Efficient Clustering-Based Genetic Algorithms in Chemical Kinetic Modelling

Lionel Elliott[1], Derek B. Ingham[1], Adrian G. Kyne[2], Nicolae S. Mera[2], Mohamed Pourkashanian[3], and Sean Whittaker[1]

[1] Department of Applied Mathematics, University of Leeds, Leeds, LS2 9JT, UK
{lionel, amt6dbi}@amsta.leeds.ac.uk, che9sw@leeds.ac.uk
[2] Centre for Computational Fluid Dynamics, Energy and Resources Research Institute, University of Leeds, Leeds, LS2 9JT, UK
{fueagk, fuensm}@sun.leeds.ac.uk
[3] Energy and Resources Research Institute, University of Leeds, Leeds, LS2 9JT, UK
fue6lib@sun.leeds.ac.uk

Abstract. Two efficient clustering-based genetic algorithms are developed for the optimisation of reaction rate parameters in chemical kinetic modelling. The genetic algorithms employed are used to determine new reaction rate coefficients for the combustion of four different fuel/air mixtures in a perfectly stirred reactor (PSR). The incorporation of clustering into the genetic algorithm allows for a considerable reduction in the number of computationally expensive fitness evaluations to be realised without any loss in performance. At each generation, the individuals are clustered into several groups and then only the individual that represents the cluster is evaluated using the expensive fitness function. The fitness values of the other individuals in the same cluster are estimated from the representative individual based on a distance measure in a process called *fitness imitation*.

1 Introduction

Many combustion phenomena are kinetically controlled; these may include the formation of pollutants in an exhaust stack, the burning velocity of a premixed flame, or the conversion of NO to NO_2 in a gas turbine combuster. In order to fully understand the chemical processes occurring in such phenomena, it is essential that a detailed chemical kinetic approach be undertaken.

One popular way to model the chemistry of combustion is to use a system of chemical reactions for which the rates of each reaction are known. Reaction rate data for simple fuels, such as hydrogen, is known with a high degree of confidence, see [1], and thus the combustion of hydrogen at temperatures in excess of 1000K may be modeled using a simple eight-step reaction mechanism. However, considerably more complex fuels, such as kerosene, require more than 1000 reaction steps with over 200 species, see [2].

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 932–944, 2004.

Genetic algorithms (GAs) can be used to optimise the reaction rate data of a chemical kinetics problem. A method utilising GAs is proposed in [3] for the rapid extraction of the chemical kinetic rate coefficients for a simplified combustion mechanism from a given set of (detailed) chemical data. In this method the matching of heat release and species production rates of the simplified mechanism to those of an underlying detailed chemical mechanism allows for speedy identification of rate coefficients for a particular operating condition.

In [4], a powerful inversion technique that is based on a binary-coded GA, is developed. Here, new Arrehenius reaction rate coefficients for the combustion of a hydrogen/air mixture in a perfectly stirred reactor are determined. In this technique, the output species profiles obtained from an original set of rate constants are reproduced by a new different set determined using a GA inversion process. A kinetic scheme defined by decoding the genetic data generated by the GA can be used as the basis for running CHEMKIN's Perfectly Stirred Reactor (PSR) computer library, see [5] and [6], and the resulting net species concentration outputs compared to known data. The PSR code can be used to establish the net species concentrations of each of the products based upon different reactor conditions for given estimates to the set of Arrhenius reaction rate coefficients. The overall goal of the inversion process is to determine the unknown reaction rate coefficients which both match the given net species concentrations at different reactor conditions and will, furthermore, correctly predict the net species concentrations at other reactor conditions.

