

# Modelling and Exploring the Systems Design Space

## Using a Hybrid Genetic Algorithm

**Shuiwei Xie**

Aerospace, Civil & Mechanical  
Engineering  
University College, UNSW  
Australian Defence Force Academy  
CANBERRA ACT AUSTRALIA 2600  
Phone: 02 6268 8258 Fax: 02 6268 8276  
E-mail: [s.xie@adfa.edu.au](mailto:s.xie@adfa.edu.au)

**Warren F. Smith**

Aerospace, Civil & Mechanical  
Engineering  
University College, UNSW  
Australian Defence Force Academy  
CANBERRA ACT AUSTRALIA 2600  
Phone: 02 6268 8262 Fax: 02 6268 8276  
E-mail: [w.smith@adfa.edu.au](mailto:w.smith@adfa.edu.au)

### Abstract

*Reported in this paper is a work in progress at the University of New South Wales at the Australian Defence Force Academy (UNSW @ ADFA). The notion of a system being represented by a set of sub-system design problems is not new. However, the ability to deal with the system's sub-problems generically and direct them to different solvers as appropriate within a design decision support environment has not been well explored. Our current research objective may be summarised as an effort to answer the following question.*

*Given that decision support and design optimisation techniques are generally applicable in the design of systems, can better real world decisions be modelled and explored using a hybrid recursive solver?*

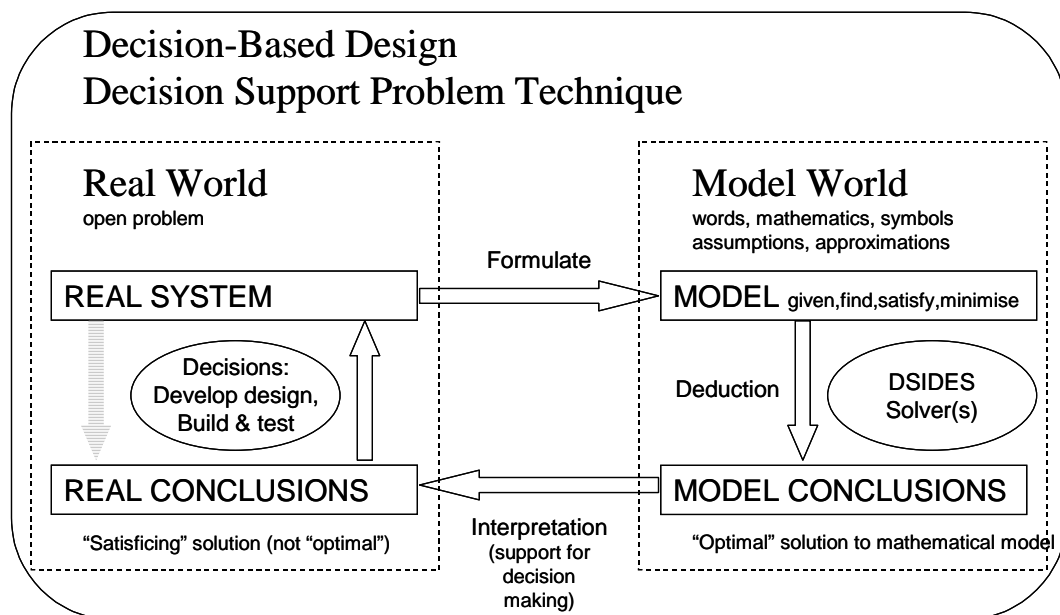
*In general, much has been done in the area of "shape" optimisation in design. While not necessarily "easy" in their own right, applications in this area are served well by the variables being continuous. What has proved "hard" is dealing with configuration / layout issues because the variables are discrete and functions are discontinuous. Thus, the innovation, significance and value to be derived from the work begun and reported is the development of an efficient and effective method to address for example shape and layout issues concurrently. This implies the ability to computationally deal with the decomposition and synthesis of a system design. A genetic algorithm (GA) is used to deal with discrete aspects of a design model (e.g., allocation of space to functions) and sequential linear programming (SLP) and pattern search (PS) methods for the continuous aspects (e.g., sizing).*

### 1. Frame of reference

In providing design decision support, we subscribe to and have applied the paradigm of "decision-based design" (DBD) using the constructs of a method known as the Decision Support Problem Technique (DSPT). The technique offers a holistic systems approach that is broadly flexible and adaptable. The DSPT has been implemented in the software environment, DSIDES (Decision Support In the Design of Engineering Systems). The DSPT and DSIDES are well documented in (Mistree, Smith et al. 1990; Mistree, Smith et al. 1991; Mistree, Hughes et al. 1992; Reddy, Smith et al. 1992).

