DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Report for Master Interdisciplinary Project

# Genetic algorithm for Mechanism reduction

**Weixuan Yuan**

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Report for Master Interdisciplinary Project

# Genetic algorithm for Mechanism reduction

# Mechanismusreduktion mit Genetischer Algorithmus

Author: Weixuan Yuan
Supervisor: Supervisor
Advisor: Advisor
Submission Date: 15.03.2023

# Contents

# 1 Introduction

Copied from abstract.

The mechanisms of combustion play a crucial role in industry. This study presents a technique utilizing a genetic algorithm for the reduction of species within a reaction mechanism for industrial applications. Sensitivity analysis is used for data preprocessing and population initialization to enhance the convergence rate of the efficient genetic algorithm. The methodology maintains accuracy by minimizing disagreement between simulated ignition delay times and experimental data. In experimentation utilizing CH4, our method demonstrates a significant reduction in the number of species while preserving accuracy.

Copied from abstract.

# 2 Theoretical background

## 2.1 The Genetic Algorithm

The genetic algorithm (GA) is a widely adopted heuristic search method in evolutionary systems [1]. Incorporating the concept of evolution as a basis for its methodology, GA treats the feasible solutions within the search space as individuals and the optimization objective as the environment. Through the simulation of natural selection within a population comprising a defined number of individuals, GA endeavors to identify the optimal solution. As a heuristic search method, GA has demonstrated efficacy in identifying superior solutions where conventional methods are impractical, for example, reducing mechanisms. In this section, The fundamental framework of GA in general instances is reviewed.