Many engineering or scientific GA-based optimisation problems, involve the use of a simulator or analysis code linked to the GA. It is this code that is used to generate fitness values for the population of possible solutions produced by the GA. One main difficulty in applying GAs to such problems is that a large number of fitness evaluations, i.e. executions of the simulator or analysis code, are usually required before a satisfactory result can be obtained. In general, the computational expense for a single execution of this code is very high compared to the expense for generating and manipulating the population of possible solutions. The main focus of this study is to the incorporation of clustering techniques into the GA in order to speed up the optimisation process without any loss in the algorithms performance. The hybrid clustering-based GAs, are tested on four different fuel/air mixture PSR problems, and the results are compared with a standard GA. The four test problems are a hydrogen/air mixture, a formyl-radical/air mixture, a methane/air mixture, and a kerosene/air mixture. The computational expense increases from the hydrogen/air mixture through to the kerosene/air mixture.

In this study, a floating point version of the GA has replaced the binary-coded GA used previously as they have been shown to be especially suited to numerical optimisation on continuous domains, see [7].

2 The Perfectly Stirred Reactor (PSR) Code

2.1 Introduction

The Perfectly Stirred Reactor (PSR) code is one of three codes available from Sandia National Laboratories that has been used in conjunction with the CHEMKIN II suite of software. The PSR code is a FORTRAN computer program that predicts the steady-state temperature and species composition in a perfectly stirred reactor, whilst the CHEMKIN suite handles the chemical reaction mechanism and the thermodynamic properties.

The stirred reactor consists of a small thermally insulated chamber that has both inlet and outlet ducts. The reactor is characterised by a reactor volume, residence time or mass flow rate, heat loss or a gas temperature, and an inlet temperature and mixing composition. High-intensity turbulent mixing of the steady inlet flow of fuel and oxidiser produces a nearly spatially uniform distribution of contents within the reactor. Because it is assumed that the mixing process is infinitely fast, the conversion of reactants to products is controlled solely by chemical reaction rates.

2.2 Reaction Rate Parameters

In general, Y_k is used to denote the mole fraction of the k^{th} species, where $k = 1,..,K$ and K represents the total number of species. The mole fractions of the k^{th} species at the inlet are denoted by Y^*_k, for $k = 1,..,K$, and the inlet temperature by T^*, whilst the temperature and composition which exit the reactor are assumed to be the same as those in the reactor since the mixing in the reactor chamber is intense.

The net chemical production rate of each species results from a competition between all the chemical reactions involving that species. It is assumed that each reaction proceeds according to the law of mass action and the forward rate coefficients are in modified Arrhenius form

$$k_{f_i} = A_i T^{\beta_i} \exp\left(-\frac{E_i}{RT}\right) \tag{1}$$

for $i = 1,..,N_R$, where T is the temperature, $R = 1.9860$ cal mol^{-1} K^{-1} is the universal gas constant, there are N_R competing reactions occurring simultaneously, and the rate equations (1) contain the three parameters A_i, β_i, and E_i for the i^{th} reaction. A_i is the pre-exponential collision frequency factor, β_i is the non-Arrhenius index, and E_i is the activation energy. It is the incorporation of clustering techniques into the GA for the determination of these parameters for each reaction, based upon outlet species mass fractions alone, which is investigated in this study.

2.3 Problem Formulation

During this study, results obtained using the clustering-based GAs are compared with those obtained from a standard GA for four test problems, namely

(i).	Hydrogen/air	9 species and 19 reactions.
(ii).	Formyl-radical/air	22 species and 67 reactions.
(iii).	Methane/air	24 species and 103 reactions.
(iv).	Kerosene/air	97 species and 460 reactions.

For each of the four test problems, an inverse solution procedure is set up in an attempt to recover the species concentration profiles predicted by a previously validated reaction mechanism over a limited range of reactor conditions. A total of N_s = 11 different PSR conditions are considered for the first three problems, corresponding to various changes to the inlet temperature (T = 1000K,..,2000K, with ΔT = 100K), whilst N_s = 5 different PSR conditions are considered for the kerosene mixture corresponding to T = 1000K, 1040K, 1080K, 1110K, and 1150K.

The inversion process aims to determine the unknown Arrhenius reaction rate coefficients (A_i, β_i, and E_i in (1)) by searching for the set of reaction rates that gives the best fit to a given set of data, i.e. the validated reaction mechanism. A ± 25% variation from the values given in the validated reaction mechanism is used to define the boundaries within which the GA searches for solutions. This small variation in the values of the reaction rate coefficients is sufficient to produce significantly larger variations in the net species concentrations.

