IOWA STATE UNIVERSITY
Digital Repository

Retrospective Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2007

Engineering design using genetic algorithms

Xiaopeng Fang
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

 Part of the Mechanical Engineering Commons

Recommended Citation
Fang, Xiaopeng, "Engineering design using genetic algorithms" (2007). *Retrospective Theses and Dissertations*. 15943.
https://lib.dr.iastate.edu/rtd/15943

This Dissertation is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

www.manaraa.com

Engineering design using genetic algorithms

by

Xiaopeng Fang

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Mechanical Engineering

Program of Study Committee:
James E. Bernard, Co-major Professor
Julie A. Dickerson, Co-major Professor
Greg R. Luecke
Daniel Ashlock
Atul Kelkar

Iowa State University

Ames, Iowa

2007

Copyright © Xiaopeng Fang, 2007. All rights reserved.

www.manaraa.com

UMI Number: 3274886

UMI

UMI Microform 3274886

Copyright 2007 by ProQuest Information and Learning Company.
All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

www.manaraa.com

TABLE OF CONTENTS

www.manaraa.com

ABSTRACT

As modern computational and modeling technologies grow, engineering design heavily relies on computer modeling and simulation to accelerate design cycles and save cost. A complex design problem will involve many design parameters and tables. Exploring design space and finding optimal solutions are still major challenges for complex systems. This dissertation proposed to use Genetic Algorithms to optimize engineering design problems. It proposed a software infrastructure to combine engineering modeling with Genetic algorithms and covered several aspects in engineering design problems. The dissertation suggested a new Genetic Algorithm (Completely dominant Genetic algorithm) to quickly identify High Performance Areas for Engineering Design. To help design engineers to explore design space, the dissertation used a new visualization tool to demonstrate high dimensional Genetic Algorithm results in dynamical graphics. Robustness of design is critical for some of the engineering design applications due to perturbation and manufacturing tolerance. This dissertation demonstrated to use Genetic Algorithms to locate robust design areas and provided a thorough discussion on robustness and diversity in depth.

www.manaraa.com

CHAPTER 1. INTRODUCTION

1.1. Introduction

In a product design process, many complex multiobjective optimization problems occur. For example, in designing an engine controller, appropriate fuel injection times and air-fuel ratios have to be decided to improve engine fuel economy and power performance. But engine fuel economy and power are also affected by hundreds of other engine conditions, such as intake manifold pressure, intake manifold temperature, coolant temperature etc. How to control fuel injection time and air-fuel ratio with respect to these conditions to achieve the optimal fuel economy and power performance is an extremely complex problem. Engineers need to improve the design using simulation and optimization techniques. There are many challenging issues in solving complex engineering problems. The first issue is how to improve the design efficiency. Current industries need to develop high quality products in a short time due to competition or design cycle requirements. Traditional design processes can be much improved by using computational engineering tools. The second issue is how to optimize the complex design. The engineering optimization problems are normally high dimensional and with conflicting objectives. The optimization algorithms need to be introduced to help explore design space and find the optimal solution. The third issue is how to meet robustness requirements. Engineering design always has uncertainties due to manufacturing tolerance and perturbation in real operation. These three issues are the main focus of this dissertation.

Rapid prototyping helps to speed up the design process and explore research and development ideas. Engineers are able to build complex computational models to simulate many physical dynamics, such as combustion dynamics, fluid dynamics, and vibration dynamics. Model accuracy has been improving as we understand more about the system and computational power is enhanced. Industries are able to study prototyping before any manufacture production happens. However, even with the help of computational modeling, the design process is a long and tedious procedure and requires a lot of experiments and simulations to explore the design concept. How to improve design process efficiency is still one of major challenges in current industrial world.

This dissertation proposes some new design methods to help engineers get through the obstacles. Automatic design is to combine optimization tools with modeling

www.manaraa.com

process. Engineers are able to explore and optimize the system during modeling process. Interactive design is to further explore the system using expert knowledge. Designers are able to provide guidelines to optimization process and redefine conditions during optimization. Data management system provides ways to handle data analysis and support automatic and interactive design process. Most importantly, new data visualization methods have been applied to help better understand the system so as to improve the design and reduce the design cycle time.

The second issue that the dissertation is addressed is complex system optimization. As engineering problems become more numerically complex, it is difficult to find a good solution due to constraints on feasible space, natural conflicts between optimization objectives, and lack of understanding of what a good solution is. In the engine example mentioned above, for restricted emission requirements, optimal fuel injection times and air fuel ratios are changing with engine conditions. Fuel economy and power are always two conflicting objectives. Considering that there are numerous other design variables, such as air pressure and injection pressure, affecting engine fuel economy and power performance, the whole design task is complex and requires tremendous design and test efforts.

As engineers face design optimization problems in daily basis, general purpose searching algorithms are needed to assist them to find solutions quickly. The dissertation proposes to use Genetic algorithms (GAs), which are a popular type of searching algorithms. GAs use the evolution idea of survival of the fittest, to do a population based search. With the help of GAs and graphical user friendly interface of GA software, engineers can solve complex optimization problems without fully understanding the system and gain deep knowledge of the system by analyzing GA searching results.

One of difficulties in engineering design and multiobjective optimization is to meet robustness requirement. The dissertation presents a new Genetic Algorithm, which is designed to handle robust optimization problems. The new Genetic Algorithm combining with Clustering algorithm is capable to guide the optimization search to the most robust area. Several examples have been used to prove the new concept.

This dissertation focuses on general multiobjective optimization problems occurring in engineering design. The goal is to speed up the design process, explore complex system design problems better, and meet design robustness requirement. The organization of the dissertation is as follows: Chapter 1 contains the general introduction

www.manaraa.com

of the dissertation. Chapter 2 is literature review in related research areas, and background information about motivation of research. Chapter 3 gives a real industrial design example and presents results generated from Engineering design Genetic Algorithm Software. Chapter 4 discusses uncertainty in optimization problem and proposes a new approach to handle multiobjective optimization in GA. Chapter 5 is the continuous discussion on multiobjective GA algorithm dealing with uncertainty, including diversity and design robustness using several examples. Chapter 6 provides some of my experience regarding to using graphics to show GA results for better understanding. Finally, a brief conclusion and some future potential research areas are given. A description about engineering design software created for applying these techniques to engineering design is in the appendix.

www.manaraa.com

CHAPTER 2. BACKGROUND

2.1. Overview

This chapter provides background information on three different multiobjective optimization areas: Engineering Design Optimization using GAs, a new Genetic Algorithm (CDGA), and robustness in multiobjective optimization. It also provides a literature review of related research areas.

Multiobjective optimization problems have several objectives to be simultaneously optimized and sometimes some of objectives are conflicting. The difficulty in optimizing conflicting multiobjective problems is lack of the global optimum and existence of many local optimal areas as dimension increases. There may be no global optimum for the conflicting multi-objective problems. Considering in vector space, if all elements in a vector are optimal, the vector is considered as the global optimum. But if there is no other ones better than one vector in all dimension. This vector is considered as a non-dominate solution. The optimum in multiobjective optimization is the Pareto Front, which is a set of non-dominant solutions. All non-dominant solutions form the Pareto Front set. There is no guarantees that the Pareto Front set is connected or convex. Fully exploration for the Pareto Front set sometimes is very difficult.

Current multiobjective optimization techniques fall into two categories: combining multiple objectives into one scalar objective, whose solution is just one point in the Pareto Front, and searching the Pareto Front. The first category changes multiobjective problems into single objective problems so that all traditional optimization methods can be applied to, such as gradient methods and simulate annealing. The disadvantage of changing to single objective problems is that the optimal solution is only the solution that is designed to be searched. The whole Pareto Front set is not explored. The second category is trying to explore the full Pareto Front set. Many traditional optimization methods are hard to apply for this kind of optimization. Heuristic search methods are the main techniques used for searching for the Pareto Front because they do not require mathematical descriptions of optimization problems and are guaranteed to find good solutions in a reasonable time. The disadvantage of heuristic search is that it might not always find the best solutions and the search is time consuming.

www.manaraa.com

The dissertation is inspired by the multiobjective optimization problems met in the real industry design. It intends to help engineers to solve multiobjective optimization problems with robustness requirements using genetic algorithms.

2.2 Dynamics and control system

System dynamics are the time series responses of a system to a set of inputs. The dynamics of the system can be viewed as a time-dependent function of the set of inputs, but the function is hard to be defined for a complex system. Dynamics are normally described using high dimensional differential equations, which can be modeled in simulation. But simulation has to change the continuous dynamical system to a discrete time system in the digital world. The simulation result is very sensitive to the simulation time step. Generally, the smaller the time step is, the closer the result is to the real value. System response errors can be controlled by simulation time step and integration algorithms. If differential equations describe system dynamics as accurate as real dynamics, simulation response can be modeled very close to real response. Dynamics in many of complex system such as vehicle dynamics and fluid dynamics have been simulated using sophisticated computer models.

Modeling uses a simplified representation of a system to enhance our ability to understand, predict, and control the behavior of the system [74]. Modeling is an important process in developing new industrial products. Thanks to the powerful modeling software, engineers are able to set up dynamic models for complicated systems very quickly when they have understand system dynamics.

