be the best way to describe the structure of the system and, actually, the system has been developed in this order.

Reaction Macro Processor – "chemgen"

One of the most essential tasks required for maintaining large kinetic mechanisms is to keep the same rule-based rate parameters for the same class of reactions consistently and thoroughly. In KUCRS, this can be accomplished by the macro substitution mechanism of the program named "chemgen". Figure 1 depicts how this mechanism works. The input contains macros instead of the actual values of rate parameters. For example, the macro "Habs/by-OH/Prim" means the hydrogen abstraction reaction by OH radical from a primary site. The actual values of the macros are defined in a separate macro definition file. The program reads the input and macro definition files and outputs the mechanism in Chemkin [16] format. By keeping the input files for "chemgen" as master files, instead of the final mechanism files in Chemkin format, one can change the rule-based rate parameters by editing the macro definition file only and rerunning the program. Most of the rule-based rate parameters implemented in the earlier releases of KUCRS were based on the works by Curran et al. [1,2], but currently, they are based on the systematic quantum chemical calculations [17].

The input files contain some additional parameters after the names of the macros. The first parameter is the shift value of the activation energy, which can be used for some ad hoc purposes. The second parameter is the multiplication factor for the pre-exponential factor, which can be also used for ad hoc purposes, but in most cases in the system, it is used to specify the number of equivalent sites. For example for the hydrogen atom abstraction from the primary sites of isobutane in Fig. 1, there are three equivalent methyl groups, and the rate constant is multiplied by a factor of three; also in the thermal decomposition of *i*-butyl radical in the third input line, there are two equivalent methyl groups which can be eliminated. The last parameter is a flag ("Y" for "yes" and "N" for "no") indicating whether or not the rate parameters for the reverse reaction are automatically calculated by the detailed balance and written into the Chemkin input files with the "rev" keyword. For this purpose, the program requires thermodynamic data in Chemkin format as another input. Additionally, a special mechanism is invoked when a macro begins with a string "Rv/". For the decomposition reactions, it may be wise to specify the rate parameters for the reverse reactions as rules since the activation energies for the reverse recombination reactions are often similar and close to zero. For this purpose, when the prefix "Rv/" is found, the program recognizes the rate parameters defined in the macro as those for the reverse reaction. An example is shown as the third input line of Fig. 1.



```
! Reaction data generated by combust (Mar 31, 2010)
 iC4H10 OH = iC4H9 H2O $ Habs/by-OH/Prim 0 3 Y
 iC4H10 OH = tC4H9 H2O $ Habs/by-OH/Tert 0 1 Y
 iC4H9 = C3H6 CH3 $ Rv/Dcmp/.CC(-C)C 0 2 Y
```

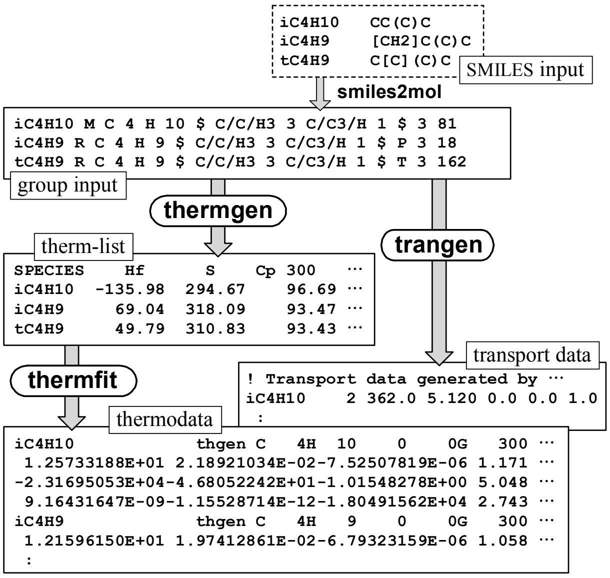
```
Habs/by-OH/Prim    8.83E+06  1.80    855.
Habs/by-OH/Tert    5.73E+10  0.51     63.
Rv/Dcmp/.CC(-C)C   8.50E+10  0.    10600.
```

```
iC4H10+OH=iC4H9+H2O         2.649e+07   1.800      855.0
                    rev /   2.188e+04   2.313    18689.0 /
iC4H10+OH=tC4H9+H2O         5.730e+10   0.510       63.0
                    rev /   4.536e+06   1.518    22283.5 /
iC4H9=C3H6+CH3              6.747e+19  -1.921    34684.0
                    rev /   1.700e+11   0.000    10600.0 /
```

Fig. 1 Reaction macro processor – "chemgen".