A set of output species concentration profiles is obtained by simulation using a previously validated reaction mechanism. If data is simulated for N_s different reactor conditions, and all K species are measured for each condition, then the data will consist of $K{\cdot}N$ species concentration measurements. A set of Arrhenius reaction rate coefficients is chosen which gives a best fit to the species concentration measurements. Such a fitting procedure is accomplished using an optimisation technique that looks for the maximum of the following function using a GA

$$f_{\mathrm{PSR}}\left(\left(A_i,\beta_i,E_i\right)_{i=1,..,N_R}\right)=\left\{10^{-8}+\sum_{j=1}^{N_s}\sum_{k=1}^{K}\left(Y_{jk}^{\mathrm{calc}}-Y_{jk}^{\mathrm{orig}}\right)^2\right\}^{-1} \tag{2}$$

where

- Y_{jk}^{calc} represents the mole concentration of the k^{th} species in the j^{th} set of reactor conditions using the set of Arrhenius rate coefficients.
- Y_{jk}^{orig} represents the corresponding original value for the same mole concentration obtained by simulation using a validated reaction rate mechanism.

3 Optimisation Using the Genetic Algorithm

3.1 The Genetic Algorithm

A floating point number encoded generational GA is used in this study in place of a previously used binary encoded GA as they have been shown to be better suited to numerical optimisation on continuous domains. Previous work incorporating a standard GA into an optimisation procedure for the recovery of reaction rate parameters highlighted the following genetic operators as most appropriate for this study, roulette selection, BLX-α crossover, crossover probability $p_c = 0.6$, and non-uniform mutation, mutation probability $p_m = 0.6$. A value of $\alpha = 0.5$ was chosen for the crossover operator as this strikes an equal compromise between exploration and exploitation of the search domain. The maximum number of generations performed by the GA was set to 1000. The population size and the size of the offspring pool will be discussed in section 4. An elitist strategy is incorporated into the GA with the two fittest individuals from the previous generation surviving into the next generation.

3.2 An Introduction to Fitness Approximation

A large proportion of real-world engineering or scientific GA-based optimisation problems involve the use of a simulator or analysis code to compute the population fitness values. In many cases, each application of the simulator code takes a non-negligible amount of time to return a fitness value, and thus for problems that require a high number of fitness evaluations in order to return a satisfactory solution, the computational expense becomes a major issue.

Several methods exist that involve the use of approximation models to replace some or all of the computationally expensive evaluations. The most popular ones include polynomial modelling whereby the fitness landscape is approximated using a polynomial function; with quadratic approximation functions being favoured, see [8], the Kriging model, see [9], neural networks including multi-layer perceptrons and radial basis function networks, see [10] and Support Vector Machines (SVM), see [11].

There are two major concerns regarding the use of these models for fitness evaluation. First, it should be ensured that the algorithm converges to the global optimum of the original fitness function. Second, the computational cost should be reduced as much as possible. The advantage of such models is that they are in, general, significantly less computationally expensive than the original fitness function. A comprehensive review of fitness approximation techniques in evolutionary computation can be found in [12].

The application of many of the more common approximation models such as those listed in [8] to [11] to problems of a high dimension is not practical as the computational expense of constructing and executing the model is in many cases comparable to the original fitness function. The remaining part of this paper concerns the use of two clustering algorithms each separately embedded into a standard GA, and each

used to form an approximate model from which it is possible to significantly reduce the total number of original fitness function evaluations without loss in performance for four PSR combustion problems of high dimension.

3.3 Fitness Approximation by Clustering

Clustering is an exploratory data analysis method applied to data samples in order to discover structures or certain groupings in a data set. The data samples are grouped so that they exhibit some degree of similarity within each group. These groups are called clusters. There are three general categories of clustering, hierarchical clustering, partitional clustering, and overlapping clustering.