The design process involves modeling, simulation, and evaluation. According to Roosenburg and Eekels [95], the design process is iterative and consists of analysis, synthesis, simulation, evaluation and decision. It is rare that the simulation of the first design will meet the expected properties. Designers have to adjust system parameters, even change system design to meet the performance criteria. It is viewed as a tuning process in controller design [12]. The process defined by Roosenburg and Eekels is shown in figure 2.1. The process can be viewed as an optimization process as stated by Simon [99].

www.manaraa.com

2.3. Optimization

Optimization finds the minimum or maximum value for a function and its location, while design problems need to meet some performance criteria. It is always possible to change a design problem into an optimization problem. Designing system parameters is changed into finding the location in the input space that optimizes the system.

Optimization problems can be formalized as the follows: Vector $\mathbf{x} \in X$ where X is a subspace of R^n, objective function $\boldsymbol{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \in R^k$. The goal is to minimize $\boldsymbol{f}(\mathbf{x})$ with $\mathbf{x} \in X$ subject to some conditions $\boldsymbol{G}(\mathbf{x})>0$. For high dimensional multiobjective problems, k and n are larger than 1. (Note that minimizing $\boldsymbol{f}(\mathbf{x})$ is equivalent to maximizing $-\boldsymbol{f}(\mathbf{x})$). The conditions $\boldsymbol{G}(\mathbf{x})>0$ are constraints.

If $\boldsymbol{f}(\mathbf{x})$ is a continuous function, according to Newton's theory, the minimum occurs either at the boundary or where $\partial \boldsymbol{f}(\mathbf{x})/\partial x = 0$. In order to solve the problem based on Newton's theory, we need to solve equation $\nabla f(x_1,...,x_n) = 0$ and find all singular points for $\boldsymbol{f}(\mathbf{x})$. The nonlinear equation $\nabla f(x_1,...,x_n) = 0$ is normally not easy to solve.

To avoid solving difficult nonlinear equations and calculating the second derivative to find out whether a point is a local minimum, local maximum, or saddle point, many optimization search algorithms have been developed. If optimization algorithms calculated the gradient, they are called gradient-based search algorithms such as steepest descent and conjugate gradient [52, 86]. The basic idea is the search begins with a random start point. At each iteration step, the search will move in the direction with the largest decrease in the value of $\boldsymbol{f}(\mathbf{x})$, which is the direction of directional derivative has the greatest value. The steepest descent method is defined as the following formula.

$$x^{k+1} = x^k - \alpha^k \nabla F^k$$

x^{k+1}, x^k = values of the variables in the k and k+1 iteration

F(x) = objective function to be minimized (or maximized)

∇ F = gradients of the objective function

α^k = the size of the step in the direction of travel

The steepest descent method is known for its simplicity but seldom converges reliably.

www.manaraa.com

It is well known that the gradient algorithms tend to get stuck in local optima. There are many variations on how to control the step size to avoid being stuck in local optimum. In practice, the gradient is often hard to compute. Newton's gradient optimization methods require the calculation of not only the first derivative, but also the inverse Hessian. Conjugate gradient methods are invented for solving the quadratic problem: minimizing (½) $x^TQx - b^Tx$. For non-quadratic problems, it is hard to approximate Q. In dynamical modeling, there are no clear mathematical equations defining the relation between output and input. To use gradient based algorithms to optimize system, the gradient of each parameter at each state has to be calculated through simulation. The calculation cost will increase exponentially as the number of parameters increases. In real engineering optimization, systems are normally nonlinear and have many complex nonlinear phenomena, such as bifurcation and chaos. In addition, many design problems involve curve design. A curve is a look up table defining a function of two variables. For example, hydraulic systems often have curves for valves that define the relationship between flow or pressure and the position of the valve. In real industry design, curves are often converted to finite dimension design variables using interpolation or curve parameterization. Two close curves will sometimes result in significantly different response. Therefore, objective functions in engineering optimization problems are often not smooth, sometimes even not continuous with respect to the curve. It is difficult to apply gradient optimization methods to problems with non-smooth objective functions.

Another approach is using stochastic search. If there is no limit on execute time and cost, the best solution can always be found through a complete search. Due to the curse of dimensionality, the search space increases exponentially with dimensionality. Modern heuristic search algorithms are based on the assumption that good solutions are more probably close to other known solutions than randomly picked solutions. The basic idea of heuristic search algorithms is only searching paths that tend to lead to the goal rather than searching the whole space. By minimizing searching space, heuristic search algorithms can find solutions much quicker than random searching. For any heuristic algorithm, it needs an evaluation function to decide how good the path is. This evaluation will decide what the next search path at the next iteration is.

www.manaraa.com

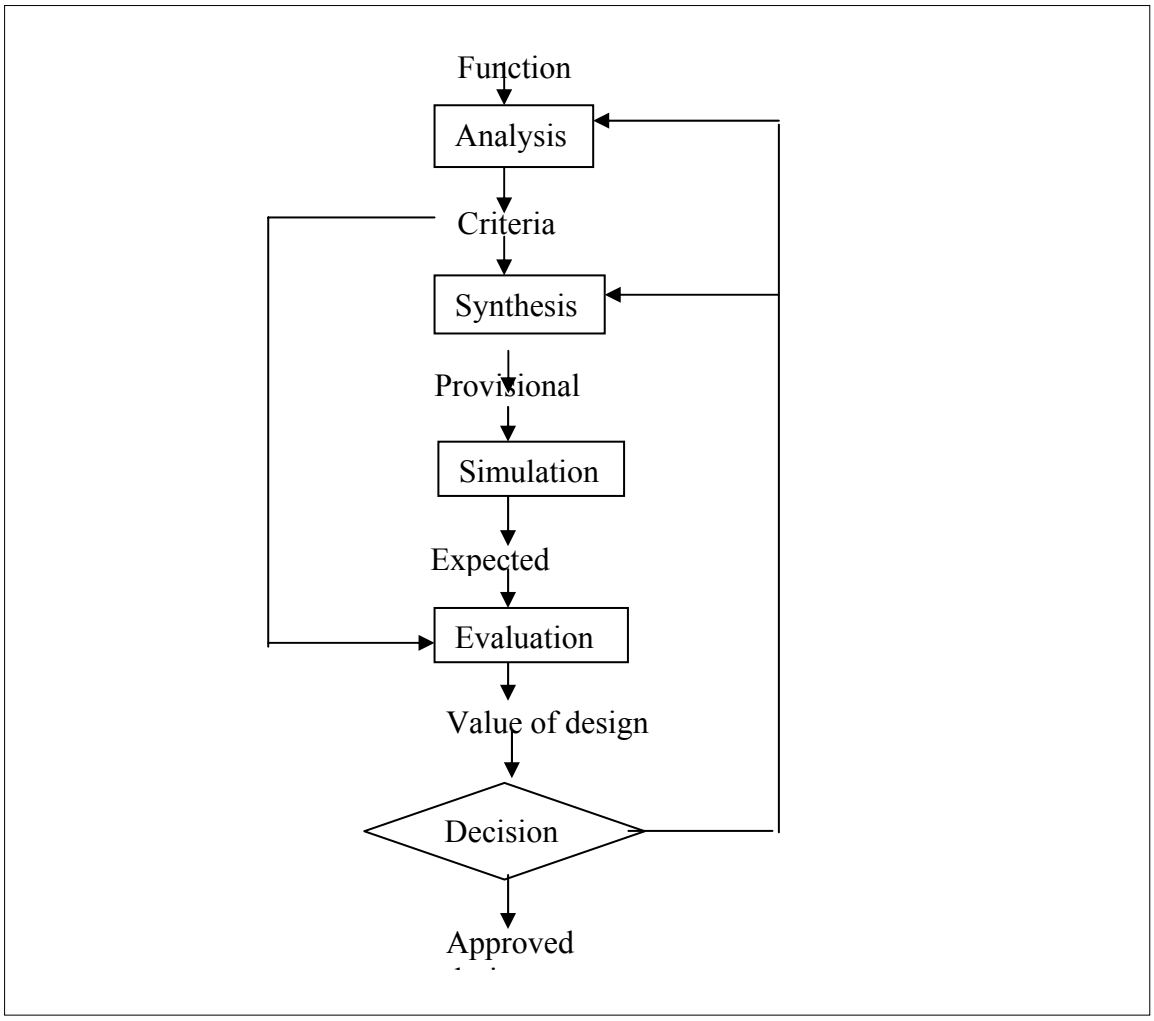


Figure 2.1 Design Process

2.4. Heuristic search algorithms

2.4.1. Hill climbing algorithm

There are many different heuristic search algorithms. One of most common earlier used algorithms is hill climbing. The basic strategy is to evaluate all possible paths and choose the best one (analogous to climbing a hill). The well-known disadvantages of hill climbing algorithm are: 1. If it starts at a foothill, it is not likely to find the hill summit. 2. If the plane is flat, hill climbing algorithm has no clue which direction it should go. 3. If the search reaches a local top, it has to go down to find the global one.