### 2.1.1 Basic Framework

The fundamental structure of GA is visualized in Figure 2.1, comprising the iterative cycle of evaluation, selection, crossover, and mutation. In each iteration, the evaluation and selection processes facilitate the evolution of the population towards the optimization objective, while crossover and mutation introduce diversity to the gene pool, thus enabling the exploration of multiple possible solutions.

```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

Figure 2.1: GA is a loop of evaluation, selection, crossover and mutation.

The subsequent sections offer a comprehensive review of each phase of the GA process, along with implementation examples relevant to general scenarios.

### 2.1.2 Encoding

Encoding is a critical step in establishing a bijective mapping between the search space and the gene space, which is commonly represented as a vector space. This process mimics the expression of genes observed in biological systems, allowing solutions in the search space to be represented as a set of genes. Encoding plays an essential role in facilitating the exchange of information between the evaluation and genetic operators, as shown in Figure 2.2. During each iteration, evaluation and selection processes act on the original solutions, while crossover and mutation operate on the corresponding genes of the solution.

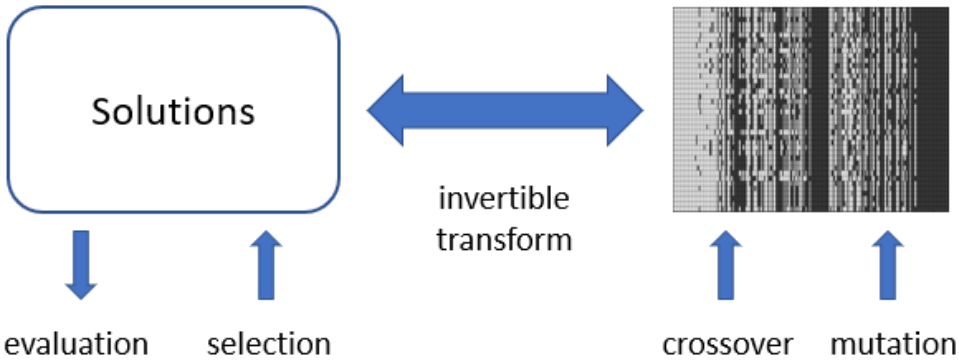


Figure 2.2: The gene pool of a population is a binary matrix, depicted in the image on the right. The matrix is comprised of binary digits, where the presence of a light pixel indicates the value of 1 and a dark pixel corresponds to a value of 0.

Various encoding methods have been proposed, including integer encoding, binary encoding, real-valued encoding, permutation encoding, and more. The choice of encoding method should be based on the specific characteristics of the problem at hand. In this work, the binary encoding scheme is employed.

The encoding (e.g. chromosome) of each solution into a binary sequence of length $L$ results in the representation of the population as a binary matrix $X \in \{0,1\}^{N\times L}$. The choice of the encoding method is dependent on the specific problem being addressed, as various methods have been developed to represent different types of solutions. The specific encoding methodology employed in this work will be described in greater detail in the subsequent chapters.

### 2.1.3 Initialization

The initialization of the first population, $X_0$, establishes the starting point for the convergence of GA. Randomly sampling $X_0$ is a commonly used approach, owing to its ease of implementation and the broad range of initial diversity it provides. However, in cases where prior knowledge is available, initializing with high-quality solutions, that are closer to the optimal, improves the convergence rate. Nevertheless, such an approach also runs the risk of failing

to explore other regions of the search space where potential optimal solutions may reside. To balance the trade-off between convergence speed and gene diversity, a combination of both approaches is often employed in practice. Figure 2.3 established an example for each initializing strategy.