Perhaps the most important aspect of clustering is the choice of similarity measure. Most common clustering methods use a distance measure to assign similarity; these include the city block distance, the Euclidean distance, and the Minkowski distance. In general, the distance d_{ij} between the i^{th} and j^{th} vector data samples $\underline{x}_i$ and $\underline{x}_j$ whose dimension is n is computed as

$$d_{ij} = d\left(\underline{x}_i, \underline{x}_j\right) = \sqrt[m]{\sum_{k=1}^{n} \left| x_{ik} - x_{jk} \right|^m} \tag{3}$$

where

- $m = 1$ City block distance
- $m = 2$ Euclidean distance
- $m = 3$ Minkowski distance

Hierarchical clustering algorithms construct a structure of clusters and provide a view of the data at different levels of granularity. The approach usually takes one of two approaches, the first is the agglomerative approach, and the second is the divisive approach, see [13]. There are several commonly used hierarchical clustering algorithms and these include the single-linkage algorithm, the complete-linkage algorithm, the average-linkage algorithm, and Ward's method.

Partitional clustering algorithms create an unstructured set of clusters whereby the original data set is partitioned into similar groups based on proximity to one another, with the view that samples close together are similar. Each cluster is completely separate from the other clusters and no overlapping occurs. There are several commonly used partitional clustering algorithms and these include the k-means algorithm, the hard c-means algorithm, and Forgy's algorithm, see [13].

Overlapping clustering algorithms create a clustering pattern similar to that of partitional clustering with exception that the clusters are allowed to partially overlap on another. Commonly used examples of this type of clustering include the fuzzy c-means algorithm, and the b-clump algorithm.

In this study the k-means partitional clustering algorithm and Ward's hierarchical method (also known as the minimum-variance method) are each separately incorporated into a standard GA and are used to cluster the population at each and every

generation after initialisation. After clustering, the representative for each cluster is chosen and its fitness is evaluated using the original fitness function (2). The fitnesses of the remaining individuals in the population are then estimated based on the Euclidean distance from their cluster representative.

The *k*-means algorithm clusters the GA population in the following way,

a. At each and every generation after initialisation the GA generates the n_{child} individuals in the usual way with the first n_{cluster} individuals forming the initial cluster sites (this is a random choice due to the individuals being generated randomly by the GA).

b. Consider the first of the n_{child} - n_{cluster} remaining individuals and place it in the cluster whose centroid is closest.

c. Re-compute the centroid of the altered cluster.

d. Repeat steps 2 and 3 until the whole population is clustered.

e. The whole population is now clustered in such a way that each individual is closer to their respective cluster centroid than to any other.

Ward's method clusters the GA population in the following way

a. Each of the n_{child} individuals generated by the GA at each generation forms an initial cluster site.

b. The number of cluster sites is reduced one at a time until n_{cluster} clusters are remaining.

c. At each cluster reduction, the method merges the two clusters resulting in the smallest increase in the total sum of squares of the distances of each individual to its respective cluster centroid. This total sum increases monotonically as the number of clusters decreases.

d. Sites clustered at a previous clustering step are never unmerged.

In problems where the components of the vector data sample differ by several orders of magnitude, as is the case with PSR or PREMIX modelling it is necessary to scale the components such that they all belong to [0,1]. This is accomplished in the following way:

$$\tilde{x}_k = \frac{x_k - x_k^{\text{lower}}}{x_k^{\text{upper}} - x_k^{\text{lower}}} \tag{4}$$

where, x_k^{upper} and x_k^{lower} are the upper and lower bounds respectively of the search space for the k^{th} component of the vector data sample.

There are many ways to choose the cluster representatives, but the easiest is to randomly select an individual from those present in each of the clusters. The fitness of the cluster representative is then evaluated using the original fitness function (2).