The advantage of hill climbing is its simplicity and its easy implementation. It has no requirement on optimization functions and pre-knowledge of the problem. The only thing it needs is an evaluation function to evaluate each generated solution. Due to its

www.manaraa.com

simplicity and generalization, it has shown great performance on some simple optimization problems.

One way to partially alleviate the pitfall of being stuck in local optima is to use multi-start hill climbing [60], which increases the probability to find the global optimum. However, the time and cost can be tremendous compared to other search algorithms. Therefore, hill-climbing algorithms are best suited for unimodal optimization problems.

2.4.2. Simulated annealing

Simulated annealing is inspired by the physical cooling process of metal materials. The molten metal has to be cooled slowly and evenly to prevent from cracking. Borrowing the same idea for optimization, simulate annealing enhances neighboring search by allowing occasional long moves to prevent from getting stuck in the local minima [21, 61, 63].

In the first stage of the algorithm, the parameters vary over a wide range. As the algorithm goes on, the search space becomes smaller and the final solution is hopefully be settled into the global optimal solution. The probability of accepting solution j from solution i at the k_{th} step is:

$$p(j,i)_k = \begin{cases} 1 & if\ f(j) \le f(i) \\ e^{\frac{f(i)-f(j)}{c_k}} & if\ f(j) > f(i) \end{cases}$$

c_k is the cooling schedule and normally decreases to close to zero as simulation is going on. Therefore, at the first stage, the algorithm basically allows any direction of search. At the final stage, as c_k is close to zero, the search will only towards the better solution. The search result is dependent on the cooling schedule.

Simulated Annealing sometimes is very slow, even though it has been proven to converge to the optimal solution if the right cooling schedule is used [61]. For high dimensional optimization problems, it often is stuck in local optimal point. A general cooling schedule with guaranteed convergerce for all optimization problems has not been found. Despite this, simulated annealing has been widely applied on a variety of areas, including scheduling, and network routing according to [61, 63].

www.manaraa.com

2.4.3. Evolutionary algorithms

Evolutionary algorithms use biological concepts to solve optimization problems by emulating evolutionary processes. The idea first came up as early as in the 1950s with limited applications [28]. In the 1970s, as computing improved, it attracted more interests from varieties of scientists and engineers.

Evolutionary algorithms have a variety of derivations. They share the same strategy:

Create an initial population
Evaluate solutions in the population
Repeat
    Select solutions to produce offspring
    Produce new solutions by copy and variation
    Evaluated new solutions and put them into the population
Until Done

Evolutionary algorithms first create an initial population of data structures. The data structure can be varied depending on algorithms. For example, binary genetic algorithms use fixed length binary numbers as their basic data structure. The data structure, which contains certain information, is called as chromosome in genetic algorithms. Genetic programming uses a parse tree as its data structure. It has various data length as the parse tree is changing. Evaluation normally uses a fitness function to compare solutions and may affect what parents are chosen to produce children for the next generation. In producing new solutions, two variation methods, mutation and crossover, are generally applied to combine parents' data structures to produce children's. Crossover exchanges parents' data structures so that children's data structures share some of parents' characteristics. Mutation changes part of the children's data structures in order to bring variations in the children's data structures. The next step is to select among the parents' children's population and to form a new population of solutions for next iteration. The main loop is iterated until the stop condition is met.

Compared with simulated annealing, evolutionary algorithms use population-based search instead of one-way search. It has more chance to skip local optimum. What's more, different search paths can exchange their information so as to speed up search process. The mutation and crossover operators are much easier to set up than the proper cooling schedule. Although evolutionary algorithms are computationally

www.manaraa.com

expensive, it is a good geneal algorithm to solve complex optimization problem, especially with multiple local optima.

Evolutionary algorithms evolve over time to find the solution digitally. Each unique solution has its own data structures containing its own information. If the data structure is not varied between solutions, it is called fixed data structure evolutionary algorithm. Otherwise, it is non-fixed data structure evolutionary algorithm. There are many common used evolutionary algorithms, including genetic algorithm, finite state machines, and genetic programming. Each of them has wide applications covering different areas [41, 44].

2.4.4. Genetic Algorithm

One of the most common evolutionary algorithms is the genetic algorithm. Genetic algorithms are based on the mechanism of natural selection. They follow the standard iteration steps as evolutionary algorithms. They use binary or floating genes to represent design variables with fixed length. At each iteration, they use pairs of two genes with high fitness to generate new genes by crossover and mutation. The next population is selected in parent and children genes according to fitness.

When genetic algorithm first came up in the late 1970s, it used binary gene representation in most cases. Genes are defined, in biology, as a sequence of DNA that represents certain characteristics. The 1 and 0 sequence in the GA gene represents a unique solution. Each of binary numbers is called one chromosome of the gene as called in biology. Mutation and Crossover are both used in Genetic algorithm. In GAs, crossover is just exchanging genes at certain crossover point. Figure 2.2 illustrates a crossover example with gene length being 5. The crossover position is the end of the second chromosome. Crossover has various types. The most common ones are one-point crossover and two-point crossover, i.e. the crossover happens at one position or two positions respectively. Mutation flips the binary bit at certain position as shown figure 2.3. Generally it assigns a small probability rate for mutation at each position.

www.manaraa.com

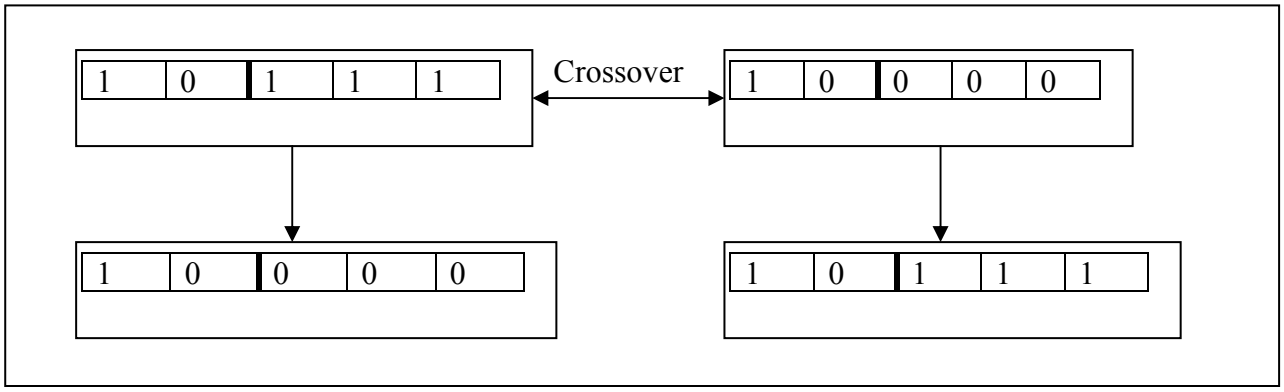


Figure 2.2 One Point Crossover Example

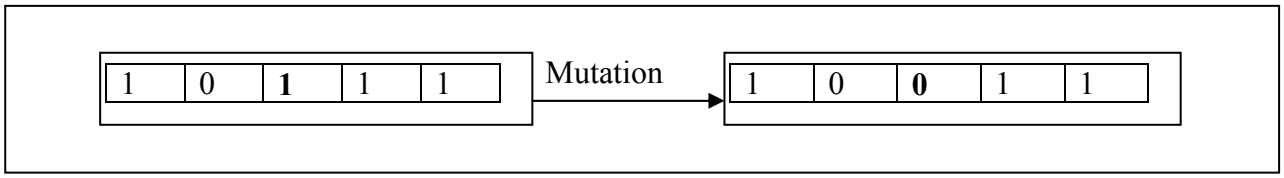


Figure 2.3 Mutation Example

Selection keeps the population size stable. It inserts some of the children into the population to replace old ones. The good solutions have large possibility to survive than the bad solutions. The selection pressure makes solutions tend to improve along the evolution process. Generally, there is a selection rule to compare solutions, for example, a fitness function which is a function of optimization objectives. Solutions with high fitness have more chance to survive in the selection process.

There are many selection methods to evolve the population that can be divided into two categories: elite and non-elite. Elite selection methods assure that the best individuals of the population go to the next population. Elitism favorites individuals with the best fitness and makes them to produce more children. Some of non-elite selection methods are roulette wheel selection and tournament selection. Roulette selection assigns each individual gene a probability, which is in direct proportional to its fitness. The individual with high fitness has high probability to be picked. Tournament selection shuffles the population randomly and divides them into small groups. At each iteration, half the population with better fitness will survive.

Each iteration in a GA is called a generation since some individuals disappear in the population and new individuals appear. The population size stays stable through

www.manaraa.com

generations. The number of generation affects running time of GA and affects its ability to locate global optimum.

Another commonly used data representation is real-value gene. Although real value number can be changed into the binary format, which may result in resolution lost, real valued GAs are intuitively suitable for engineer problems. Real valued representation keeps each design variable as a unique chromosome so that crossover does not happen in the middle of one design variable as using binary representation. Crossover in the real value GA is almost the same as binary GA while mutation is quite different. Real valued GAs normally use one point mutation, i.e., only mutating one design variable each time. The value of the chosen mutating point changes in a certain range.