Thermodynamic and Transport Data Generators – "thermgen" and "trangen"

For thermodynamic data generation, KUCRS adopted THERM program [18] developed by Bozzelli and co-workers for its extensive group data and stability. It is based on the Benson's group additivity method [19] with a modification to use "bond-dissociation" groups for radicals. The program "thermgen" is essentially the same as the "therm" main program developed by them, but was rewritten for batch processing. Since KUCRS uses group database and "thermfit" program in it, users should obtain the THERM program separately. The program "trangen" generates transport data based on the group contribution method of Joback [20] for critical properties and their relations with Lennard-Jones potential parameters proposed by Chung et al. [20]. Figure 2 schematically displays the thermodynamic and transport data generation processes done by the programs



```
iC4H10   CC(C)C
iC4H9    [CH2]C(C)C
tC4H9    C[C](C)C
```

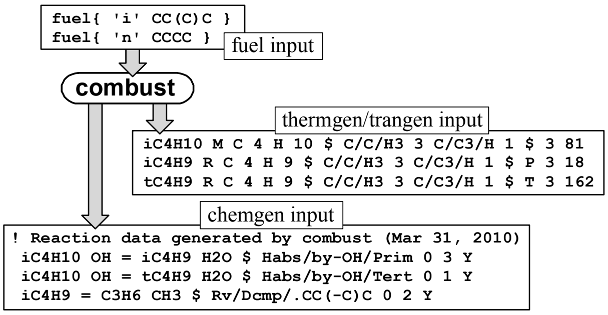
```
iC4H10 M C 4 H 10 $ C/C/H3 3 C/C3/H 1 $ 3 81
iC4H9 R C 4 H 9 $ C/C/H3 3 C/C3/H 1 $ P 3 18
tC4H9 R C 4 H 9 $ C/C/H3 3 C/C3/H 1 $ T 3 162
```

```
SPECIES     Hf       S     Cp 300   ···
iC4H10  -135.98  294.67     96.69  ···
iC4H9     69.04  318.09     93.47  ···
tC4H9     49.79  310.83     93.43  ···
```

```
! Transport data generated by ···
iC4H10    2 362.0 5.120 0.0 0.0 1.0
 :
```

```
iC4H10            thgen C   4H  10    0    0G   300 ···
 1.25733188E+01 2.18921034E-02-7.52507819E-06 1.171 ···
-2.31695053E+04-4.68052242E+01-1.01548278E+00 5.048 ···
 9.16431647E-09-1.15528714E-12-1.80491562E+04 2.743 ···
iC4H9             thgen C   4H   9    0    0G   300 ···
 1.21596150E+01 1.97412861E-02-6.79323159E-06 1.058 ···
 :
```

Fig. 2 Thermodynamic and transport data generators – "thermgen" and "trangen".



```
fuel{ 'i' CC(C)C }
fuel{ 'n' CCCC }
```

```
iC4H10 M C 4 H 10 $ C/C/H3 3 C/C3/H 1 $ 3 81
iC4H9 R C 4 H 9 $ C/C/H3 3 C/C3/H 1 $ P 3 18
tC4H9 R C 4 H 9 $ C/C/H3 3 C/C3/H 1 $ T 3 162
```

```
! Reaction data generated by combust (Mar 31, 2010)
 iC4H10 OH = iC4H9 H2O $ Habs/by-OH/Prim 0 3 Y
 iC4H10 OH = tC4H9 H2O $ Habs/by-OH/Tert 0 1 Y
 iC4H9 = C3H6 CH3 $ Rv/Dcmp/.CC(-C)C 0 2 Y
```

Fig. 3 Main mechanism generator – "combust".

"thermgen", "thermfit", and "trangen". The group input consists of the chemical composition, groups contained in the molecule, bond-dissociation group if it is a radical, number of internal rotors, and symmetry number including those for internal rotors. The same group input file is used for the thermodynamic data generation and transport data generation. As shown in Fig. 2, a tool named "smiles2mol" can be used to prepare the group input file from SMILES input.