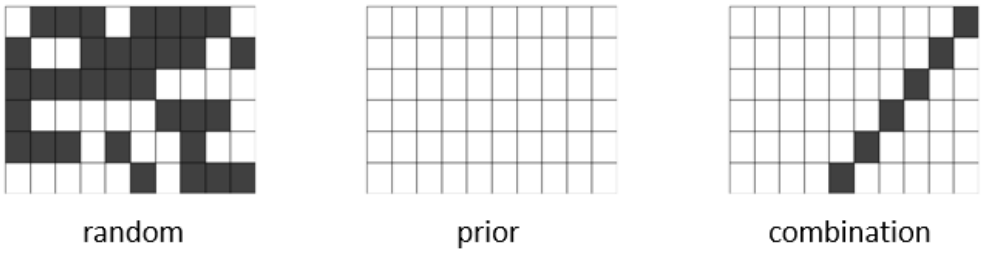


Figure 2.3: Different initialization strategies leads to distinct start points for GA. In this example, the gene for high-quality solution is assumed to be all ones.

### 2.1.4 Evaluation

The assessment of population quality is a fundamental aspect of GA, and is achieved through the use of a fitness function, denoted as $F$, that assigns a fitness value to each individual in the population. The fitness value reflects how effectively the individual solution addresses the problem at hand, and the optimization process of GA can be likened to a game in which the goal is to identify a solution that maximizes the fitness value:

$$x = \arg\max_{x \in \{0,1\}^L} F(x). \tag{2.1}$$

It is important to note that the fitness function in GA operates on the solutions, rather than on the encoded gene representation. However, to maintain clarity, the process of transforming the encoding into the solution is typically omitted in the formula for the fitness function. (Alternatively, the task of decoding is incorporated into the fitness function to streamline the optimization process.) The design of fitness function is problem-specific and the design for the task in this work will be introduced in later sections.

### 2.1.5 Selection

To mimic the natural selection process in evolution, selection in GA determines which individuals in the current population will be used to create the next generation. Naturally, individuals with higher fitness values are more likely to be selected.

There exist many commonly used selection methods in GA, including roulette wheel selection, rank-based selection, tournament selection, among others. In this work, tournament selection has been employed due to its ability to promote large gene diversity within the population.

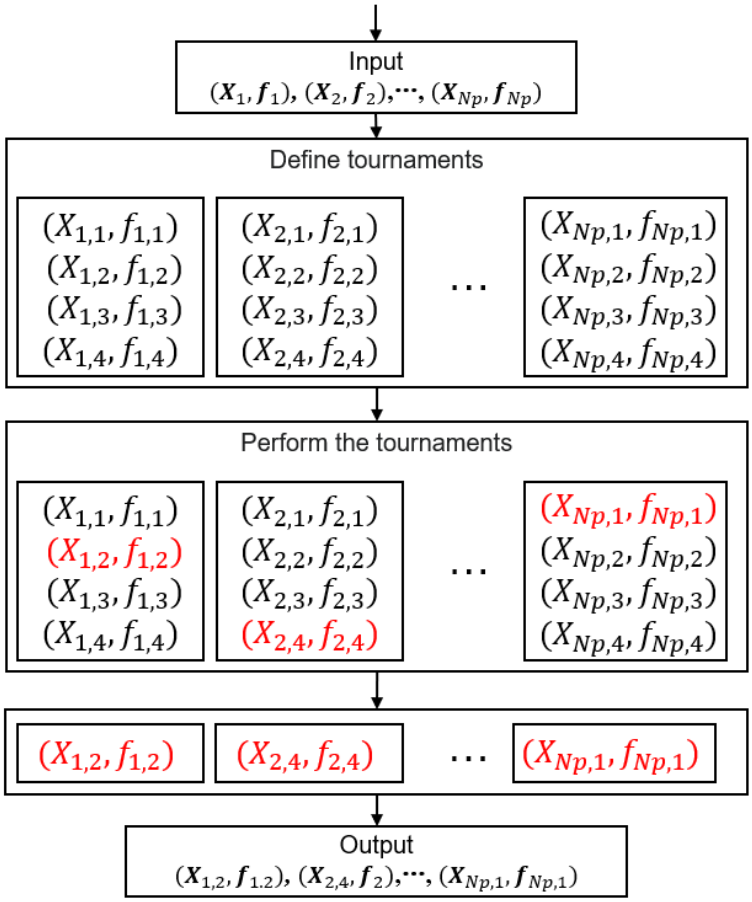


Figure 2.4: The tournament selection. $Np$ is the population size,

As illustrated in Figure 2.4, the input to tournament selection in GA is a sequence $(X_i, f_i)$, representing the chromosomes and corresponding fitness values. To proceed with tournament selection, $Np$ tournaments are defined, each consisting of four chromosomes randomly selected from $(X_i, f_i)$ with equal probability as participants. After the tournament stage, the chromosome with the highest fitness rank is declared the winner, and advances to the selection stage of the optimization process. It is important to note that each tournament produces exactly one winner, ensuring that the population size is remained consistent by selecting $Np$ to be equal to the population size.

### 2.1.6 Crossover

The crossover operation in GA is inspired by the biological process of genetic recombination, where homologous chromosomes of parents exchange segments of genetic material, resulting in offspring with a combination of genetic information from each parent. The practical implementation of crossover in GA involves randomly pairing up parent solutions selected from the previous generation, with each pair producing one or more offspring chromosomes.