The fitness of the remaining individuals is estimated from the their cluster representative in proportion to the Euclidean distance from the representative. Using the Euclidean distance metric, the distance from the cluster representative of the p^{th} cluster to the q^{th} individual in the p^{th} cluster is computed as follows:

$$d\left(\underline{x}_p^{\text{rep}}, \underline{x}_p^q\right) = \sqrt[2]{\sum_{k=1}^{n} \left| \tilde{x}_{pk}^{\text{rep}} - \tilde{x}_{pk}^q \right|^2} \tag{5}$$

In order to make the clustering model independent of the dimension of the problem, and thus ensuring its effectiveness on the four test problems, which vary from a dimension of 57 for hydrogen through to 1380 for kerosene, the Euclidean distances need to scaled such that they all belong to [0,1]. This is accomplished by dividing the Euclidean distance given in (5) by the maximum theoretical distance as follows:

$$\tilde{d}\left(\underline{x}_p^{\text{rep}}, \underline{x}_p^q\right) = \frac{d\left(\underline{x}_p^{\text{rep}}, \underline{x}_p^q\right)}{\sqrt[2]{n}} \tag{6}$$

The final stage in the application of the clustering algorithm is to compute the indirect fitness of a non-representative cluster individual based on its distance from the cluster representative. A simple but effective form for this indirect evaluation is as follows:

$$\frac{\text{Fit}\left(\underline{x}_p^q\right)_{\text{indirect}}}{\text{Fit}\left(\underline{x}_p^{\text{rep}}\right)_{\text{direct}}} = \left(1 - \tilde{d}\left(\underline{x}_p^{\text{rep}}, \underline{x}_p^q\right)\right) \tag{7}$$

4 Numerical Results

As outlined in section 2.3 the two clustering-based GAs will each be tested on four different mixture problems and their results compared to those obtained with a standard GA (s1GA) that uses population sizes of $n_{\text{pop}} = 50$ and $n_{\text{child}} = 60$. The *k*-means clustering GA (kmGA) and the Ward's method GA (WmGA) both use population sizes equal to those of the s1GA, and a value of $n_{\text{cluster}} = 10$ is chosen in each case. This value for n_{cluster} strikes a good compromise between the quality of solution obtained and the overall computational requirements of the GA. The computational expense of clustering the population and evaluating the indirect fitnesses at each generation is negligible for both the kmGA and the WmGA when compared to the cost of evaluations involving the original fitness function (2). Thus, since only $n_{\text{cluster}} = 10$ evaluations of the original fitness function (2) are made at each generation for both the kmGA and the WmGA, an approximately six-fold decrease in operational runtime is observed over the s1GA for an equivalent number of generations performed. It is worth noting that the clustering of the population in the WmGA is significantly more computationally demanding than that in the kmGA but is still negligible when

compared with original fitness function evaluations, with the extent of this negligibility becoming more pronounced as the number of species increases. The results from the clustering-based GAs are also compared to those obtained from a second standard GA (s2GA) that uses population sizes of $n_{pop} = 8$ and $n_{child} = 10$. The computational requirements of the s2GA are approximately comparable to both the kmGA and the WmGA as all three perform an equal number of evaluations of the original fitness function (2) per generation.

Figures 1(a), 1(b), 1(c), and 1(d), corresponding to the hydrogen-PSR problem, the formyl-PSR problem, the methane-PSR problem, and the kerosene-PSR problem respectively, present evolutions of the average fitness value of the fittest individual at each generation as a function of generation number for the four different GAs used. In Figure 1(a), the kmGA clearly outperforms the other algorithms attaining a significantly higher fitness value after completing 1000 generations. For this problem the overall performance of the WmGA and the s1GA are comparable as they both attain a similar fitness value after 1000 generations. The overall performance of the s2GA is significantly worse than any of the other three algorithms due in most part to the reduced population sizing used in this algorithm. With only $n_{child} = 10$ individuals created at each generation in comparison to $n_{child} = 60$ for the other algorithms; the s2GA is at a significant disadvantage when it comes to effectively searching the solution domain. This performance trend for the s2GA can also clearly be observed in Figures 1(b), 1(c), and 1(d).