The relationship of DBD, the DSPT and DSIDES in designing through modelling the real world is portrayed in Figure 1. Once a model is appropriately formulated using the DSPT constructs (typically, a non-linear goal programming styled formulation), DSIDES, with its operations research tools, is used to deduce “model conclusions”. These may be seen to be optimal solutions to the closed form mathematical formulation presented. The designer is then at liberty to interpret these conclusions for the real world in the light of the formulated assumptions and approximations.

Reported in this paper is a work to enhance the “model world” in the context of Figure 1. Because we work with mathematical constructs, the “model” is written as a set of constraints and goals that can be expressed in matrix form.



**Figure 1 – Modelling the Real World**

## 2. The question

Modelling a system by decomposing it into a set of DSP sub-problems is not new. There is a large body of literature dealing with issues of decomposition and synthesis. However, exercising the model, exploring the design space and prosecuting satisficing<sup>1</sup> real world solutions, using different solvers for different sub-problems within a common design environment have not been fully explored. Our current research objective may be summarized as an effort to answer the following question.

*Given that decision support and design optimisation techniques are generally applicable in the design of systems, can better real world decisions be modelled and explored using a hybrid recursive solver?*

<sup>1</sup> The first use of the term “satisficing” in the context of optimisation is attributed to Herbert Simon [Simon, H. A. (1982), *The Science of the Artificial*, Cambridge, Massachusetts, The MIT Press] and it is used to connote not the “best”, but “good enough”.

### 3. Step 1 – Setting the Direction

To answer this fundamental question, a range of issues arises regarding decomposition and naturally the subsequent reconstructive synthesis.

#### Decompose – Why?

Decomposition of a design problem is generally effected to aid the human designer in managing the problem. If a problem is decomposed, three possible benefits follow.

- *Easier and Cheaper Solutions Are Obtained*

Some problems are difficult to solve using conventional optimisers without decomposition. If problems are addressed without decomposition, often unique and expensive solvers need to be employed. However, the problem complexity generally still forces simplification and approximation in the models utilized and a compromise between accuracy and computing cost is necessarily made.

- *Faster Solutions Obtained*

In an effort to save calendar time, or in some cases, total aggregate time, parallel / distributed computing / analysis can be employed. This demands the use of decomposition.

- *Choice of Solvers Provided*

When a systems design problem is decomposed into a number of levels and sub-problems, these “modules” may have different mathematical characteristics. To solve these system problems efficiently, different optimisation algorithms can be applied independently to the modules.

It is fundamentally this opportunity that has motivated the current research effort at UNSW @ ADFA. If one is then to use decomposition, it follows to ask how.

#### Decompose – How?

- *Traditionally*

Traditionally, large-scale problems are decomposed along discipline boundaries. This decomposition procedure generally draws on the designer’s insight to choose an appropriate scheme. For example, structures and controls in aircraft design (James 1993), the hull form and propellers in ship design (Smith 1992), and journal bearing and thrust bearing in spindle system design (Montusiewicz and Osyczka 1990). Traditionally, no mathematical character of the problem is taken formally into account; rather the engineering boundaries are the decomposition criteria. The decomposition may well reflect the company organizational structure. It may also reflect the cultural bias within the specific branch of engineering.

- *Non Traditionally*

In this method, we imply that a problem is decomposed along the lines of the model’s mathematical characteristics rather than “engineering”. The less interaction (“feedback links” or “coupling”) among flexibly defined sub-problems, the more independent the sub-problems are. This non-traditional mathematically based approach can lead to difficulties in interpretation as the coefficient matrix is forced to be more diagonally dominant. Constraints and goals that become grouped in a module may not intuitively sit well together when

considering function. However, closer examination and lateral thinking may lead to better understanding.

One point to be made in considering decomposition and then the reconstitution or synthesis that must follow, is that it should not negatively impact any of the overall system goals. There is a possibility in dividing a problem that the effect on the overall design is not properly considered, akin to combining the results of multiple single objective optimisations. This may limit the usefulness in that the interactions are missed and satisficing solutions or even feasibility at the system level is not achieved.

Our current research direction is not being defined by either a traditional or non-traditional approach to decomposition. Rather, we plan to explore the opportunities afforded by both.

### **Decompose – What Structures?**

The structures chosen are important. The strength of the sub-problem interactions (variously referred to as strong or weak) clearly influences model and solver performance. One key consideration is whether it is efficient for a variable to be optimised in two sub-problems or whether a variable in one sub-problem can only be seen as a parameter in the other sub-problems. This said, there are two broad structures that could be employed, namely, heterarchical and hierarchical.