Main Mechanism Generator – "combust"

The task done by the main mechanism generator named "combust" is illustrated in Fig. 3. It requires only one input file containing the chemical structures of fuels in SMILES representation. The program generates all the necessary species which are not found in the base mechanism into the input file for "themgen" and "trangen", and their reactions into the input file for "chemgen". The built-in base mechanism in the current release of KUCRS contains H_2/O_2 reactions in LEEDS methane mechanism [21] and C_1–C_2 reactions adopted from LLNL iso-octane mechanism [2]. The base mechanism is fully user-customizable; for example, it may contain mechanism up to C_3 or C_4. Major features implemented in this program will be described below in the SYSTEM FEATURES section.

Molecular Library Interface Tools

These are the tools to maintain the molecular library in the system. The first part of the library, *base library*, contains the chemical species in the base mechanism, and the program "combust" does not generate further reactions of them. The second part, *name library*, contains the species for name reference. The program does generate reactions of these species, but by using the names defined here. So, the name library provides a way to control the names of the chemical species automatically given by the program. It should be noted that the generator gives unique names for species generated in one job, but it does not warrant the uniqueness of the names among the separately generated mechanisms. The "mollib" program converts the molecules in arbitrary internal format to canonical (i.e., unique) ones, and dumps the molecular structures in human-readable forms including SMILES representation. The "smiles2mol" can be used to construct the molecular library from SMILES strings. The "mol2smiles" program converts molecular structures in the internal format to SMILES.

SYSTEM FEATURES

In this section, major features of the automatic mechanism generation program "combust" are described. Most of the *objects* in the system are kept in dynamic arrays and there is no size limit due to the hard-wired declarations. For example, there is no limit for the number of fuel input, number of total species, number of atoms contained in a molecule, etc. A limit may arise only when the shortage of the system resource, memory or disk storage, is encountered, though the author has never experienced such situation for up to C_{16} fuels.

Identification of Chemical Species

An object of the "molecule" class contains sufficient information necessary for the generation of gas-phase reactions in the edge(bond)-vertex(atom) representation. One of the most essential features of the "molecule" is the unique identification of the chemical species. The concept of the species identification is illustrated in Fig. 4 for some simple examples. For large molecules, the chemical bonding structure can be drawn in various ways. For example for the 3-methylpentane shown in Fig. 4a, it can be drawn, in the IUPAC rule, as a five-carbon chain with a methyl branch at the third carbon, but can be also represented by a four-carbon chain with an ethyl branch at either second or third position. Since they are only the different representations of the same chemical structure, they must be recognized to be the same in the system. This can be accomplished by keeping the structure in a "molecule" object always in the canonical form. The canonical form in the system is similar to the IUPAC naming rule, but not exactly the same since the rules that can be achieved faster in computation were adopted. Second example (Fig. 4b) shows the radicals formed by hydrogen abstraction reactions from isobutane. Since the three methyl groups in isobutane are equivalent, three radicals in Fig. 4b are identical. The identification can be properly done by keeping the structures in a canonical form shown in the right hand side of Fig. 4b. More complex and sophisticated example is the localized structures of a radical with allylic resonance shown in Fig. 4c. The radical produced by the hydrogen abstraction from the edge carbon of 2-butene and that produced by the hydrogen abstraction from third carbon of 1-butene are identical. The "molecule" class properly identifies the structures even with the multiple resonances, for example, those of the pentadienyl radical. The system can also list-up all the distinguishable

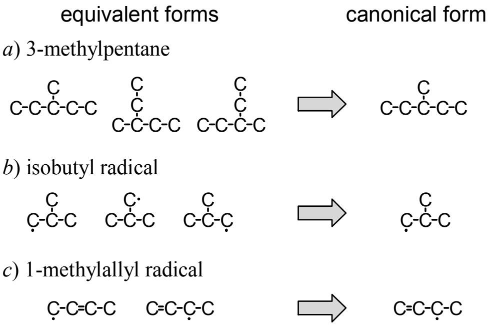


Fig. 4 Unique identification of chemical species.