2.5. Multiobjective Optimization

As stated above, optimization problems are described as optimizing $\boldsymbol{f(x)} = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_n(\mathbf{x}))$. Assuming minimization, if $\boldsymbol{f(x)}$ is a scalar value, the optimization goal is to minimize this value. But if $\boldsymbol{f(x)}$ is a vector, i.e., it is a multiobjective optimization problem. The optimization goal is to minimize all the objectives, $f_k(\boldsymbol{x})$, simultaneously.

If an optimization problem has only one objective to minimize, many optimization methods can be used to minimize the objective, for example, hill-climbing and simulated annealing mentioned above. However, if it has multiple objectives, it is sometimes not possible to find an optimal solution with respect to all objectives. Figure 2.4 shows a two objective optimization example that has no global optimum. Objectives f_1 and f_2 have a feasible area due to limitation on inputs and function characteristics (left figure). There is no one global optimum for this example because it is not possible to achieve minimal f_1 and minimal f_2 at the same time. The optimum for multiobjective problem has new definition to deal with conflicting objectives.

If all objectives of solution A are smaller than ones of solution B, A is considered as dominating B. For example, a vector (3, 4) dominates vector (4, 6) but does not dominate vector (2, 10) according to domination definition above. If a solution cannot be dominated by all other solutions, it is considered to be a Pareto optimal solution. All Pareto optimal solutions formulate a Pareto optimal front. Figure 2.4 shows the Pareto optimal front for the example (right figure).

In multiobjective optimization problems, any solution on the Pareto optimal front is an optimal solution. The multiobjective optimum is the Pareto optimal front. The task is

www.manaraa.com

changed to find the Pareto optimal front, which is normally a high dimensional area. Then the decision which solution is the best is taken with respect to other criteria such as robustness and cost. Human (decision maker) need to be involved in this selection process.

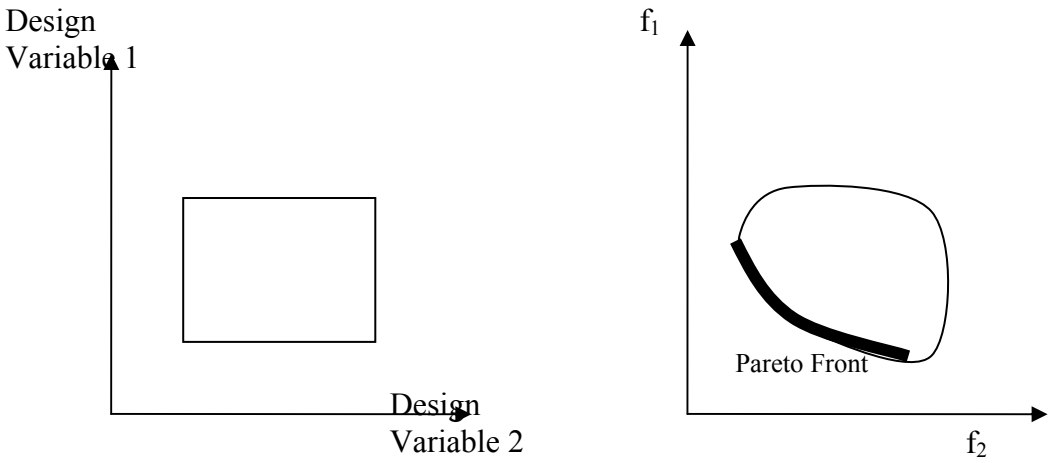


Figure 2.4 Multiobjective Optimization Problem Example

To solve multiobjective optimization problems, there are many different algorithms. Generally they can fall into three basic approaches: 1. aggregating method: transfer the multiobjective into a single objective; 2. criteria method: optimize one objective at one time; 3. Pareto method: Use the Pareto optimal idea to find Pareto optimal front then select the final solution. For a multiobjective problem, if a mapping from all objectives to a fitness function is constructed, then the multiobjective problem are changed to single objective optimization problem. The mapping can be generally represented as follows:

$$\boldsymbol{f(x)} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})) \rightarrow g(\mathbf{x})$$

There are several different methods for forming multiobjective functions such as weighted sum and fuzzy logic fitness. This single objective function can be optimized using many optimization algorithms such as the algorithms mentioned. The disadvantage of the approach is that it only finds one solution not the whole Pareto optimal front. Since the solution found is directly decided by the mapping, the decision maker has to know which direction search should go before the search. If the decision maker knows how to trade-off all objectives, it is suitable and very efficient to use this method. For example, if engine designers want to optimize engine's emission and fuel economy simultaneously,

www.manaraa.com

one way to trade-off two objectives is to choose the best fuel economy with meeting emission requirement.

Criteria methods can only minimize one objective for each search process. However, the idea can be borrowed to use in evolutionary algorithms. An individual with one minimal objective is at the Pareto front. It contains some information to help EA to locate the overall Pareto front. A practical application is to start multiple search processes, each of which is attending to optimize one objective. Individuals among different processes can exchange information by crossover. The interactive GA that Parmee suggested is using such techniques.

For some multiobjective problems, it is hard to choose a trade-off from the Pareto optimal front. The Pareto front needs to be further explored to make the decision. In the engineering design, typically designers have no idea what value of each objective can be gotten through the design. For example, in designing diesel engines, there are two emission limits: NOx and Particulate Molecular (PM) to be met. Before the design, designers have no idea how low NOx and PM can be reached. The objective set must be explored to find a design that can meet emission standard. In this situation, the whole Pareto front should be found in optimization process. Population based search algorithms are more suitable for this task.

2.5.1. Multiobjective GA

GAs have been extensively applied to multiobjective optimization problems since they can locate the Pareto front. Traditional GAs use a fitness function to evaluate solutions. This is only suitable for a single objective problem. For multiobjective problems, this method results in converging to single point on the Pareto Front. Instead of using fitness functions, another evaluation method such as Pareto dominance, which is able to to compare solutions for GA selection, is needed. Pareto dominance is clearly a right choice.

Pareto dominance is used to compare two solutions. If a solution is dominant to the other, it is a better one. Selection favors the dominant solution and makes it produce more children. Goldberg [46] has suggested a ranking method for population comparison. In his formulation, at each iteration, the population is searched for nondominant solutions. These solutions are ranked as 0 and are removed from the population. Then another set of nondominant solutions are found in the remaining population. They are ranked as 1

www.manaraa.com

and also removed from the population. The process is repeated until all solutions have been ranked.

There are some other variations of ranking methods. Goldberg's ranking is divided the population into many layers of Pareto front. In Multiobjective Genetic Algorithm (MOGA) ranking method [43], each individual's ranking is determined by the number of individuals by which it is dominated. The global Pareto front has the same rank 0 as in Goldberg's method. But the rest has quite different rankings. There are many multiobjective genetic algorithms such NPGA II [31], SPEA II [113] with variations in selection methods and diversity control techniques.

Any population based ranking method takes lots of computation time, especially for a large population. But GAs need to use large population sizes to find the whole Pareto front. Otherwise, the population will be filled with all non-dominant solutions. To make the search more efficient, tournament selection methods can be used for multiobjective GAs. Local dominance ranking is only needed for tournament selection. However, tournament group size becomes an important factor to affect selection process, which complicates its application in practical use.

2.6. Diversity

One of the major problems in evolutionary algorithms (EAs) is that simple EAs tend to converge to local optima. If there are several local optima in a single objective problem, EAs sometimes will be stuck in a locally optimal solution. In multiobjective problems, the Pareto optimal front can be very large. EAs have a tendency to converge to local areas without covering the whole optimal front space. For both single objective optimization and multiobjective problems, diversity is important to prevent from being trapped in local optima.

Diversity and convergence are two conflicting factors in any evolutionary algorithms. If a high selection pressure is applied, individuals will quickly be replaced by better fit ones and diversity will decline in a short while. If a low selection pressure is applied, EAs will take too long to converge. Several studies have been carried out for keeping diversity in the population while allowing rapid convergence. Bosman and Thierens [16] state that the existing best MOGAs behave similarly or individually preferable by different diversity metrics (performance indicator), i.e. Most of MOGAs'

www.manaraa.com

performance is problem dependent and may perform better for some problems due to its unique diversity metrics.

There are several techniques reported to avoid premature convergence for EAs, such as crowding [16] and random immigrants [17]. Crowding techniques create offspring to replace existing individuals based on their similarities [19]. Random immigrants bring some entirely new randomly generated elements into the gene pool [18]. But the most common one used in GAs is sharing. Sharing distributes non-dominate genes over a number of peaks on its Optimal Pareto front. At each iteration, it calculates a sharing fitness function, which is related to objectives and how crowded of neighborhood, for each individual and picks only a fraction of the population around each peak in proportion to height of the peak.

Sharing can be performed in objective space or decision parameter space (input space). As from the Pareto dominance definition, the Pareto front is defined in objective space. Designers like to see the Pareto front well distributed so that they can choose a good solution meeting their requirements. However, from a GA' stand point, diversity in the input space is important. If the population is filled with a lot of similar solutions, it has a strong tendency to be trapped in a local optimum. Input space diversity affects objective space diversity, but the opposite is not sufficient. Horn [53, 54] has suggested a sharing technique that combine both spaces called nested sharing. But the way to balance two diversity requirements still lacks general procedures.