- *Heterarchical Structures*

A heterarchy is a formal organisation of nodes without any single permanent uppermost node – a non-hierarchical organisation. There is no system level in such a structure. All sub-problems are at the same level and have the ability to share variables and information among sub-problems in all directions. The relationships between sub-problems are not ordered and two-way communication occurs in all directions.

- *Hierarchical Structures*

In this structure, a sub-problem exchanges data directly with the system (or parent) level only and not with any other sub-problem at the same level directly. There could be more than two levels. The relationships between “parents” and “children” are ordered and two-way communication only occurs between them.

Our thesis in respect to structure is that a hierarchical approach is most advantageous. It offers the ability for coordination at the system level (or parent levels) and maximises the potential for understanding and tracking the process when applied in the early stages of design. This gives the further flexibility of working only to the appropriate “depth” of level (detail of solution) in searching for approximate solutions to the most significant variables.

Given a large scale problem has been hierarchically decomposed into a number of sub-systems according to an adopted criteria, the sub-problems can be solved using different solvers. The next step taken then is to develop a computational environment that facilitates this.

## 4. Step 2 – Solver Development

### 4.1 Hybrid genetic algorithms in general

Genetic algorithms are widely used due to their abilities to solve many kinds of optimization problems in a simple way. It is shown theoretically that genetic algorithms have the ability to find optima of problems in a stochastic way, (Eiben, Aarts et al. 1991; Rudolph 1994; He, Kang et al. 1995). But, many practical applications show genetic algorithms are not always satisfactory due to their likelihood to converge prematurely. They lack rigorous local search ability as well as exhibiting low efficiency when applied to specific domain problems.

On the other hand, optimization techniques such as gradient search, sequential linear and quadratic programming, hill climbing, simplex method and simulated annealing are very efficient in local search. Some specific domain knowledge based and heuristic methods also perform very well in solving some kinds of problems. A number of hybrid genetic algorithms integrating genetic algorithms and optimization techniques such as those mentioned above, have the potential to make full use of the advantages of the methods employed, providing an efficient solution method for many optimization problems.

By a hybrid algorithm we mean an optimization algorithm that combines a genetic algorithm with either one or more non-genetic algorithms.

Most existing hybrid genetic algorithms can be categorized as sequential or parallel schemes.

- *Sequential:*

In this combination, genetic algorithms and local improvement methods are applied to the same group of potential solutions sequentially. A local improvement algorithm may be applied once when the GA converges, to get a more precise final result (Regué, Ribó et al. 2001) or applied in each GA iteration until termination conditions are satisfied (Ishibuchi, Yamamoto et al. 1994; Freisleben 1997; Huo, Xu et al. 2000).

- *Parallel:*

Alternatively, a GA and a non-GA algorithm are applied to different candidates to enhance both global and local search capability. As an example, an “elitism GA” (one with a selection strategy of copying the best candidate to next generation without suffering crossover or mutation) combined with simplex method by proportion of the whole population was demonstrated in (Yen 1995; Narieda 2000). A proportion based combination of a GA and a hill-climbing method are applied in (Lobo 1997). Papadopoulos (Papadopoulos, Mack et al. 2000) proposed a method dividing the population into three parts that applied different operators. Normal crossover and mutation was applied to one part, Gauss-Newton local search to the second part and direct copy plus a mutation process was applied to the third part.

Different hybridisation schemes (GA and non GA combinations) have been published. Examples include the use of a heuristic rule / specific knowledge based method (Gockel 1996; Wang 1997; Lo and Chang 1999; Kwan 2000), the use of Neural Network Dynamic Programming (Jih and YungJen Hsu 1999), gradient based methods (Grimbleby 1999), and the simplex method (Harris 1998, Dec.).

## 4.2 UNSW @ ADFA proposed hybrid scheme

A genetic algorithm solver has been successfully written and integrated in the DSIDES environment (Xie and Smith 2002). A summary of a paper, titled, “Towards a Hybrid Solver – Integration of a Genetic Algorithm within DSIDES” that reports this activity follows.

Through introducing a genetic algorithm as a solver in DSIDES, the intention was to improve the likelihood of finding the global minimum (for the formulated model) as well as the ability of dealing more effectively with non-linear problems, which have discrete variables, undifferentiable objective functions or undifferentiable constraints. Using some numerical examples and a practical ship design case study, the proposed GA based method was demonstrated to be better in maintaining diversity of populations, preventing premature convergence, compared with other similar GAs. It also has similar effectiveness in finding the solutions as the original adaptive (sequential) linear programming DSIDES solver.