Table 1 Some Member Functions of "molecule" Class

function	description

`operator==,`
`operator!=`
	equality and inequality operators

`beUnique()`	make the object canonical
`getChemFormula()`	returns a string containing the chemical formula
`beRadicalAt()`	remove an hydrogen atom at the specified position
`makeNewBond()`	make a new bond between two atoms specified
`decBondOrder()`	decrease the bond order between two atoms specified

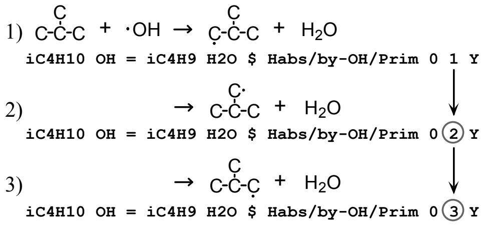
localized structures of such radicals since the recombination reactions can occur at either of the localized radical centers. For symmetric molecules, there are several equivalent structures even for the canonical form when quantum mechanically indistinguishable atoms are labeled. For example, isobutane has three rotationally commutable equivalent structures when carbon atoms are labeled. The number of the equivalent structures is also important in determining the rotational symmetry numbers in the evaluation of thermodynamic data. The group input for iC4H10 (isobutane) in Figs. 2 and 3 was generated by the program "combust". The program properly counts out the three equivalent structures in determining the rotational symmetry number and the resultant total symmetry number is $3\times3^3 = 81$ including those for three methyl groups. The program also properly counts out the symmetry number of *t*-butyl radical (tC4H9) as $6\times3^3 = 162$ as shown in Fig. 3.

Properties and Handling of Chemical Species

The "molecule" class has many functions to evaluate species properties. One of the important properties is the groups of the Benson's group additivity method [19]. All the group definitions and their group values are stored in external text files and users can add new groups or modify the values of them. The identification of groups is not only important in evaluating the thermodynamic and transport properties, but also useful to determine the rate constants for reactions since, for example, the rate parameters for H-atom abstractions from primary, secondary, and tertiary sites are usually different. The "molecule" class also provides manipulation functions that are needed to prepare products of the chemical reactions. Several typical examples of these functions are listed in Table 1.

Species Administration

As the number of the chemical species increases, the administration of the chemical species becomes one of the most time-consuming and complex tasks. In KUCRS system, this task is done by an object of the "specAdmin" class in cooperation with an object of the reaction generator class named "reaction". The 'reaction queue' in a "specAdmin" object contains chemical species for which reactions should be generated, and initially it contains fuel molecules only. The species in the queue are processed by the reaction generator by first-in first-out principle. During the course of reaction generation, many product species will appear, but the "reaction" only needs to throw them to "specAdmin". If the product species is new to the system, it generates a new name, not duplicating with the existing names, and return the name to the "reaction". The new species is put into the queue to generate its reactions later. If the product species is already in the queue or in the 'done' list, "specAdmin" returns its name without doing anything else. At the end of the reaction generation "reaction" returns the reactant species to "specAdmin" to store it in the 'done' list. This mechanism of queue avoids the generation of 'dead-end' species, which is produced during the reaction but no subsequent reaction of it is written at all, and often causes significant problem during the numerical simulation since it remains unreacted even after the thermal ignition. Also the 'done' list avoids unintended duplication of reactions.



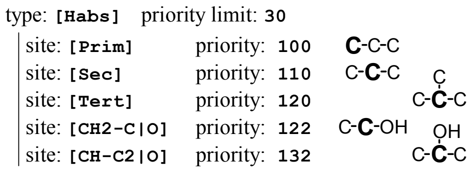
```
iC4H10 OH = iC4H9 H2O $ Habs/by-OH/Prim 0 1 Y
iC4H10 OH = iC4H9 H2O $ Habs/by-OH/Prim 0 2 Y
iC4H10 OH = iC4H9 H2O $ Habs/by-OH/Prim 0 3 Y
```

Fig. 5 Output buffering mechanism to avoid duplicated reaction generation.