The current state-of-the-art evolutionary algorithms in multiobjective evolutionary optimization which include the Nondominant Sorting Genetic Algorithm II(NSGA-II) by Deb *et al.* [30], the Strength Pareto Evolutionary Algorithm(SPEA) by Zitzler and Thiele [114], the SPEA-II by Zitzler *et al.* [113], the *Pareto Archived Evolution Strategy* (PAES) by Knowles and Corne [55], have presented different ways to handle diversity. The basic idea to keep dominant solutions spreading out is all the same, but each algorithm uses different selection and elitism approaches. They are considered as "*a Pareto set of MOEAs*" because each of them has been proved to have good performance on certain problems.

2.6.1 Niched sharing example

A niche represents a competition for limited resources. In multiobjective GA, population size is fixed and it is outnumbered by feasible solutions in the Pareto front. To

www.manaraa.com

make sure the final population covers the Pareto front uniformly, niching has to be applied to maintain high quality diversity.

MOGA [46] has one of the most common used niching sharing scheme. It is a fitness sharing method, which degrades the fitness according to the number of similar (in input space sense) individuals. Each individual i has its objective fitness $f_i = f(i)$ that is calculated from its Pareto ranking. Designers need to define a sharing distance σ_{share}. It is a fixed radius threshold for similarity. If two individuals *i, j* have larger distance *d(i,j)* than σ_{share}, they won't affect their sharing fitness each other. A sharing value is defined as:

$$sh(d(i,j)) = \begin{cases} 1 - (\frac{d(i,j)}{\sigma_{share}})^k & d(i,j) < \sigma_{share} \\ 0 & otherwise \end{cases}$$

where k is a real number determining sharing function shape and is often set to one.
For an individual i, its niche count m_i is the sum of its neighboring sharing values:

$$m_i = \sum_{j=1}^{N} sh(d(i,j))$$

where N is the size of the population. The adjusted fitness (shared fitness) of an individual *i* is then given by

$$f_{sh}(i) = \frac{f_i}{m_i}$$

σ_{share} is a very important factor in above niche sharing scheme. It not only affects each individual's shared fitness, but also affects diversity around the Pareto front. Choosing an appropriate σ_{share} is an optimization problem itself, which makes it hard to apply for general multiobjective optimization.

2.7. Uncertainty

Real world problems always involve uncertainty. It may come from modeling uncertainty and parameter estimation. For example, in designing a vehicle, the weight of vehicle varies in different conditions, such as fully loaded and half loaded. It will bring uncertainty for vehicle modeling. Optimizing vehicle design has to be done across all the conditions. If a design variable has a linear relationship with objectives, optimization can

www.manaraa.com

be done in its extreme conditions. But if the relationship is nonlinear, it is hard to estimate at what point the variable affects objectives most.

There are two categories of uncertainty problem in evolution algorithm:

1. Two successive evaluations of one chromosome return two different sets of objectives.
2. Two successive evaluations of one chromosome return the same set of objectives. But the objectives of any chromosome are not accurate and can be varied in a certain range.

The first category means that objectives have certain disturbance and have a probability distribution at a certain range. Normally, these uncertainty problems are dealed with by using objective's mean and variance. The second category means chromosomes have certain disturbance, which results in disturbance in objective functions.

In engineering design problems, the first case seldom happens. Because when one combination of design variables is selected, one physical system is determined and system outputs are fixed as well. Only if a system has random signal source, for example, one of control variables has a probability distribution, then the system has different responses for the same design. Most engineering problems fall into the second case. If the system are not modeled accurately or any design parameter is estimated, inaccuracy will happen in calculating objectives, which results in drift in objectives.

This kind of uncertainty makes it difficulty to compare two sets of objectives in evolutionary algorithms. Comparisons show that one set of objectives dominates another one, but actually it may not better. Several studies have been carried out with comparing two uncertain fitness measurements [47, 48]. Objective measurements are treated as values with distribution probability. The most common distributions such as uniform and normal distribution have been studied. The difficult thing is to know the exact distributions of uncertainty.

In Jin's survey [50], uncertainties are categorized into four areas: noise, robustness, fitness approximation, time-variant fitness functions. In Engineering design problems, the second area - robustness is the essential problem we care about. Only a few studies have been done to study robustness in multiobjective optimization. Deb [27] suggested that robustness can be achieved by optimizing mean effective fitness function, which is the average of a set of neighboring solutions. This method will significantly increase computation times due to calculating neighboring objective functions, which requires running a lot of additional simulation in dynamical modeling and is very time consuming. Ray [92] suggested adding robustness to objective functions and it samples a

www.manaraa.com

set of neighboring solutions to get the mean and standard deviation, which are added into objective functions. This method still uses fitness concept and make it difficult to apply to multiobjective optimization problems. Some other research areas dealing with uncertainty such as approximate fitness function and dynamic optimization are targeted to dynamic fitness function (time variant) problems, which is not the focus of this dissertation.

Robust design means system meets requirement in worst case (disturbance). Robustness is sometimes conflicting with optimization. To meet consumers' need, designers want to achieve the highest possible optimum with meeting robustness requirement. Little research has been conducted to study robust engineering design problems using GA. GAs are thought to be capable of finding the robust designs. But this is arguable because diversity techniques affect GA converging to robust areas. Moreover, the robust areas GA found are not always able to meet designers' requirement.

2.8 Genetic Algorithms in Engineering Design

One of the popular heuristic search algorithms is genetic algorithm. GA not only has all heuristic algorithm's characteristics, but also is a multi-directioned search method. It originally is designed for single objective optimization problem since it uses a fitness to do evaluation. As GAs are applied to multiobjective optimization, the fitness concept has been extended to dominance rank, which is created for searching the Pareto Front. Since then, GAs begin to become popular in multiobjective optimization areas, especially in finding the Pareto Front.

GAs use dominance rank to push the population close to the Pareto Front and it has been proved to be an effective way to explore the Pareto Front. One of the difficulties in exploring the Pareto Front is the curse of dimension. As dimension of the problem increases, the Pareto Front becomes very complicated. GAs tend to be stuck in some local Pareto Front areas. To make the optimal solutions well covering the Pareto Front and quickly converting to the optimum is what most of research on GAs are focusing on. How to balance proximity and diversity in exploration is a multiobjective optimization problem itself.

There are many studies on diversity preservation, diversity estimation, and metric comparison to improve the population diversity. Many different GAs have been presented to improve diversity and convergence such as RAND, FFGA, NPGA, HLGA, VEGA, NSGA listed in Ziltzler's paper. There are new research ideas such as co-evolutionary

www.manaraa.com

GAs and Clustered Oriented GAs. No matter what GAs are, they are trying to make the search converge to the Pareto Front (or close to the Pareto Front) as quick as possible and make the population cover the Pareto Front (or close to the Pareto Front) as even as possible at the least computation time. Since these goals are conflicting themselves, we often find out that any GA is a tradeoff of these goals and it may perform well on some certain problems but bad on others.

Chapter 4 suggests a new Genetic Algorithm – Complete Dominant GA. It loosens the dominance concept and allows a tolerance in comparing two solutions. Its idea is very simple and can be easily combined with any existing GAs. The advantage is its simplicity doesn't require any additional computation but preserve diversity in some degree and converge to the Pareto Front very well. The most important characteristic is that it can be easily to use in searching for robustness.

2.8.1. Engineering design using GAs

Since GAs have shown excellent performance in optimization problems, especially in multiobjective optimization, engineering design optimization problems have been explored with GAs. Engineering design optimization problems normally are multi-objective problems with high dimensional design variables. They also are complicated in that system dynamics are always non-linear and with uncertainty. In addition, engineering design problems often have constraints on design variables. All these issues have been well addressed in different GAs.

As engineering design becomes more and more complex in modern industry, computer modeling is one of the essential method to achieve reducing design cycle and improve design quality. Genetic algorithms have been used in a lot of complex design problems. There have been a number of activities from developing GA software for engineering design to improving GAs for engineering design.

2.8.2 Robustness in engineering design

Robustness is key to designing products that work in a range of condictions. From this aspect, robustness is sometimes in higher priority than optimuality. Engineering design has to deal with uncertain environment, manufacturing tolerance and un-modeled effects. Real industry problems have shown that uncertainty can result in failure in the field.

www.manaraa.com

Previous researches focused on uncertain objective function problems, i.e., objective functions will return different values with the same design inputs. The techniques used for these problems are to estimate distribution of objective functions. These methods have been used to apply to mathematical problems to deal with uncertainty. However, this method is limited in that engineering design has to deal with uncertain design variables, especially curves. Few researches are oriented for this area at present.