The GA method, which is a modified version of Gen’s “Genetic Algorithm for nonlinear goal programming” (Gen and Cheng 1996) and implemented in DSIDES, is characterised by five steps, namely: parameter setting, population initialisation, evaluation and ranking, creation of a new population (generation) and termination. To create a new population the processes employed are selection, crossover and mutation.

Genetic algorithms can be slow to converge. In an effort to accelerate convergence, the better candidates can be more heavily weighted (using their “fitness” directly or their rank). There are advantages and disadvantages to each approach. However, over-weighting can lead to a selection process that results in a population lacking diversity, which in turn leads to premature convergence.

An innovative strategy to minimise domination is to eliminate duplication in the leading positions of the ranking vector of a population. This has shown good robust results as documented in (Xie and Smith 2002).

In this method employing a lexicographical goal programming approach, satisfying the constraints is set as a first priority goal. That is, the model is reformulated and solved as an unconstrained multi levelled problem, thus creating flexibility for a chosen solution algorithm. At the same time a proportion of infeasible individuals may now exist in the population, which can positively contribute to the diversity of the population.

As a subset of constraints, the variable bounds are treated specially in this method. The bounds form “walls” for the feasible search space. Individuals attempting to violate a wall during crossover and mutation are restrained to the corresponding bound.

Given now a direction and a toolbox, which includes a hybrid genetic algorithm, we have begun to develop and explore a case study based on a 3-stage gearbox parameter optimisation.

## 5. Step 3 – Development of a Case Study

The main purpose of this example is to illustrate the ability of the hybrid GA-based DSIDES solvers to deal with complex real-world non-linear goal programming problems. The

example involves the determination of that set of principal parameter values that satisfy a multitude of constraints and best achieve a set of goals.

An overview of the design requirements and the word form of the DSPT template for the gearbox design follows. The model has been developed from a single stage gearbox design reported in (Golinski 1970; Liu 1994) and is initially presented in an “all-in-one” form.

## **“ALL-IN-ONE” 3 STAGE GEARBOX MODEL**

### **GIVEN**

Transmitted power  
Input shaft speed  
Total transmission ratio  
Permissible bending stress of gear teeth  
Permissible surface compressive stress for both gears  
Permissible bending stress for both shafts  
Maximum deflection of shaft  
....

### **FIND**

The values of the design parameters:  
 $Z_i$ ,  $i=1$  to 3 -Number of tooth of each pinion  
 $M_i$ ,  $i=1$  to 3 -modules of gears  
 $B_i$ ,  $i=1$  to 3 -face width  
 $I_i$ ,  $i=1$  to 3 -transmission ratios of each stage  
 $D_j$ ,  $j=1$  to 4 -diameters of four shafts  
 $L_j$ ,  $j=1$  to 4 -length of four shafts  
and the deviation variables:  $d^+$  and  $d^-$

### **SATISFY**

The following system constraints:

1. Upper bound on the bending stress of the gear tooth for gear sets 1, 2 and 3.
2. Upper bound on the contact stress of the gear tooth gear sets 1, 2 and 3.
3. Upper bound on the transverse deflection of the shaft 1, 2, 3 and 4.
4. Upper bound on the stress of the shaft 1 and 4.
5. Relative face width conditions for pinion 1,2 and 3.
6. Upper bound on the bending stress of the gear tooth for gear sets 1, 2 and 3.
7. Upper bound on the contact stress of the gear tooth gear sets 1, 2 and 3.
8. Upper bound on the transverse deflection of the shaft 1, 2, 3 and 4.
9. Upper bound on the stress of the shaft 1 and 4.
10. Relative face width conditions for pinion 1,2 and 3.
11. Design conditions for the shaft based on experience shaft 1 and 4.
12. Length of shaft 2 and 3 to sit two gears.
13. Interferences between Shaft 3 and Gear 2, Shaft 4 and Gear 4.
14. 40 boundary constraints for 20 variables

## **MINIMISE**

The deviation function, representing the designer's preference for goal satisfaction

$$Z = [f_1(\underline{d}^-, \underline{d}^+), \dots, f_k(\underline{d}^-, \underline{d}^+)]$$

## **GOAL HIERARCHY**

### **Level 1**

- achieve the desired total volume of the gear sets.

### **Level 2**

- achieve the desired accumulative distance between shaft centres.

As an alternative, the above design problem can be decomposed into two loose coupled sub-problems according to a hierarchical goal programming oriented decomposition method in which the goal or objective that reflects the decision maker's preference is taken into consideration.