Reaction Generation

The reaction generator retrieves the top entry of the reaction queue and calls a type-specific reaction processor according to the type of the reactant. During the generation of the same type of the reactions, the list of the reactions is kept in an output buffer to avoid the duplicated reactions. The mechanism is depicted in Fig. 5. In many cases for hydrocarbon species, the reactant has several equivalent sites for reactions, for example the three equivalent methyl groups of isobutane in Fig. 5. By keeping the list of reaction temporarily, the reaction processor does not need to take care about the equivalent sites, and can simply continue to generate reaction of hydrogen abstraction reactions as shown in Fig. 5. The output buffering mechanism automatically detects the same reaction generated, and avoids adding it as a new reaction, but, instead, it multiplies the pre-exponential factors of the existing reaction. This mechanism is also efficient for counting the number of equivalent sites correctly.

The reaction generator has a mechanism to control the generation of the reactions over several reaction sites. Figure 6 shows an example of the reaction priority



```
type: [Habs]  priority limit: 30
  site: [Prim]      priority: 100
  site: [Sec]       priority: 110
  site: [Tert]      priority: 120
  site: [CH2-C|O]   priority: 122
  site: [CH-C2|O]   priority: 132
```

Fig. 6 Definition of reaction priority.

definition for H-abstraction reactions from hydrocarbons. For fuels, H-atom abstraction reactions from all the registered sites are generated irrespective of the priority. For the intermediate species, for example aldehydes and ketones formed during the oxidation of fuels, generating the reactions for all the sites may make the mechanism unnecessarily huge. Thus, for this type of reactants, the reaction site lists contains the information on the priority as shown in Fig. 6. For non-fuel species, the behavior of automatic generation is determined by the priority and may be changed by users without modifying the source codes. The reaction generator searches the site with maximum priority in the reactant, and generates the reactions for the sites with priority equal to or higher than (maximum priority) – (priority limit). For example for the priority definition shown in Fig. 6, when the highest priority site is [CH-C2|O] with priority of 132, reactions are generated for the sites with priority equal to or higher than 102 (= 132 – 30). For abstraction reactions, the reagents, for example, OH, H, O, HO_2 etc., can also be controlled: for fuels, full reactions for the all the reagents are generated but reactions with a limited number of reagents are generated for non-fuel species.

At the end of the reaction generation for one species, the system checks whether at least one reaction has been generated for the species, that is, it has not been left as dead-end. Current version of the software can generate oxidation mechanisms for non-cyclic and mono-cyclic alkanes and non-cyclic alkenes (mono-ens) and alcohols, as well as any mixture consisting of any number of supported fuel compounds.

Mixed-Fuel Input and Cross Reactions

The oxidation mechanisms for fuel blends are often constructed by merging the independently developed oxidation mechanism for each fuel. However, this procedure requires enormous efforts for identifying the same chemical species with different names, eliminating duplicated reactions, and adding cross reactions. The "combust" program in KUCRS accepts multiple-fuel input and eliminates such difficulties. By processing multiple fuels simultaneously, the same chemical species are identified properly and generation of the duplicated reactions is avoided. It also automatically generates the cross reactions important in the low-temperature oxidation, which include;

$$R + R'O_2 \rightarrow RO + R'O, \qquad (1)$$

$$RO_2 + R'O_2 \rightarrow RO + R'O + O_2, \qquad (2)$$

$$RO_2 + R'H \rightarrow ROOH + R', \qquad (3)$$

where R and R' denote radicals originated from different fuels. They are alkyl or substituted alkyl radicals, for example, alkyl radicals formed from the H-atom abstraction from alkanes or hydroxyalkyl radicals formed by the OH addition to alkenes and H-atom abstraction from alcohols. In order to avoid making mechanism unnecessarily huge, the system remembers the R's and RO_2's originated directly from the fuels and these reactions were generated only for R's and RO_2's directly originated from fuels, since these types of the reactions for the R and RO_2 generated from the degraded species are usually not important except for the relatively long-lived CH_3 and CH_3O_2 radicals, for which, the reactions are always generated.

MECHANISM VALIDATION

One of the most fuel-sensitive characteristics of combustion is the autoignition phenomena of fuel-air mixtures. The automatic mechanism generation will be useful to predict and analyze the fuel-specific behaviors of the spontaneous ignition, for example, those in the HCCI combustion and engine knock. For this reason, the mechanism validations were made for (1) the autoignition behaviors of typical fuel components and (2) systematic trend of autoignition of various alkanes based on the well known anti-knock properties, research octane numbers (RON). The mechanisms were generated by using the built-in default rate rules and base mechanism implemented in the latest release, KUCRS-20110107m5 [22].