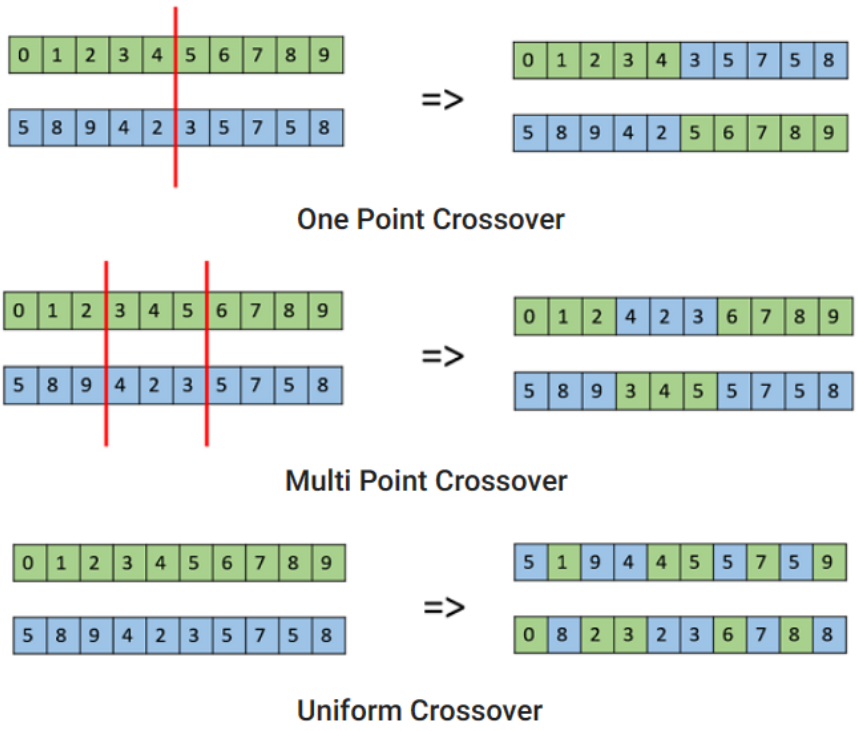


Figure 2.5: Source: Crossover examples.

Figure 2.5 illustrates three common ways of implementing crossover. Single-point crossover randomly selects a point along the length of the parent chromosomes and the genes beyond that point are exchanged between the two parent solutions to create two offspring solutions. Two-point crossover is implemented similarly, but two crossover points are selected instead of one, and the genes between the two points is exchanged between the two parent solutions. Uniform crossover is a method of crossover in which genetic material is selected from each parent with a predefined probability to produce offspring chromosomes. This approach is more diverse compared to single-point or two-point crossover, as it allows for a greater mixing of genetic material between the parent solutions.

### 2.1.7 Mutation

While both crossover and mutation introduce diversity, they do so in different ways. Crossover creates new genetic combinations that may be more effective than either parent solution, while mutation generates completely new genetic material that may not exist in the previous population.

The operation of mutation depends on the encoding scheme used for the solutions. For example, in problems with real-valued encoding, the mutation operation may involve adding a small amount of Gaussian noise to one or more genes on the chromosome. On the other hand, in problems with binary encoding, mutation could be implemented by randomly

flipping a small number of bits on the chromosome as visualized in Figure 2.6.

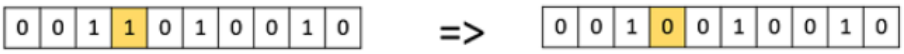


Figure 2.6: Source: Bit-flip mutation.

## 2.2 Disagreement measure

Disagreement between the simulated and experimental IDT is used to determine the accuracy of the reduced mechanism in this work. This chapter introduces the method for calculating simulated ignition delay time (IDT) and defining the fitness value to evaluate the accuracy of a reduced mechanism.

### 2.2.1 Ignition Delay Time

The ignition delay time (IDT) is a critical measure of the reactivity of fuels, defining the time taken by a specific fuel mixture to oxidize and release heat under particular thermodynamic conditions of pressure and temperature [2]. In the Chemkin software [3], IDT is defined in two ways: (1) as the peak in the hydroxyl radical (OH), and (2) as the point of maximum change in temperature over time.

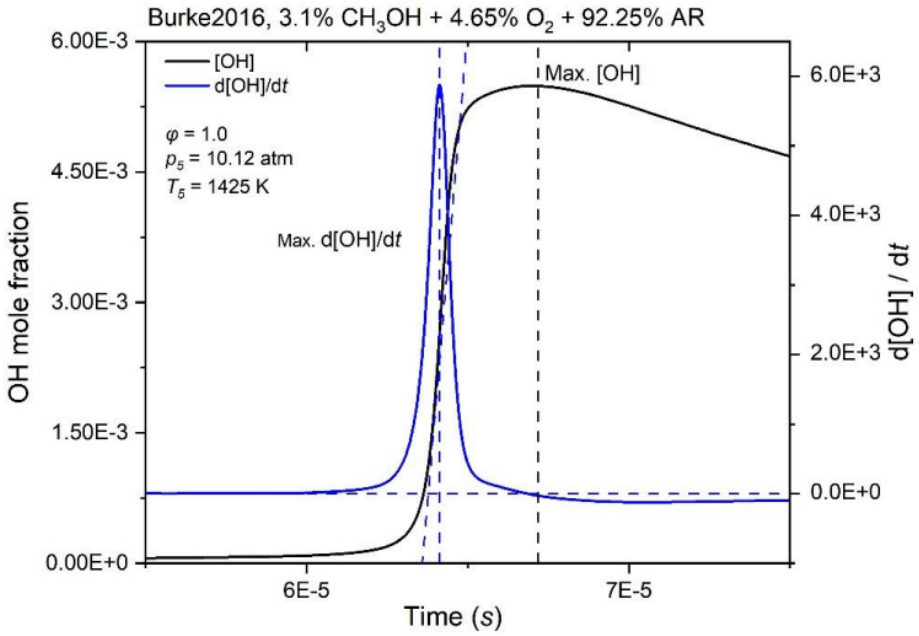


Figure 2.7: IDT. Source: [4]

Figure 2.7 depicts the former definition with a vertical dotted line at the peak of the OH curve. Alternatively, IDT can also be defined by the intersection of the tangent and the abscissa, which occurs where the blue (d[OH]/dt) curve shows the steepest slope, represented by the diagonal dotted line.