Similar trends to those observed in Figure 1(a) can also be observed in Figure 1(b) with the exception that now both clustering-based GAs exhibit a marked performance gain over the standard GAs. It is worth noting that when moving from the hydrogen problem (Figure 1(a)) to the formyl problem (Figure 1(b)), corresponding to an increase in the dimension of the solution domain, the relative performance of the kmGA with respect to the WmGA decreases. Figures 1(c) and 1(d) show that the WmGA now outperforms the kmGA as well as the standard GAs, demonstrating that this is a more suitable clustering algorithm when dealing with vector data sets of very high dimension.

Figures 2(a), 2(b), 2(c), and 2(d), corresponding to the same sequence of test problems as those in Figure 1, present the average percentage error in predicting the mole fractions of selected output species at a given temperature based on reaction mechanisms generated by the various GAs. The prediction errors indicate the extent to which each of the four GA-optimised mechanisms can recover the output species mole fractions that are produced from a simulation experiment involving a previously validated original reaction mechanism. All four GA-optimised mechanisms are in very good agreement with the original validated mechanism for the first three test problems at temperature $T = 1500$K as shown in Figures 2(a), 2(b), and 2(c). The agreement in the case of the kerosene problem for the same output species is in general, not as good; see Figure 2(d). This could be due in part to the difficulty of simultaneously recovering the mole fractions of 97 output species as is required for the kerosene mechanism.

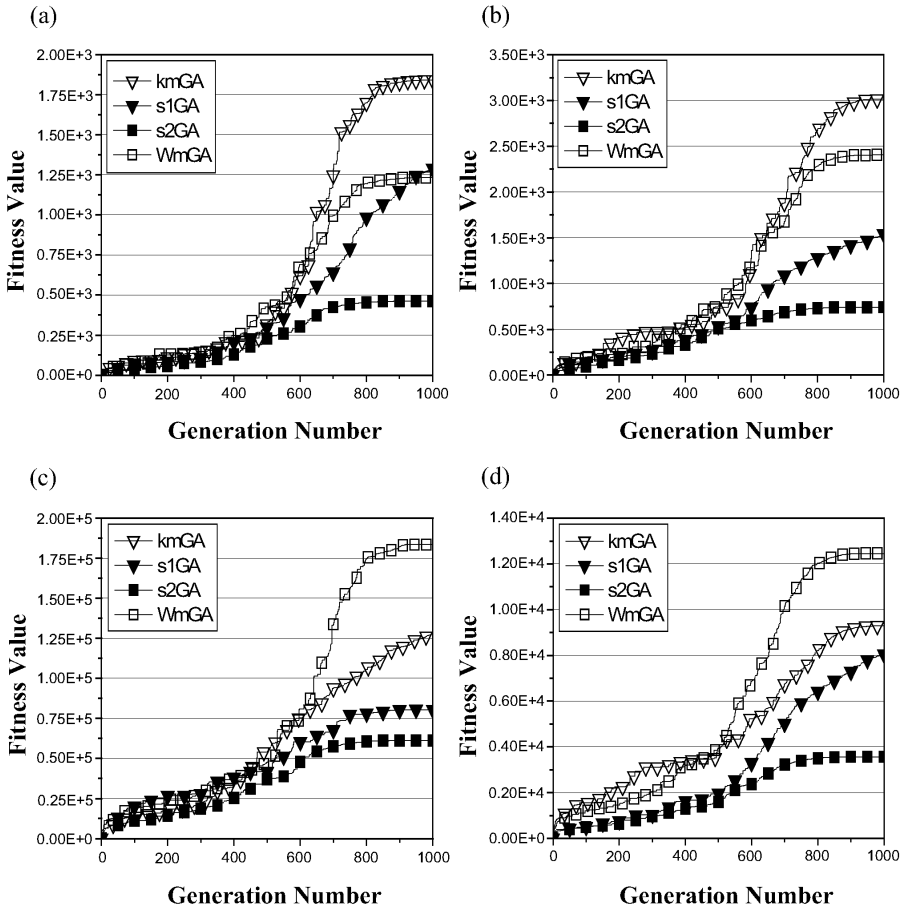

Fig. 1. The average fitness value of the fittest individual at each generation based on six differently seeded GA runs as a function of generation number for the *k*-means genetic algorithm (kmGA), the Ward's method genetic algorithm (WmGA), the first standard genetic algorithm (s1GA), and the second standard genetic algorithm (s2GA). Results are displayed for four different fuel/air mixtures, (a) hydrogen/air, (b) formyl-radical/air, (c) methane/air, and (d) kerosene/air.