Chapter 4 proposes a new Complete Dominant Genetic Algorithm to help solve robustness problem in engineering design. It is an innovative way to explore robustness problem. CDGA will push the GA search to high performance region instead of the Pareto Front, so robustness of each solution in high performance region can be explored by the help of clustering algorithm. Chapter 5 uses sevearal multiobjective problems to present the whole idea and show the robust solution it has found. Further discussion on diversity and robustness is provided in details as well.

www.manaraa.com

CHAPTER 3. HIGH DIMENSIONAL SYSTEM DESIGN USING GENETIC ALGORITHMS & VISUALIZATION

Published in 2003 American Control Conference

Xiaopeng Fang[1], Brian Kellogg[2], Tye Conlan[2], Julie Dickerson[1], Di Cook[3]
[1]Department of Electrical Engineering
Iowa State University, Ames IA 50010
[2]JohnDeere Corporation, Dubuque IA 52001
[3]Department of Statistics, Iowa State University, Ames, IA 50010

Abstract

This paper uses genetic algorithms (GA) to explore and optimize a high dimensional multiobjective system for brake control. The design goal is to make a hydraulic brake system efficient and comfortable for a variety of vehicles. High dimensional visualization has been used to visualize the design space and design the fitness function. The effectiveness of methods is demonstrated by the brake design example.

Keywords: genetic algorithm; data visualization; high dimensional optimization; dynamical modeling;

Introduction

Dynamical modeling provides engineers an accurate way to simulate dynamics of complicated systems. The design process often includes design, evaluate, and redesign cycles. It can require many repetitions to select a design that performs well under many conditions and is feasible to be built. This problem is particularly acute for hydraulic systems. Valve performance is often specified using area curves that mathematically define an ideal transfer function between the valve input and output flow. In practice, these curves cannot be achieved exactly and tolerances must be estimated. The designer must also be sure that the control space has been explored thoroughly for different input conditions to ensure that there are no regions of unexpected behavior. Intelligent and efficient methods to help discover the optimal design and reduce design time are needed.

www.manaraa.com

Evolutionary algorithms have been shown to be robust and efficient in finding the global minimum *(1)*. Genetic algorithms have been applied to many real-world multiobjective problems including control engineering design *(10)*, industrial design *(11)*, and transportation planning (*12)*.

Genetic algorithms normally decide a design whether good or not based on a fitness function. In practical design problems, there are often several objectives to be satisfied. Normally, there is no complete optimal solution to minimize or maximize all the objectives. Instead, people try to find Pareto optimal solutions, a set of solutions that cannot improve any objective function without sacrificing at least one of the other objective functions *(6)*, and use their own expert knowledge and experience to choose.

It is difficult to select a suitable fitness function to guide the genetic algorithm. There is often no analytical method available to explore the input and output space besides looking at simulation results. High dimensional data visualization methods such as parallel coordinate plots, tours, and linked plots, available in statistical packages for example GGobi, can assist in examining the space *(8)*. Designers are able to pick a good solution from the Pareto space with the help of data visualization.

The system structure is shown in figure 1. Engineers build complicated dynamics models for their problems of interest using a dynamical modeling tool such as EASY5 *(13)* and Simulink *(14)*. A database is used to store and manage simulation results. GA tools with user interface support are connected with the simulation model and database directly. Database, GA tools, and visualization are all independent from the simulation and can be coupled with other modeling software. Decision-makers use the GA and visualization to explore the design space and choose the optimal solution. A hydraulic system design project has been used to test the whole system.

www.manaraa.com

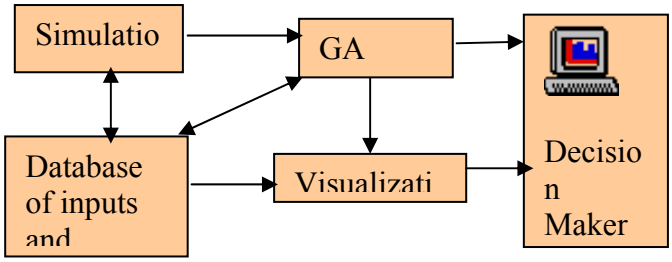
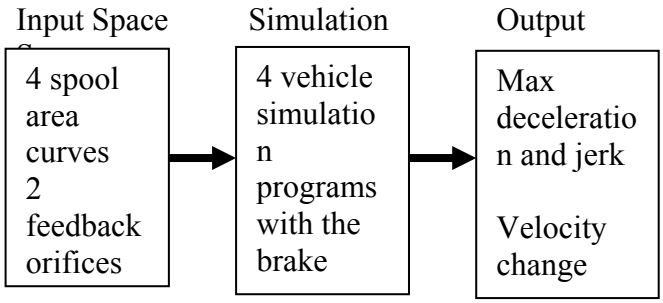


Figure 1. System structure



Background

This project uses genetic algorithms to assist design engineers in finding optimal designs for a hydraulic brake model. The hydraulic brake design project studies what combinations of inputs (such as supply pressure and area curves) result in a comfortable and efficient brake system that works on a variety of machine models. For this goal, desired response of the brake system has its max deceleration smaller than 0.2~0.3g, its max jerk smaller than 1g/s *(9)*, and its velocity change because of deceleration as large as possible. System diagram is shown in figure 2.

The brake system model is shown in figure 3. Brake valve area tables coupled with pressure drop across the valve control the flow rate through the brake valve. The brake valve has two spools. Each spool has its own two area tables (in/out brake) and control flow and pressure to either the front and rear axle. The brake valve feedback orifice for each spool controls the brake valve's response to pressure in the brakes. The supply pressure specifies the system inlet pressure to the accumulators and the brake valves.

Genetic Algorithms

Genetics algorithms are optimization techniques inspired from evolution. Based on the survival of the fittest strategy, GAs exploit the best solution and explore the search

www.manaraa.com

space through genetic operators: selection, mutation, and crossover *(3)(4).* They have been successfully applied to the optimization area *(5).*

The genetic algorithm used in this work uses floating point representation, since floating point representation is more suitable for multidimensional, high-precision numerical problems *(5).* The genetic operators and parameters for the brake system are shown in table 1.

GA type	Floating point
Initial Population	16
Mutation	Non-uniform
Crossover	One point
Selection	Tournament
Termination	40 generations

Table 1. Genetic operators and parameters

Gene:

There are four spool area curves (SP1P2B, SP1B2T, SP2P2B, and SP2B2T) for the hydraulic Brake model, each of which controls the flow rate through the brake valve. Each area curve is represented by one scalar factor. The other design variables are the size of the two orifices in millimeters and the supply pressure in kPa. Each individual has a gene with seven variables. Each variable is a real value number representing the scalar of the original area curve or the actual value for orifice and pressure. For example, the gene (3.5 2 1 3.5 1 1.5 4900) means the first area curve (SP1P2B) is 250% larger than the original SP1P2B curve (as in figure 4) and 100%, 0%, 250% larger for other area curves respectively. The first orifice size is 1.0 mm, and the second one is 1.5 mm. The supply pressure is 4900 kPa.

www.manaraa.com

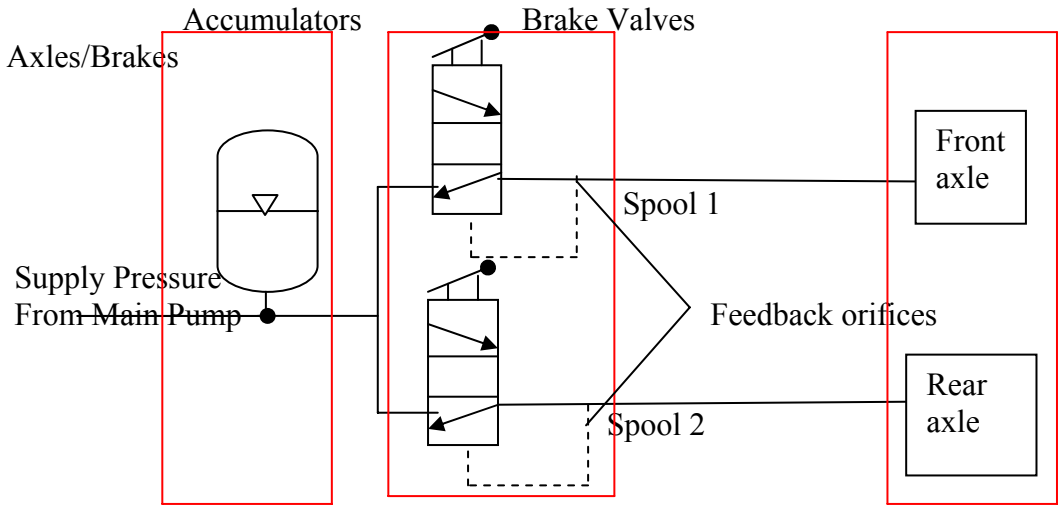


Figure 3. Brake system schematic

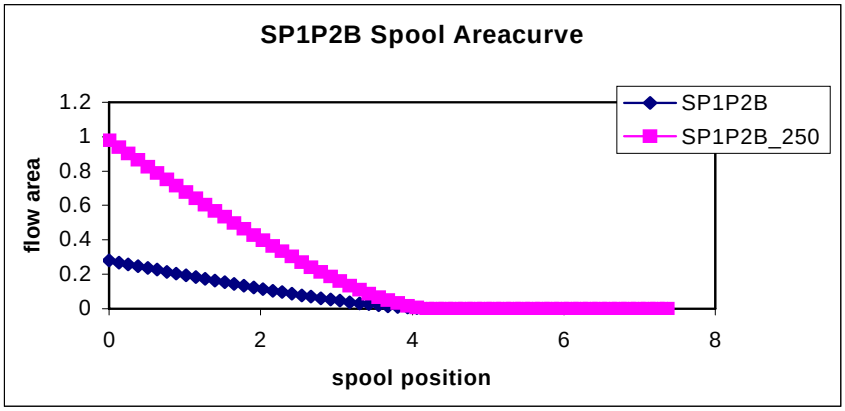


Figure 4. Scalar of original areacurve

Crossover:

The crossover operator uses the traditional one point crossover method *(5).* This method randomly picks a position then interchanges two parents' genes. For instance, if the position two is picked, assuming parents' genes are (1.0 2.0 3.3 3.5 1 2 5000) and (3.0 2.9 1.5 1.7 2 1 7000), the children's genes will be (1.0 2.0 1.5 1.7 2 1 7000) and (3.0 2.9 3.3 3.5 1 2 5000) after crossover.