### 2.2.2 Computation Approaches

In this study, the hydroxyl radical and temperature curves were simulated using Cantera [5], an open-source software, through its Python interface, which is integrated into the program. The simulated IDT is calculated using the same method as the experimental data, as described in Section 2.2.1, and the resulting IDT values are used to compute the disagreement between the simulation and experimental data. The disagreement under one single thermodynamic condition ($c$) is defined as followed:

$$D_c = \frac{(log(IDT_c^{Exp}) - log(IDT_c^{Sim}))^2}{(\frac{log(1+w)-log(1-w)}{2})^2}, \tag{2.2}$$

where $IDT_c^{Exp}$ and $IDT_c^{Sim}$ represent the experimental and simulated IDT, respectively. The denominator includes a weighting factor for uncertainty ($w$).

The overall disagreement is calculated as the sum of disagreement values under all thermodynamic conditions tested:

$$D = \sum_{c \in C} D_c, \tag{2.3}$$

where C is the set of thermodynamic conditions.

## 2.3 Sensitivity analysis

# 3 Experiment

This report presents the Combustion Mechanism Reduction Method, which was developed to reduce the number of species in reaction mechanisms while maintaining accuracy. Experiments were conducted on CH4 and C6H12, demonstrating that the method is effective in reducing the number of species while maintaining the disagreement level. This chapter provides a detailed description of the method, as well as its theoretical underpinnings and implementation strategies.

## 3.1 Dataset

To reduce the mechanism, the detailed mechanism and experimental data need to be provided by users in the form of input files.

### 3.1.1 Experimental IDT

| i.d. | T5 | P5 | O2 | cyC6H12 | AR | Expidt | ExpIDT | EXPun | Condition |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 8 | 4 | 0 | 8 | 4 | 8 | 4 | 239022 |
| 2 | 1 | 8 | 9 | 1 | 8 | 9 | 8 | 4 | 236147 |
| 3 | 2 | 8 | 1 | 2 | 8 | 1 | 8 | 4 | **229949** |
| 4 | 0 | 8 | 9 | 0 | 8 | 9 | 8 | 4 | 233344 |
| 5 | 1 | 5 | 9 | 1 | 5 | 9 | 8 | 4 | 245426 |

Table 3.1: Hyperparameter search results. $N_3$ is the number of blocks in the encoder. $N_2$ is the number of preserving cells in each block.

In this study, the quality of the reduced mechanism is evaluated based on experimental data, for example, obtained from real-world combustion settings. In the example presented in Table 3.1, the experimental data corresponds to ignition delay times (IDT) conducted under various operating conditions.

The first column of the form lists the serial number of each sample. The second and third columns record the initial temperature and pressure for each operating condition, respectively. The subsequent columns document the initial specie proportions. For instance, the first row indicates that the initial gas composition comprises 90% rare gas AR, 5% fuel C6H12, and 5% oxygen. Expidt and Exprun represent the experimentally measured ignition delay time (IDT) under the corresponding conditions and the maximum simulation time in Cantera simulation,

respectively. The last two columns in the form correspond to the confidence levels and IDT calculation method for each dataset.

### 3.1.2 Detailed Mechanism

The detailed mechanisms can be provided through a YAML file. Or alternatively through CHEMKIN files, which consist of an INP file with species and reactions and a DAT file with associated thermodynamic data.