Autoignition of Primary Reference Fuels

For *n*-heptane (C_7H_{16}) and iso-octane (C_8H_{18}), which are the primary reference fuels for octane rating, model simulations using mechanisms generated by KUCRS are compared with the shock-tube ignition delay times [23] in Fig. 7. The mechanisms contained 1071 reactions and 373 species for *n*-heptane, and 1469 reactions and 537 species for iso-octane. The model simulations well reproduced the experiments although the ignition delay times for iso-octane at lower temperatures are slightly overestimated.

Autoignition Critical Compression Ratio and RON

Although the chemical-structure dependence of the octane numbers of pure hydrocarbons have been known from more than 60 years ago [24], it has not yet been explained by systematically generated detail kinetic mechanisms. So, it is interesting to investigate to what extent the auto-generated mechanisms can predict the trend of RON. Although the actual engine knock phenomena involve complex physical and chemical processes, simplified zero-dimensional critical compression ratios for the autoignition of ϕ = 1 homogeneous alkane-air mixtures were calculated in this study and the results are shown in Fig. 8. The abscissa

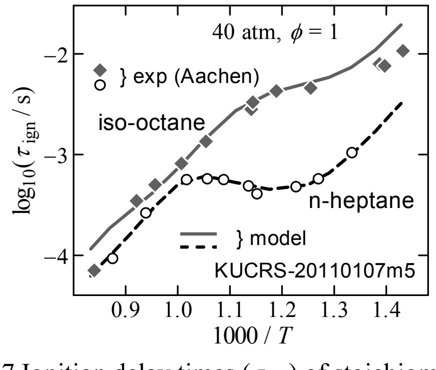


Fig. 7 Ignition delay times (τ_{ign}) of stoichiometric *n*-heptane/air and iso-octane/air mixtures.

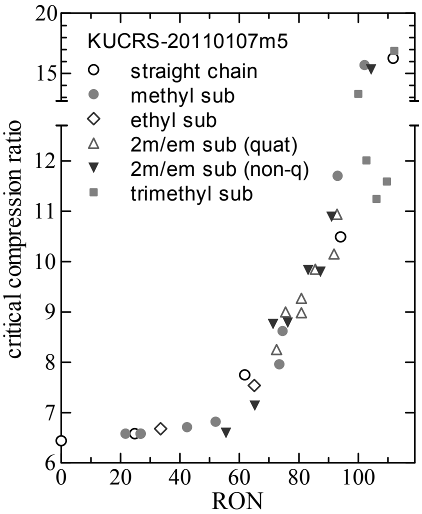


Fig. 8 Correlation between critical compression ratio predicted by kinetic mechanisms and RON of alkanes. 2m/em sub = dimethyl or ethyl-methyl substituted, quat = with quaternary carbon, non-q = no quaternary carbon.

is the experimental RON of pure alkanes [24] and the ordinate is the calculated critical compression ratio for homogeneous autoignition. The initial temperature was assumed to be 373 K at –180 ATDC and the engine speed was 600 rpm. The air composition was assumed to be 78.2% N_2, 20.7% O_2 and 1.1% H_2O. The calculations were made for 36 non-cyclic alkanes. The good correlation between the simulated critical compression ratio and RON suggests the fruitfulness of the rule-based auto-generation technology for the kinetic mechanisms for hydrocarbons.

CONCLUSIONS

A software framework for the construction of chemical kinetic models has been developed which is comprised of advanced features to reduce the efforts for construction, improvement and maintenance of large kinetic models. It can generate kinetic models for alkanes, cyclic alkanes, alkenes and alcohols and the rule-based chemistry is shown to be able to predict systematic trend of autoignition properties of alkanes.

Acknowledgements

The author thanks beta-testers of the KUCRS system, especially Dr. Ellen Meeks (Reaction Design), for useful comments and suggestions for improvement. The author appreciates continuous encouragement by Prof. Mitsuo Koshi (University of Tokyo) and Prof. Hiromitsu Ando (University of Fukui) for the development of this system.