The goal of the gearbox parameter optimal design is minimization of total volume of the 3-stage gearbox. The following 2 sub-goals contribute to this goal:

1. Minimization of distance between shaft centres and,
2. Minimization of solid volume of gears and shafts.

The first sub-goal can be achieved by optimisation of gear set parameter design. Independent variables include speed ratios, face widths and number of teeth. The second sub-goal is shaft parameter optimal design. Diameters and the lengths of shafts are variables.

....

## **DECOMPOSED MODEL**

### **"PARENT" – SUB-PROBLEM 1 – GEARS**

#### **GIVEN**

Transmitted power  
Input shaft speed  
Total transmission ratio  
Permissible bending stress of gear teeth  
Permissible surface compressive stress for both gears

....

#### **FIND**

The values of the design parameters:  
 $Z_i$   $i=1$  to 3-Number tooth of each pinion  
 $M_i$   $i=1$  to 3-modules of gears  
 $B_i$   $i=1$  to 3-face width  
 $I_i$   $i=1$  to 3 -transmission ratios of each stage  
and the deviation variables:  $\underline{e}^+$  and  $\underline{e}^-$

## **SATISFY**

The following system constraints:

1. Upper bound on the bending stress of the gear tooth for gear sets 1, 2 and 3.
2. Upper bound on the contact stress of the gear tooth gear sets 1, 2 and 3.
3. Relative face width conditions for pinion 1,2 and 3.
4. Interferences between Shaft 3 and Gear 2, Shaft 4 and Gear 4.
5. 24 boundary constraints for 12 variables

## **MINIMISE**

The deviation function, representing the designer's preference for goal satisfaction

$$Z = [f_1(e^-, e^+), \dots, f_k(e^-, e^+)]$$

## **GOAL HIERARCHY**

### **Level 1**

- achieve the desired accumulative distance between shaft centres

### **Level 2**

- achieve the desired total face width.

## **DECOMPOSED MODEL**

### **“CHILD” – SUB-PROBLEM 1.1 – SHAFTS**

#### **GIVEN**

Transmitted power

Speed of each shaft

Tooth number, module, face width of each gear

Transmission ratio of each stage

Permissible bending stress for both shafts

Maximum deflection of shaft

....

and a starting design

$$X^0 = (X_i^0 : i = 1, \dots, 8)$$

#### **FIND**

The values of the design parameters:

$D_j, j=1$  to 4 -diameters of four shafts

$L_j, j=1$  to 4 -length of four shafts

and the deviation variables  $c^+$  and  $c^-$

#### **SATISFY**

The following system constraints:

1. Upper bound on the transverse deflection of the shafts
2. Upper bound on the stress of the shafts
3. Design conditions for the shaft based on experience
4. Length of shaft 2 and 3 to sit two gears.
5. Interferences between Shaft 3 and Gear 2, Shaft 4 and Gear 4.
6. 16 boundary constraints for 8 variables

**MINIMISE**

The deviation function, representing the designer's preference for goal satisfaction

$$Z = [f_1(c^-, c^+), \dots, f_k(c^-, c^+)]$$

**GOAL HIERARCHY****Level 1**

- achieve the desired the total volume of the gear sets

## 6. Indicative Case Study Results

The undecomposed “all-in-one” gearbox model has a discrete variable  $M$  (module) and an integer variable  $Z$  (tooth number) for each pinion. This leads to the GA as the chosen solver due to the presence of mixed variables. In the decomposition scheme, Sub-Problem 1 (SP1) has the same integer and discrete variables, and the GA is again chosen. Since for Sub-Problem 1.1 of the decomposed scheme all 8 variables are continuous, pattern search is perceived to be more efficient than the GA to handle this sub-problem.

The coupling of SP1 and SP1.1 is based on the physical constraint that non-mating gears on one shaft cannot interfere with neighbouring shafts. This is achieved with a constraint that the shaft centre distance must be greater than the shaft radius plus the radius of the non-mating gear. If interference occurs after finishing the SP1.1 optimisation, the “design” is deemed infeasible and further iteration is required.

## 7. Results and analysis

The results from the two comparative models (all-in-one and decomposed) using a GA / Pattern Search hybrid environment are presented in Table 1. In Table 1, “Ten GA runs” mean that ten different initial populations were trialled due to the random nature of the GA. Similarly, ten different starting points were assigned to each pattern search (PS) run for comparison. All initial starting points for the pattern search algorithm were infeasible and 99% or more individuals in the initial populations for each GA run were also found to be infeasible.