## 3.2 Method

This work aims to reduce the number of species in combustion models through the use of genetic algorithms and sensitivity analysis. This chapter provides a detailed explanation of the principles behind the proposed method.

### 3.2.1 Basis Framework

Our method, as shown in Figure 3.1, starts by performing sensitivity analysis on the raw mechanism to produce an input mechanism for the genetic algorithm. The input mechanism is then simplified using the genetic algorithm to reduce the number of species while maintaining the accuracy of the combustion model.

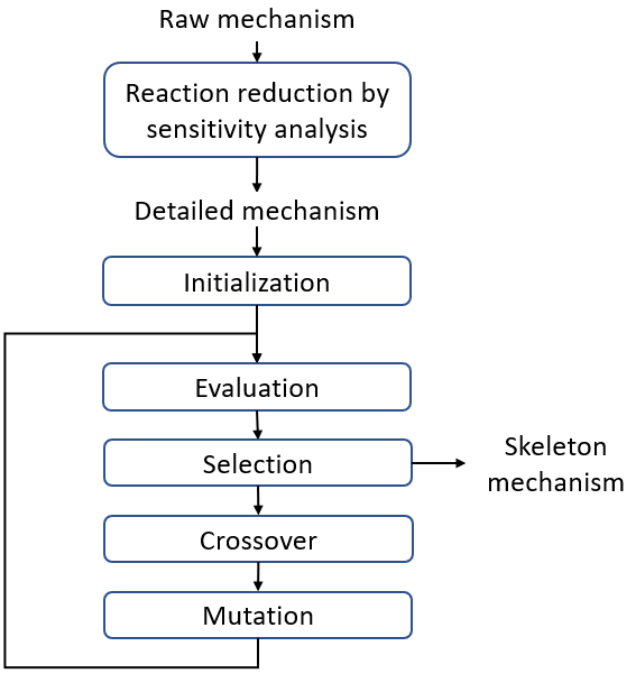


Figure 3.1: GA is a loop of evaluation, selection, crossover and mutation. We do initialization with respect to the sensitivity analysis results.

### 3.2.2 Preprocessing with Sensitivity Analysis

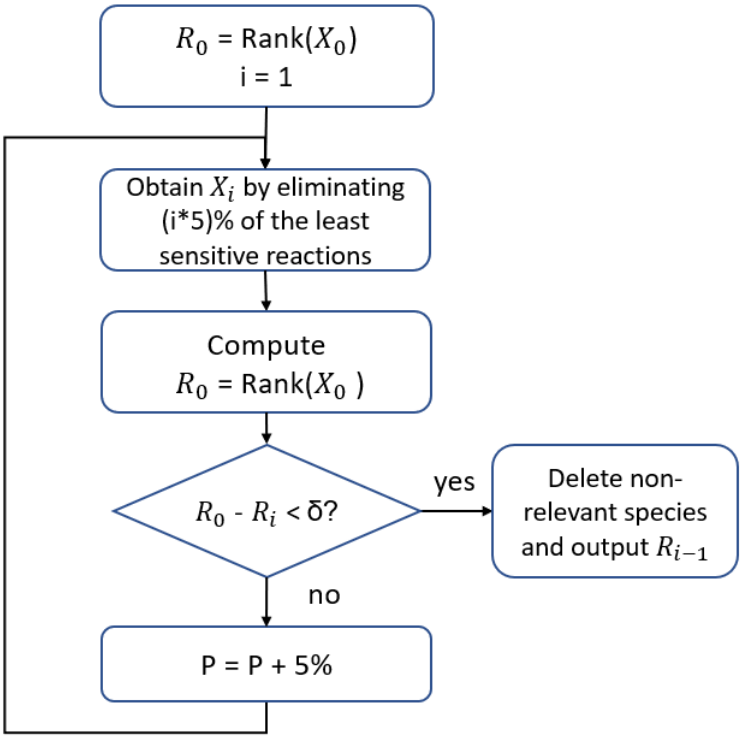


Figure 3.2: Reaction reduction through sensitivity analysis, where $\delta$ is a user defined hyper-parameter.

To accelerate the genetic algorithm, the number of reactions can be reduced in advance through sensitivity analysis. As depicted in Figure 3.2, each reaction in the detailed mechanism is assessed in terms of its sensitivity with respect to OH, for different reaction cases, using Cantera. Based on this, the least important reactions are progressively removed in a step-wise manner, in the reverse order of sensitivity, as long as the resultant fitness of the mechanism remains within an acceptable range. Note that "Rank" refers to the fitness value introduced in Section 2.2.

### 3.2.3 Genetic Algorithm

Following the principles of genetic algorithm introduced in Section 2.1, appropriate encoding, initialization approaches, and genetic operators are selected and applied.

#### Encoding

The binary encoding method is selected to represent which species are kept or eliminated, defined as $X \in 0, 1^L$, where $L$ is the number of species in the original mechanism. Here,

$x_i = 1$ denotes that the $i$-th species is kept, and 0 denotes that it is eliminated. Furthermore, reactions involved in eliminated species are also deleted. Using this encoding, the size of the mechanism can be easily computed by $sum(X)$.

### Initialization

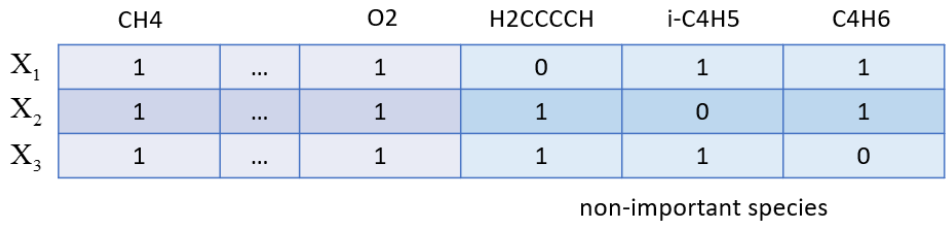


Figure 3.3: First generation.

For the first generation of chromosomes, we eliminate a user-defined number $k$ out of $n$ non-essential species (that are also user-defined) from the input mechanism, resulting in $\binom{k}{n}$ possible unique chromosomes, assuming no duplicates. This process is illustrated in Figure 3.3, where one non-essential species is removed from a set of three.

### Evaluation

During the evaluation process, each chromosome's fitness value is computed based on the normalized disagreement described in Section 2.2 and the normalized size:

$$Fitness = -\frac{D(\mathbf{decode}(\mathbf{X}))}{D(X^0)} - \beta \cdot \frac{sum(X)}{sum(X^0)}, \tag{3.1}$$