The ability of the s1GA-optimised mechanism to recover the output species mole fractions for each of the four problems is in general not as good as that of the other three mechanisms, and this accounts for its relative lack of performance as shown in Figure 1. Similar validation results, although not presented here, are observed at the other PSR operating conditions.

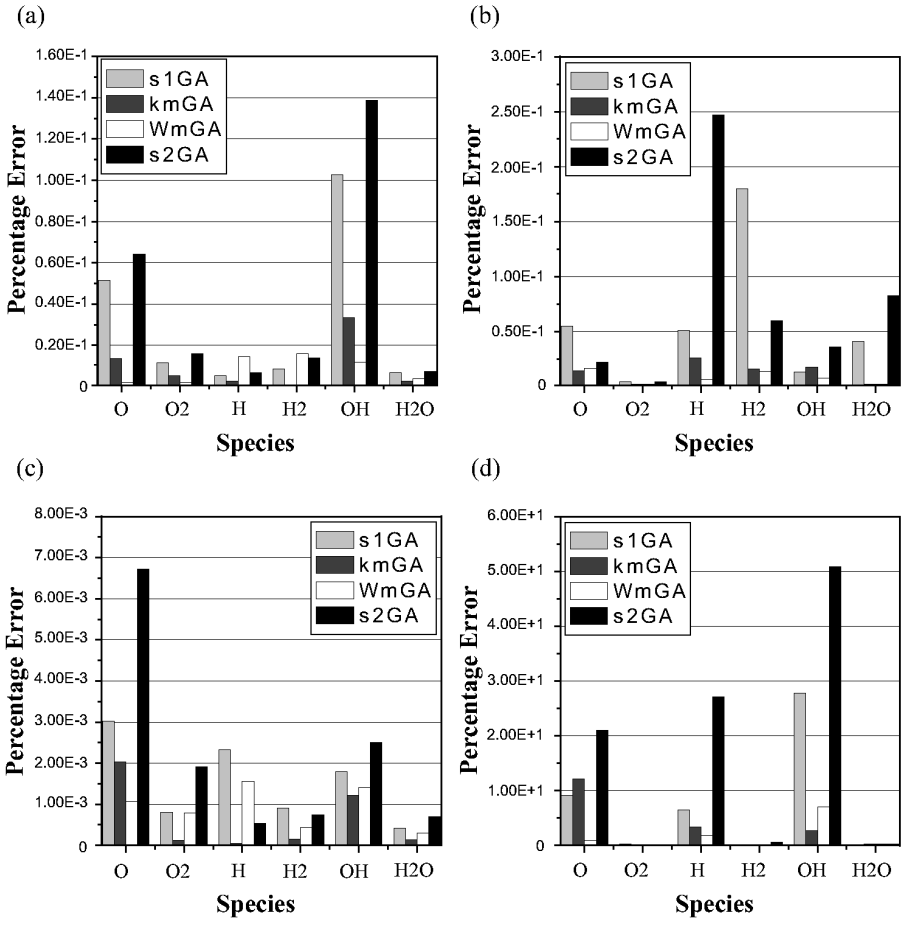

Fig. 2. The average percentage error in predicting the mole fractions of various output species based on reaction mechanisms generated by the *k*-means genetic algorithm (kmGA), the Ward's method genetic algorithm (WmGA), the first standard genetic algorithm (s1GA), and the second standard genetic algorithm (s2GA). Results are displayed for four different fuel/air mixtures, (a) hydrogen/air at temperature T = 1500K, (b) formyl-radical/air at temperature T = 1500K, (c) methane/air at temperature T = 1500K, and (d) kerosene/air at temperature T = 1080K.