REFERENCES

[1] Curran, H.J., Gaffuri, P., Pitz, W.J., and Westbrook, C.K., Combust. Flame 114: 149–177 (1998).

[2] Curran, H.J., Gaffuri, P., Pitz, W.J., and Westbrook, C.K., Combust. Flame 129: 253–280 (2002).

[3] Dente, M. and Ranzi, E., Comput. Chem. Engng. 3: 61–75 (1979).

[4] Chinnick, S.J., Baulch, D.L., and Ayscough, P.B., Chemomet. and Intell. Lab. Syst. 5: 39–52 (1988).

[5] Hillewaert, L.P., Dierickx, J.L., and Froment, G.F., AIChE Journal 34: 17–24 (1988).

[6] Chevalier, C., Pitz, W.J., Warnatz, J., Westbrook, C.K., and Melenk, H., Proc. Combust. Inst. 24: 93–101 (1992).

[7] Quam, R.J. and Jaffe, S.B., Ind. Eng. Chem. Res. 31: 2483–2497 (1992).

[8] Di Maio, F.P. and Lignola, P.G., Chem. Eng. Sci. 47: 2713–2718 (1992).

[9] Broadbelt, L.J., Stark, S.M., and Klein, M.T., Ind. Eng. Chem. Res. 33: 790–799 (1994).

[10] Blurock, E.S., J. Chem. Inf. Comput. Sci. 35: 607–616 (1995).

[11] Ranzi, E., Faravelli, T., Gaffuri, P., and Sogaro, A., Combust. Flame 102: 179–192 (1995).

[12] Prickett, S.E. and Mavrovouniotis, M.L., Comput. Chem. Engng 21: 1325–1337 (1997); Mavrovouniotis, M.L. and Prickett, S.E., Knowledge-Based Syst. 10: 199–211 (1998).

[13] Susnow, R.G., Dean, A.M., Green, Jr., W.H., Peczak, P., and Broadbelt, L.J., J. Phys. Chem. A 101: 3731–3740 (1997); Matheu, D.M., Dean, A.M., Grenda, J.M., and Green, Jr., W.H., J. Phys. Chem. A: 107: 8552–8565 (2003); Van Geem, K.M., Reyniers, M.-F., Marin, G.B., Song, J., Green, W.H., and Matheu, D.M., AIChE Journal 52: 718–730 (2006).

[14] Warth, V., Battin-Leclerc, F., Fournet, R., Glaude, P.A., Côme, G.M., and Scacchi, G., Comput. Chem. 24: 541–560 (2000); Buda, F., Heyberger, B., Fournet, R., Glaude, P.-A., Warth, V., and Battin-Leclerc, F., Energy & Fuels 20: 1450–1459 (2006).

[15] Weininger, D., J. Chem. Inf. Comput. Sci. 28: 31–36 (1988); Weininger, D., Weininger, A., and Weininger, J. L., J. Chem. Inf. Comput. Sci. 29: 97–101 (1989); Weininger, D., J. Chem. Inf. Comput. Sci. 30: 237–243 (1990).

[16] Kee, R.J., Rupley, F.M., and Miller, J.A., Sandia Report SAND89-8009B (1995).

[17] Miyoshi, A., J. Phys. Chem. A 115: 3301–3325 (2011).

[18] Ritter, E.R. and Bozzelli, J.W., Int. J. Chem. Kinet. 23: 767–778 (1991).

[19] Benson, S.W., Thermochemical Kinetics (2nd ed), John Wiley and Sons, New York, USA, 1976.

[20] Poling, B.E., Prausnitz, J.M., and O'connell, J.P., The Properties of Gases and Liquids (5th ed), McGraw-Hill, Boston, USA, 2001.

[21] Hughes, K.J., Turányi, T., Clague, A.R., and Piling, M.J., Int. J. Chem. Kinet. 33: 513–538 (2001).

[22] Miyoshi, A., KUCRS software library, revision 20110107m5, available at http://www.frad.t.u-tokyo.ac.jp /~miyoshi/KUCRS/

[23] Fieweger, K., Blumenthal, R., and Adomeit, G., Combust. Flame 109: 599–619 (1997).

[24] Lovell, W.G., Ind. Eng. Chem. 40: 2388–2438 (1948).