where $X^0$ is the detailed mechanism and $\beta$ is a hyper-parameter that controls the weight of two components. In this work, $\beta$ is set to 3.0.

### Selection

The tournament selection method discussed in Section 2.1.5 is then applied to optimize the generation, while keeping the number of chromosomes constant. The reason for choosing the tournament selection over other methods is that it promotes a large diversity of solutions.[cite]

### Crossover

Uniform crossover, as discussed in Section 2.1.6, is applied where each gene of both parents is assigned to both children with equal probability. This method is preferred for large search spaces due to its ability to introduce a high level of diversity [6].

**Mutation**

One-directional mutation is applied, with each bit having a chance to independently flip to zero, but zeros will not flip to ones. It is important to note that bits representing predetermined, important species will not mutate and will remain as ones. Also, user can define a certain number of important-species that can never be deleted. Bits corresponding to such species will not mutate and always stay 1s.

## 3.3 Numeric Implementation

The proposed method is implemented in Python using numerical computations. The program is divided into three parts: the combustion simulation component that utilizes the Cantera interface, the genetic algorithm section, and the control unit that serves as an interface between the two, storing and transforming information. The architecture of the program is illustrated in Figure 3.4.

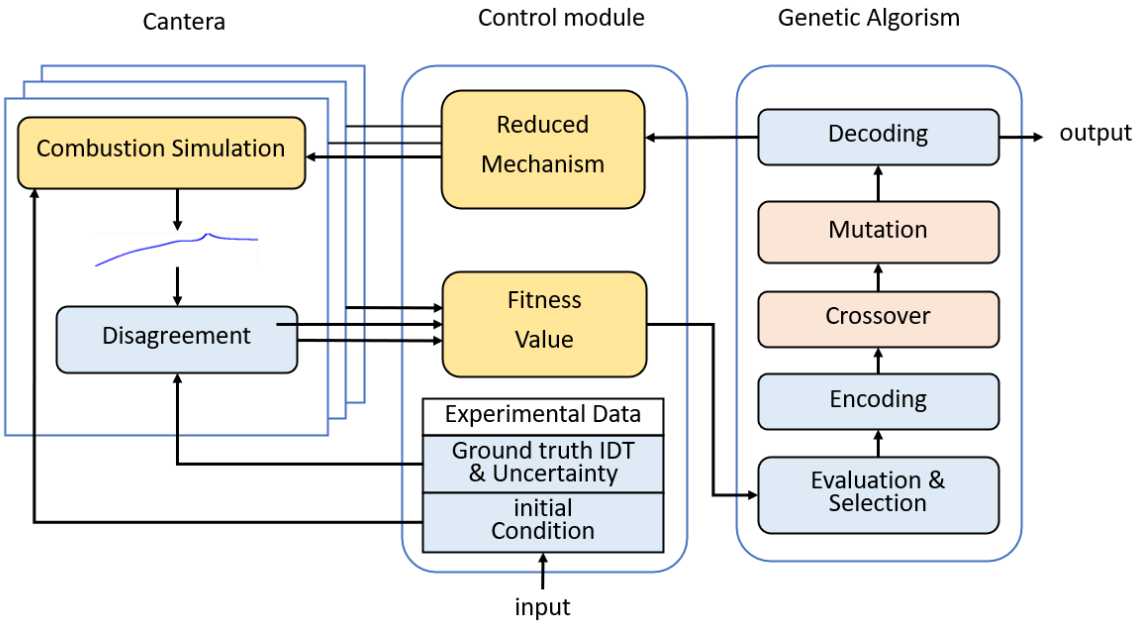


Figure 3.4: The architecture of the program. The data thread is represented by the counter-clockwise circular arrow.

To reduce the mechanism, the program first takes the detailed mechanism and experimental data as input, preprocesses them (with sensitivity analysis), and saves them in the control unit. The genetic algorithm is then applied iteratively along the data pipeline, with evaluation, selection, crossover, and mutation performed in sequence. During evaluation, the combustion simulation module is invoked for each combustion condition, which is accomplished via multi-processing for efficient computation. After all processes have joined, the fitness value is

computed using the methodology described in Section 2.2.

# 4 Results

In this Section, we show the current results for methods that has been implemented.

## 4.1 Diagrams

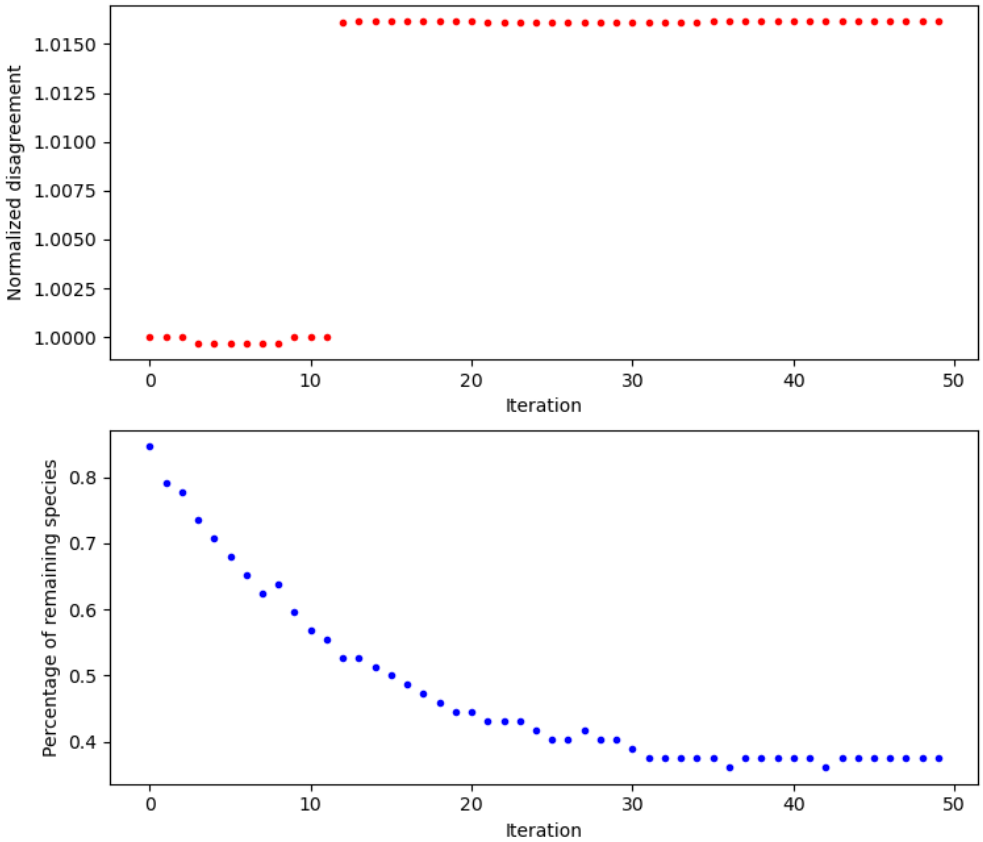


Figure 4.1: Results for CH4 mechanism reduction.

We conducted experiments on CH4 and C6H12 to demonstrate the effectiveness of our proposed reduction method. The results of the CH4 experiment, including the model size curve (on top) and the effect of omitted species on the model accuracy (below), are presented in Figure 4.1. Notably, the solution with the highest fitness value in each population is used to represent the corresponding iteration. The model size curve shows the percentage