5 Conclusion

In this study two efficient clustering-based genetic algorithms (GAs) have been applied to the problem of optimising reaction rate parameters for the combustion of four different fuel/air mixtures in a perfectly stirred reactor (PSR) over various operating

conditions. The clustering algorithms divide the entire population at each generation into groups based on a distance similarity measure. A representative is chosen for each group and its fitness is calculated directly using the computationally expensive original PSR fitness function. The fitness values of the remaining individuals in each cluster are calculated indirectly from their cluster representative, thus significantly reducing the total number of original PSR fitness function evaluations.

The clustering-based GAs are shown to outperform a standard GA in simulated experiments over the four test problems, whilst at the same time yielding an approximate six-fold reduction in the computational time required to complete an equal number of generations.

There remains the potential for future research applying clustering-based GAs to PSR combustion. One key area of consideration is the choice of similarity measure. At present, a Euclidean distance measure is used to assign similarity amongst individuals, but perhaps a more suitable choice involves using the rates of production associated with each species. It is these rates of production that are inexpensively calculated within the PSR program and then used to expensively determine the output species mole fractions. It is hoped that such a change of similarity measure will bring the clustering closer to the physical nature of PSR combustion.

Acknowledgements. The author would like to thank the EPSRC for their funding towards this study.

References

1. Dixon-Lewis, G., Goldsworthy, F.A., and Greenberg, J.B.: Proceedings of the Royal Society, London, A346, pp. 261-275, (1975).
2. Dagaut, P., Reuillon, M., Boetner, J.C., and Cathonnet, M.: Kerosene Combustion at Pressures up to 40atm: Experimental Study and Detailed Chemical Kinetic Modelling. Proceedings of the Combustion Institute, Vol. 25, pp. 919-926, (1994).
3. Polifke, W., Geng, W., and Döbbeling, K.D.: Optimisation of Rate Coefficients for Simplified Reaction Mechanisms with Genetic Algorithms. Combustion and Flame, Vol. 113, pp. 119-135, (1998).
4. Harris, S.D., Elliott, L., Ingham, D.B., Pourkashanian, M., and Wilson, C.W.: The Optimisation of Reaction Rate Parameters for Chemical Kinetic Modelling of Combustion Using Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering, Vol. 190, pp. 1065-1083, (2000).
5. Kee, R.J., Miller, J.A., and Jefferson, T.H.: CHEMKIN: A General-Purpose, Problem-Independent, Transport Table, FORTRAN Chemical Kinetics Code Package. Sandia Report SAND80-8003, Sandia National Laboratories, (1980).
6. Glarborg, P., Kee, R.J., Grcar, J.F., and Miller, J.A.: PSR: A FORTRAN Program for Modelling Well-Stirred Reactors. Sandia Report SAND86-8209, Sandia National Laboratories, (1988).
7. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York, (1996).

8. Rasheed, K.: An Incremental-Approximate Clustering Approach for Developing Dynamic Reduced Models for Design Optimisation. Proceedings of the 2000 Congress on Evolutionary Computation, pp. 986-993, (2000).
9. Ratle, A.: Accelerating the Convergence of Evolutionary Algorithms by Fitness Landscape Approximation. Parallel Problem Solving from Nature - PPSN V, Springer-Verlag, pp. 87—96, (1998).
10. Rasheed, K., Vattam, S., and Ni, X.: Comparison of Methods for Using Reduced Models to Speed Up Design Optimisation. Proceedings of the Genetic and Evolutionary Computation Conference 2002, pp. 1180-187, (2002).
11. Gunn, S.R.: Support Vector Machines for Classification and Regression. Technical Report, University of Southampton, Faculty of Engineering and Applied Science, Department of Electronics and Computer Science, (1998).
12. Jin, Y.: Fitness Approximation in Evolutionary Computation – A Survey. Proceedings of the 2002 Genetic and Evolutionary Computation Conference 2002, pp. 1105-1112, (2002).
13. Gose, E., Johnsonbaugh, R., and Jost, S.: Pattern Recognition and Image Analysis. Prentice Hall PTR, (1996).