It is obvious that in the undecomposed all-in-one model, the GA had difficulties in achieving a global minimum in most of its ten runs, and high numbers of iterations were needed to obtain convergence. The difficulty, due to the complexity of the optimisation problem, is thought to be a function of the ratio of population size to number of variables. Shown in Table 2 is that when solving the “all-in-one” model, it was found that during all ten GA runs all individuals in each initial population are infeasible.

By comparison, in the decomposed model, the same genetic algorithm can handle the parent sub-problem, SP1, very well. From Table 2 it can be seen that in the decomposed scheme, the number of feasible solution in both initial and final population increased.

Both the employed GA and PS algorithms could reach a similar local minimum in their ten runs as shown in Table 1. The results of this case study point to better and more stable solutions being achieved from a decomposed structure than an all-in-one structure using these solvers.

**Table 1-Computing results of 3 models**

| Of 10 random runs   | All-in-one Undecomposed (GA)      | Decomposed Level 1 (GA)       | Decomposed Level 2 (GA) | Decomposed Level 2 (PS) * |
|---------------------|-----------------------------------|-------------------------------|-------------------------|---------------------------|
| Average achievement | 4406 cm <sup>3</sup><br>175.15 cm | 164.8 cm<br>12.2 cm (f width) | 3255 cm <sup>3</sup>    | 3253 cm <sup>3</sup>      |
| Best achievement    | 3680 cm <sup>3</sup><br>165.8 cm  | 164.8 cm<br>11.6 cm (f width) | 3254 cm <sup>3</sup>    | 3253 cm <sup>3</sup>      |
| Average iterations  | 16460                             | 14000                         | 9100                    | 24.6                      |

*\*1 of the ten PS runs converged at an infeasible solution with the achievement value 3296.71, the violation of constraints was very small, 1.43051e-007 can could be considered feasible.*

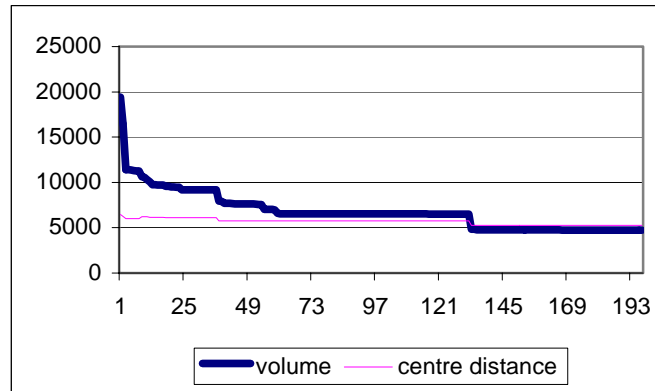
**Table 2 -Number of feasible solution in GA iterations**

| Number of feasible solutions | All-in-one Undecomposed (GA) | Decomposed Level 1 (GA) | Decomposed Level 2 (GA) |
|------------------------------|------------------------------|-------------------------|-------------------------|
| Initial population           | 0                            | 1                       | 3                       |
| Final population             | 10                           | 43                      | 35                      |

In considering only the child sub-problem 1.1, the difference in the results gained by the genetic algorithm and the pattern search algorithm shows that in this case, pattern search is more efficient based on average numbers of iteration. This is believed to be a result of the fact that the sub-problem variables are continuous.

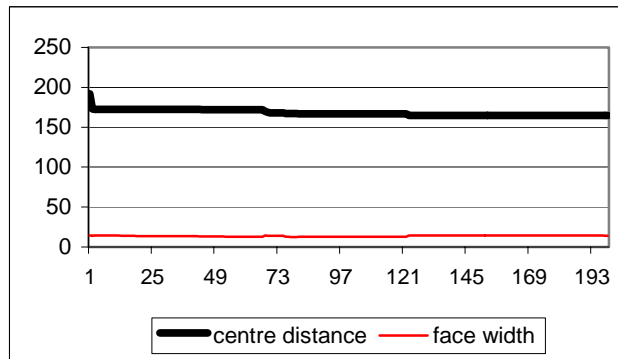
The disadvantage of a pattern search approach, which is very common in hill climbing methods, is that there is a risk in failing to find a feasible solution due to bad starting points being selected. One of the ten PS runs converged at a technically infeasible solution.

In Figures 2 to 5, the convergence of sample solutions using each approach is plotted. The main observation is that the decomposed model whether using pattern search or genetic algorithm for sub-problem 1.1, exhibits quicker and easier convergence as a whole.