of remaining species at each iteration of the GA, with 15.3% of species eliminated during preprocessing and an additional 47.2% eliminated during the GA, resulting in a final model with 37.5% of the original species. The effect of omitted species on the model accuracy is shown to be negligible, with the error normalized by that of the detailed mechanism remaining stable throughout the iterations. The disagreement increased by approximately 1.5% in the output reduced mechanism.

## 4.2 Visualization

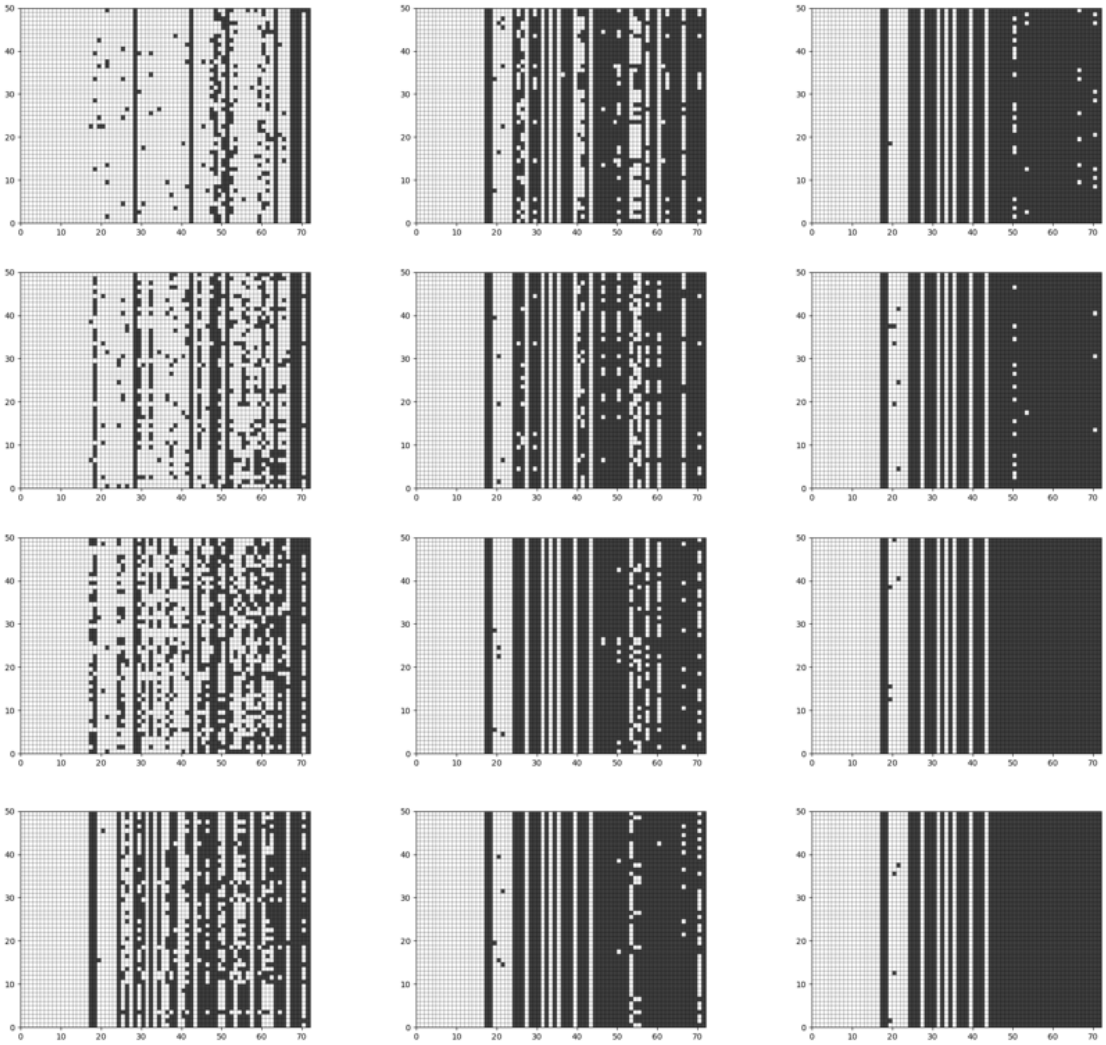

Figure 4.2: Visualization for CH4 mechanism reduction.

In Figure 4.2, the gene pool code changes are visualized as the genetic algorithm progresses. The images are arranged chronologically from left to right, top to bottom, with intervals of

generally 4 iterations but not uniformly. The top left image shows the gene pool code after the first iteration, and the image in the second column of the top row corresponds to the fifth iteration. The bottom right corner of the figure shows the gene pool code after the 49th iteration, which is the last one.

Each sub graph consists of 72 pixels, which are either light or dark, representing 1 or 0, respectively. The rows represent the genetic code of an individual in the population, and there are 40 rows for a population size of 40. As the genetic algorithm progresses, large areas of light pixels turn dark, indicating that unimportant species are gradually eliminated. The final few sub-graphs show stable columns, suggesting the algorithm has converged.

# 5 Conclusion and Outlook

In this study, we propose and implement a genetic algorithm-based reduction method for chemical combustion mechanisms. Our experiments on CH4 and C6H12 show that the proposed method successfully reduces the size of the model while maintaining its accuracy. Additionally, we utilize sensitivity analysis and multi-processing to significantly speed up computation. In future research, we plan to explore additional measuring metrics, such as PFR and flame speed, and support more types of react containers. We also aim to create a more user-friendly interface, which may require rewriting the project in C++ to maintain consistency with Cantera.

# Bibliography


[1] M. D. Vose. *The simple genetic algorithm: foundations and theory*. MIT press, 1999.

[2] D. F. Davidson, Y. Zhu, J. Shao, and R. Hanson. "Ignition delay time correlations for distillate fuels". In: *Fuel* 187 (2017), pp. 26–32.

[3] R. J. Kee, F. M. Rupley, E. Meeks, and J. A. Miller. *CHEMKIN-III: A FORTRAN chemical kinetics package for the analysis of gas-phase chemical and plasma kinetics*. Tech. rep. Sandia National Lab.(SNL-CA), Livermore, CA (United States), 1996.

[4] N. M. Marinov. "A detailed chemical kinetic model for high temperature ethanol oxidation". In: *International Journal of Chemical Kinetics* 31.3 (1999), pp. 183–220.

[5] D. G. Goodwin, H. K. Moffat, I. Schoegl, R. L. Speth, and B. W. Weber. *Cantera: An Object-oriented Software Toolkit for Chemical Kinetics, Thermodynamics, and Transport Processes*. `https://www.cantera.org`. Version 2.6.0. 2022. DOI: `10.5281/zenodo.6387882`.

[6] W. M. Spears and K. D. De Jong. *On the virtues of parameterized uniform crossover*. Tech. rep. Naval Research Lab Washington DC, 1995.