Mutation:

www.manaraa.com

The mutation operator used in the project is the non-uniformly mutation method *(5)*. It randomly picks a variable in the gene, j, and sets it equal to non-uniformly random number:

$x_i^{'} = x_i + (b_i - x_i)*f(G)$ if $r< 0.5$,

$x_i^{'} = x_i - (x_i - a_i)*f(G)$ if $r>= 0.5$,

Where

$f(G) = r_2(1-G/G_{max})$

r_2, r_1 = a uniform random number between (0,1),

a_i, b_i = the low and high boundary.

G = the current generation

G_{max} = the maximum number of generations

Initialization:

The initial population is 16. Possible parameters are randomly picked from the database of the previous simulation runs. The population size is stable at 16 for each generation.

Selection:

The genetic algorithm uses the tournament selection method to select the next generation. It randomly picks two individuals as parents from the current generation and two children are generated by crossover and mutation. Each individual is evaluated by a fitness function. Only the best two will be selected from each family. For instance, two parents A and B have fitness 1.0 and 0.8 respectively, and two children C and D have fitness 2.0 and 0.5 respectively. Only A and C will be selected for the next generation.

Termination:

The simulation stops when the user-defined tolerance has been met or the maximum number of generation is reached. The tolerance is how much the average fitness increases comparing to the last generation (0.01 in the experiment). The maximum generation is how many generations are allowed.

Fitness function:

The fitness function for the brake model can be separated into three parts: deceleration, jerk, and stopping ability, which is represented by the area of deceleration

www.manaraa.com

time response. The ideal deceleration response is smaller than 0.2g, which makes the driver feel comfortable. The ideal jerk response is as small as possible. The ideal stopping ability is as high as possible. The final fitness is the sum of all three parts for four different vehicles. Equations for each part are listed as follows:

$f_{i1} = 10 * (0.2 - a)$ a<0.2 g

$f_{i1} = 5 * (a - 0.2)$ Otherwise

$f_{i2} = 5 * (1 - J) * |1 - J|$

$f_{i3} = 10 * \Delta V$

$$\text{fitness} = \sum_{i}^{4} (f_{i1} + f_{i2} + f_{i3})$$

Where

a = deceleration peak (g)

J = jerk peak (g/s)

ΔV = area of deceleration response (m/s)

i = vehicle type

Simulation Results:

The simulation stops after 40 generations. The final population converges to some certain areas of the control space. The average fitness increased from –10.21 to 2.9. As shown in figures 5 and 6, the high fitness response has lower jerk peak by 30% and larger area of deceleration time series curves by 14% which means that it stopped faster.

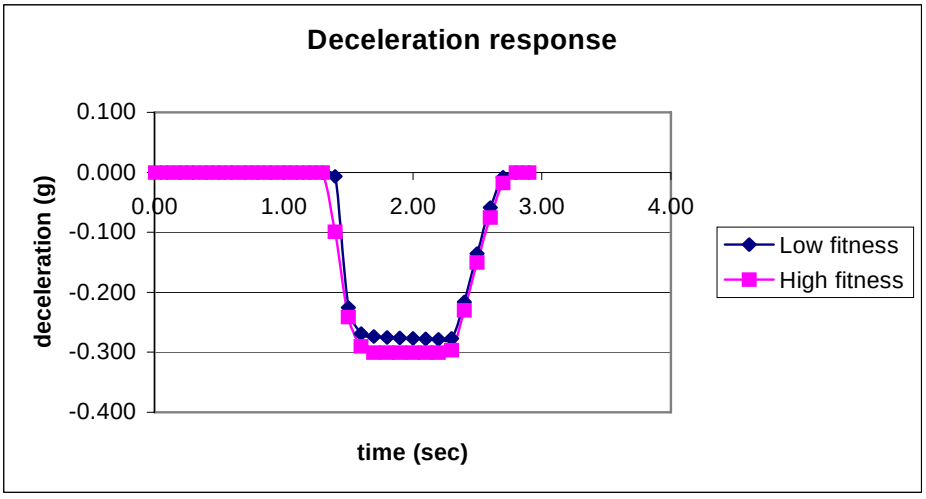


Figure 5. Comparison of responses of low fitness and high fitness

www.manaraa.com

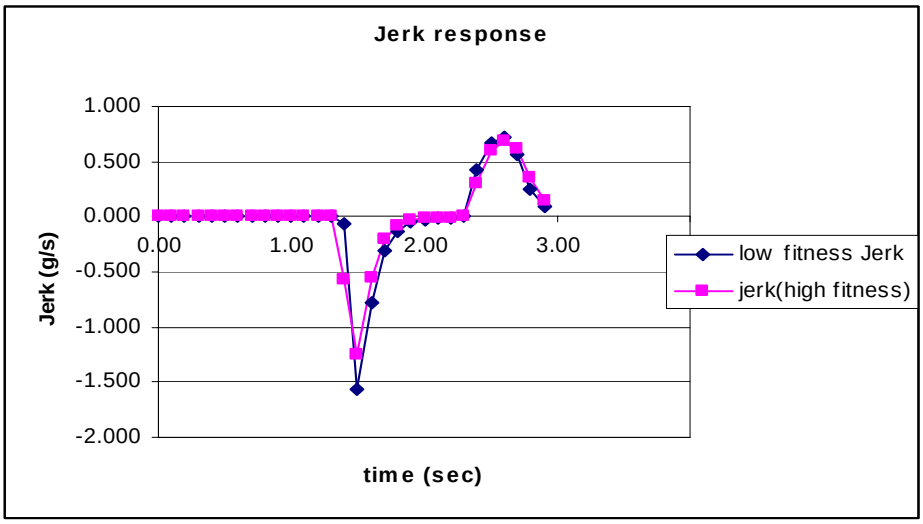


Figure 6. Comparison of jerk responses of low fitness and high fitness

Visualization can then be used to explore the design input space by linking input space with the fitness. By choosing a high fitness area, see in figure 7, users can examine the combinations of input parameters that correspond to these values. The highlighted combinations of inputs are as follows: mid-range BV1P2B and BV2P2B, low-value BV1B2T, two orifices, and Psupply, high-value BV2B2T. Some variables such as BV1B2T and BV2B2T have wider range choices. It suggests that they are less sensitive than other variables. Physically it is because they are variables controlling fluid flow from brake valves to tank, which are less important than variables controlling fluid flow from pump to brake. Now it is also a good idea to look at the multiple outputs that constitute the fitness value. It is common that not all the objectives can be satisfied at the same time. The values of the objectives that correspond to overall high fitness can be displayed in several windows (figure 8). Each window contains information of jerk and acceleration for one vehicle type. With an optimal design gotten by GA, the first, third, and fourth types of vehicles all have low jerks and acceptable high accelerations (highlighted points). But vehicle two has a relatively high jerk and low acceleration. If the jerk in vehicle model 2 is needed to reduce, other vehicle models' performance have to be sacrificed.