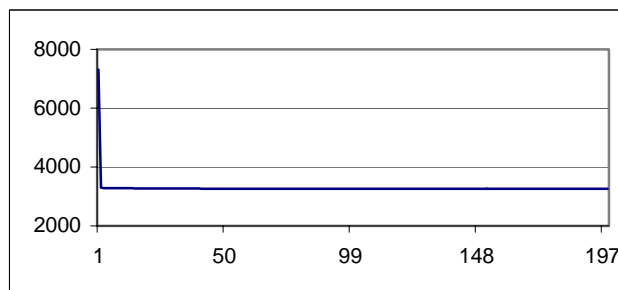
**Figure 2 – Iteration history of the best fitness for the undecomposed all-in-one scheme using the GA solver**

(X axis: hundreds iteration, Y axis: cm<sup>3</sup> for volume and cm multiplied by 30 for face width)



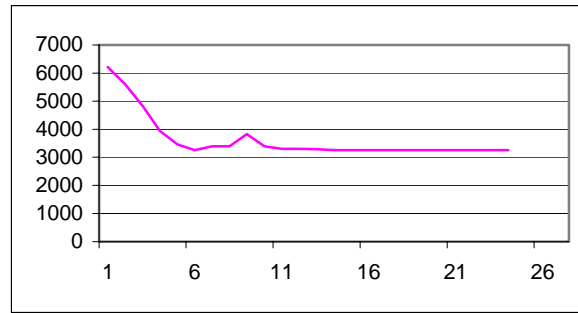
**Figure 3 – Iteration history of the best fitness of the parent sub-problem 1 for the decomposed scheme using the GA solver**

(X axis: hundreds iteration, Y axis: cm for centre distance)



**Figure 4 – Iteration history of the best fitness of the child sub-problem 1.1 for the decomposed scheme using the GA solver**

(X axis: hundreds iteration and Y axis: volume [cm<sup>3</sup>])



**Figure 5 – Iteration history of the best fitness of the child sub-problem 1.1 for the decomposed scheme using the PS solver**

(X axis: iterations and Y axis: volume [cm<sup>3</sup>])

## 8. Future work

The results obtained are encouraging but are not yet generally conclusive. The success of the employed decomposed scheme mainly lies in the fact that the problem was decomposed appropriately and the decomposed gearbox parameter optimisation problem is loosely coupled. Pure continuous variables in the child sub-problem 1.1 made an application of a hybrid algorithm possible. Here a question rises naturally. Is the hybrid genetic algorithm generally applicable in systems design optimisation problems? If not, in terms of objective functions, constraint functions, variables and search space, what characteristic should exist in a problem to make the decomposed scheme and hybrid optimiser applicable and efficient? In other words, where and why does it work and where doesn't it work?

To answer this question, more case studies are being developed. As stated earlier, we are reporting a work in progress.

## 9. Conclusions

Fundamentally, the concept for the course of research is to contribute to the fields of DBD and multi-objective optimisation. The emphasis on building a hybrid solver is founded on the desire to offer sub-problems a “choice” in solution algorithm. This recognises the possibility of improved efficiency and / or effectiveness due to the matching of model and method.

From the gearbox case study, to solve complex engineering optimisation problems, there is a choice to solve with or without decomposition depending on the mathematical characteristics. It can be argued that better and more stable solutions can be achieved from a decomposed structure compared with an all-in-one structure. For some problems, quicker and easier solutions may be achieved from a decomposed scheme.

## REFERENCES

- Eiben, A. E., E. H. Aarts and H. K. M. Van (1991). Global Convergence of Genetic Algorithms: An Infinite Markov Chain Analysis. 1st International Conference on Parallel Problem Solving from Nature, Heidelberg, Berlin.
- Freisleben, B. S., A. (1997). Color Quantization with a Hybrid Genetic Algorithm. Image Processing and Its Applications, Sixth International Conference.
- Gen, M. and R. Cheng (1996). Genetic Algorithms and Engineering Design. New York, John Wiley & Sons Inc. press.
- Gockel, N., Pudelko, G., Drechsler, R., Becker, B. (1996). A hybrid genetic algorithm for the channel routing problem. Circuits and Systems, 1996. ISCAS '96., Connecting the World., 1996 IEEE International Symposium.
- Golinski, J. (1970). "Optimal Synthesis Problems Solved by Means of Nonlinear Programming and Random Methods." Journal of Mechanisms **5**: 287-309.
- Grimbleby, J. B. (1999). Hybrid Genetic Algorithms for Analogue Network Synthesis. Evolutionary Computation CEC 99. Proceedings of the 1999 Congress.
- Harris, S. P., Ifeachor, E.C. (1998, Dec.). Automatic design of frequency sampling filters by hybrid genetic algorithm techniques. Signal Processing, IEEE Transactions.
- He, J., L. S. Kang and Y. J. Chen (1995). "Convergence of Genetic Evolution Algorithms for Optimization." Parallel Algorithms and applications **5**: 37-56.
- Huo, H., X. Xu, J. Xu and B. Zheng (2000). Solving vertex covering problems using hybrid genetic algorithms. Signal Processing Proceedings, 2000. WCCCICSP 2000. 5th International Conference.
- Ishibuchi, H., N. Yamamoto, T. Murata and H. Tanaka (1994). "Genetic Algorithms and Neighborhood Search Algorithms for Fuzzy Flowshop Scheduling Problems." Fuzzy Sets and Systems **67**: 81--100.
- James, B. B. (1993). Multidisciplinary Optimization of a Controlled Space Structure Using 150 Design Variables, NASA.
- Jih, W. and J. YungJen Hsu (1999). Dynamic vehicle routing using hybrid genetic algorithms. Robotics and Automation Proceedings. IEEE International Conference.
- Kwan, R. S. K. W., A.; Kwan, A.S.K. (2000). Hybrid genetic algorithms for scheduling bus and train drivers. Evolutionary Computation, 2000. Proceedings of the 2000 Congress.
- Liu, W. (1994). Mechanical Optimal Design (Second Edition). Beijing, Tsinghua University Press.
- Lo, C. and W. Chang (1999). A Multiobjective Hybrid Genetic Algorithm for the Capacitated Multipoint Network Design Problem. ICC '99. IEEE International Conference on Communications.
- Lobo, F. G. G., D.E. (1997). Decision Making in a Hybrid Genetic Algorithm. Evolutionary Computation, IEEE International Conference.
- Mistree, F., O. F. Hughes and B. Bras (1992). Compromise Decision Support Problem and the Adaptive Linear Programming Algorithm. Structural Optimisation: Status and Promise. M. P. Kamat. Washington, DC, AIAA. **150**: 251 - 290.

- Mistree, F., W. F. Smith, B. Bras, J. K. Allen and D. Muster (1990). "Decision-Based Design: A Contemporary Paradigm for Ship Design." Transactions SAME **98**: 565-597.
- Mistree, F., W. F. Smith, S. Z. Kamal and B. A. Bras (1991). Designing Decisions: Axioms, Models and Marine Applications. Fourth International Marine Systems Design Conference, IMSDC, Kobe, Japan, Society of Naval Architects of Japan.
- Montusiewicz, J. and A. Osyczka (1990). "A Decomposition Strategy for Multicriteria Optimization with Application to Machine Tool Design." Engineering Costs and Production Economics **20**: 191-202.
- Narieda, S. Y. C. Y., K. (2000). Blind nonlinear channel identification based on higher order statistics using hybrid genetic algorithm. Systems, Man, and Cybernetics, 2000 IEEE International Conference.
- Papadopoulos, S., R. J. Mack and R. E. Massara (2000). A Hybrid Genetic Algorithm Method for Optimizing Analog Circuits. 43rd IEEE Midwest Symp. on Circuits and Systems, Lansing MI.
- Reddy, R., W. F. Smith, F. Mistree, B. A. Bras, W. Chen, A. Malhotra, K. Badhrinath, U. Lautenschlager, R. Pakala, S. Vadde and P. Patel (1992). DSIDES User Manual, Systems Design Laboratory, Department of Mechanical Engineering, University of Houston.
- Regué, J.-R., M. Ribó and J.-M. Garrell (2001). Radiated Emissions Conversion from Anechoic Environment to OATS Using a Hybrid Genetic Algorithm -- Gradient Method. IEEE.
- Rudolph, G. (1994). Convergence of Canonical Genetic Algorithms. IEEE transactions on Neural Networks.
- Simon, H. A. (1982). The Science of the Artificial. Cambridge, Massachusetts, The MIT Press.
- Smith, W. F. (1992). The Modeling and Exploration of Ship Systems in the Early Stages of Decision-Based Design. Interdisciplinary Operations Research Program. Houston, Texas, University of Houston.
- Wang, H. Z., J. (1997). The Hybrid Genetic Algorithm for Solving Nonlinear Programming. IEEE International Conference on Intelligent Processing Systems, Beijing.
- Xie, S. and W. Smith (2002). Towards a Hybrid Solvers- Integration of a Genetic Algorithm within DSIDES. DECT'02 ASME 2002 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Montreal, Canada, American Society of Mechanical Engineers, ASME.
- Yen, J., Randolph, D., Bogju Lee, Liao, J.C. (1995). A Hybrid Genetic Algorithm for the Identification of Metabolic Models. Tools with Artificial Intelligence Proceedings, Seventh International Conference.