www.manaraa.com

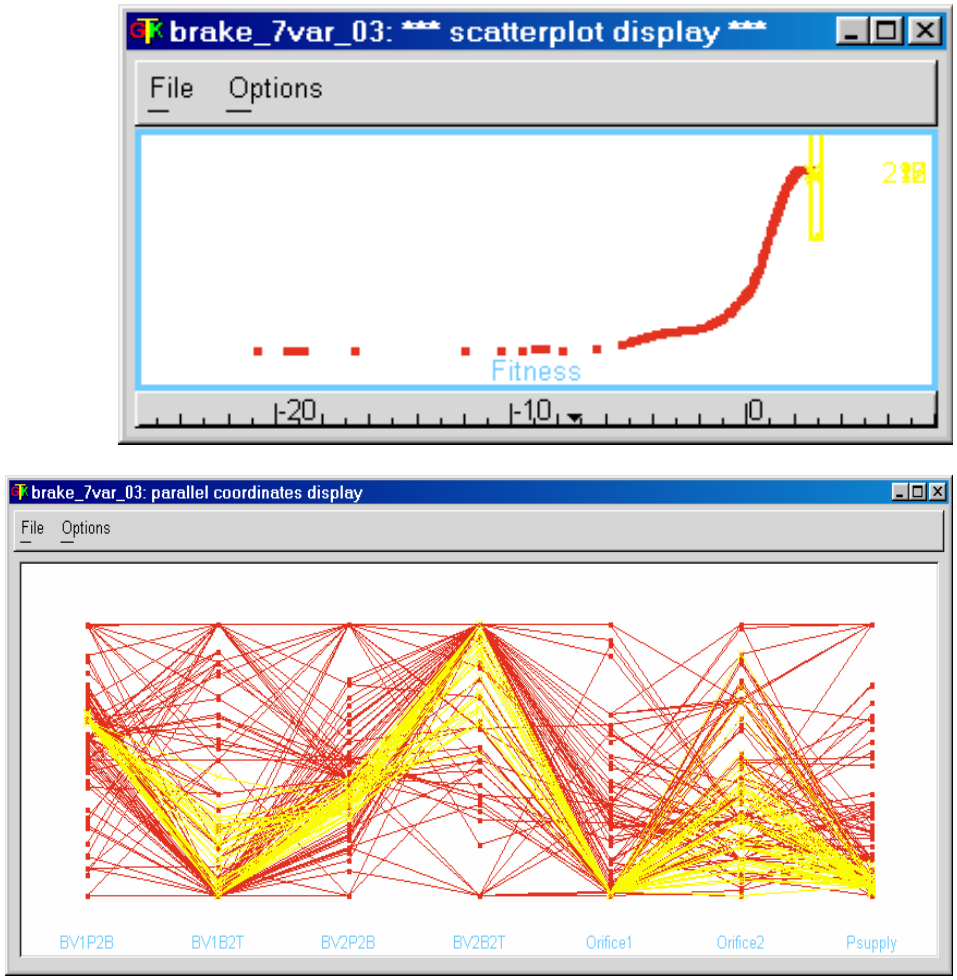


Figure 7. Explore input space while brushing fitness

A similar process could be used to choose good fitness functions. Several different fitness functions could be computed, and their values could be examined in relation to the output space. The fitness function can be chosen by matching the best fitness values with the best values in the objective space. Design experts have some intuition in picking an optimal space that optimizes and balances multi-objectives of physical system.

www.manaraa.com

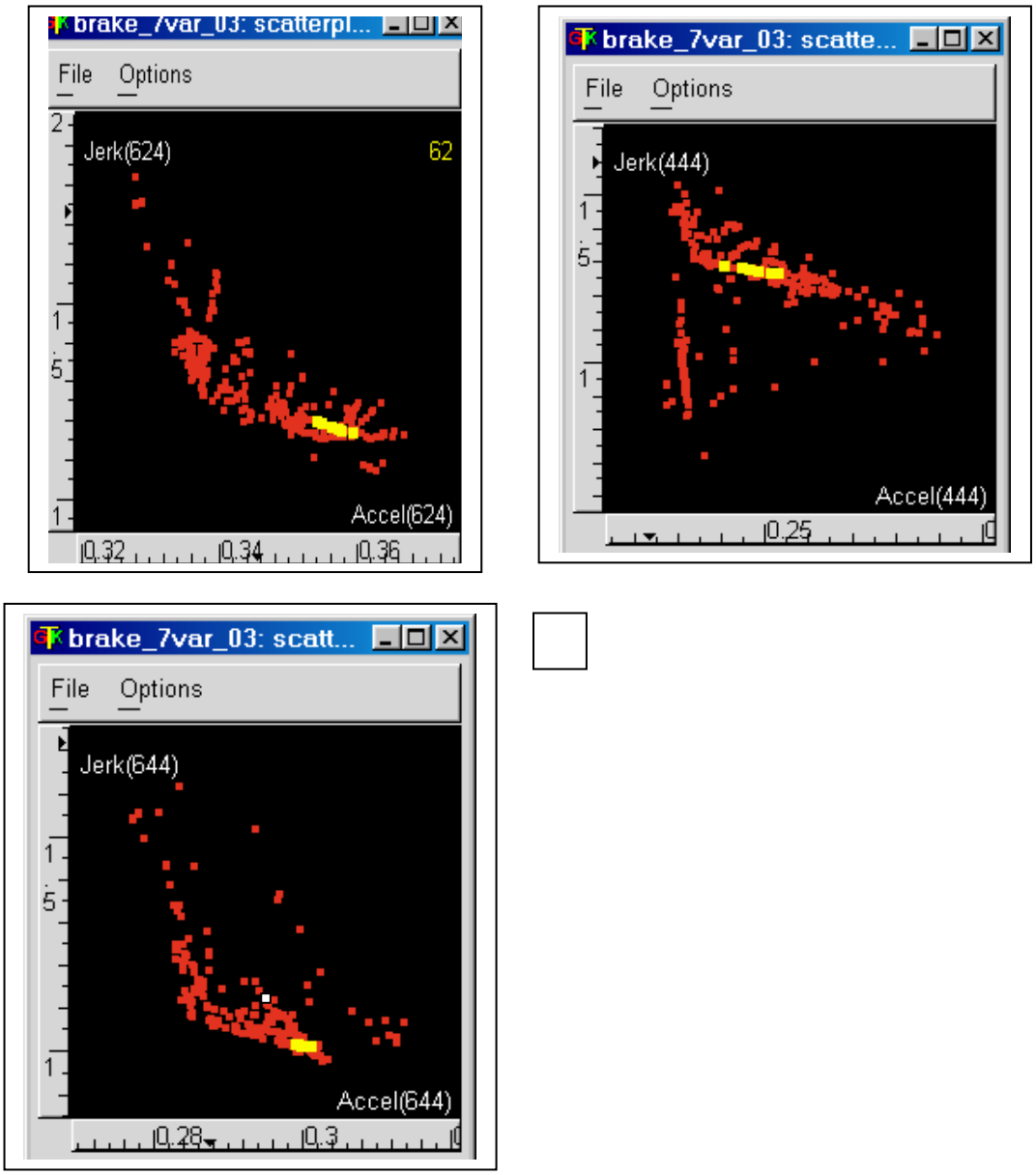


Figure 8. Exploring output space and visualizing multiple objectives

Conclusions and Discussions

The paper presents a general process to solve multiobjective optimization problems, particularly engineering problems with dynamical modeling. GAs are used to search the high dimensional space and find the optimal solution. High dimensional visualization, such as available in GGobi, is ideal for exploring input and output space to understand the relationship of input and output variables, understand relationship of multiple objectives, and design the fitness function to guide GAs. With these two tools and engineers' expert knowledge, designers may be able to arrive at optimal solutions

www.manaraa.com

quickly and correctly. This approach can be applied to many practical design problems and is demonstrated here by the brake control application.

In the paper, multi-objective functions have been combined into one fitness function using weighting coefficients. Other available approaches in multi-objective genetic algorithms (*15*) (*16*) are to locate Pareto-optimal solutions without defining a fitness function. It often requires extensively exploration in the input space, and thus needs larger population size and generation iteration. Dynamical modeling needs several minutes' computing time or even more, depending on model complexity and time step, to finish each individual run. The whole GA optimization process might take weeks' time. However, GA parallel feature may make multi-objective genetic algorithm applicable to dynamical modeling with parallel computing technology. Future research on this area is needed.

Acknowledgement

This project is supported by John Deere Corporation. We would like to thank Brian Kellogg, Tye Conlan, Lary Williams and Mac Klingler of John Deere Dubuque Works for many helpful suggestions and comments. Di Cook of Iowa State University has guided us on the use of the GGobi statistical visualization software.

References

1. J. Bernard, J. Gruening and K. Hoffmeister, "Evalution of Vehicle/Driver Performance Using Genetic Algorithms", SAE 980227.
2. P. N. Koth, J.P. Evans, and D. Powell, "Interdigitation for Effective Design Space Exploration using iSIGHT", www.engineous.com.
3. K. C. Tan, T. H. Lee, et al., "A Multiobjective Evoluationary Algorithm Toolbox for Computer-Aided Multiobjective Optimization", IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, v31 n4, 2001, P537-556.
4. C. R. Houch, J. A. Joines, and M. G. Kay, "A Genetic Algorithm for Function Optimization: A Matlab Implementation", NCSU-IE TR 95-09, 1995.
5. Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", Springer-Verlg Berlin Heidelberg 1994.
6. M. Gen and R. Cheng, "Genetic Algorithms & Engineering Optimization", John Wiley & Sons, Inc. 2000.

www.manaraa.com

7. D. Quagliarella, J. Périaux, et al., "Genetic Algorithms and Evolution Strategies in Engineering and Computer Science", John Wiley & Sons, Inc. 1998.
8. D. F. Swayne, D. Cook et al., "GGobi manual", www.ggobi.org.
9. R. Goudy and S. Andrews, "Brake control strategy for optimized safety and comfort", Proc. 5th world Congress ITS, Seoul, Korea, Oct. 1998.
10. T. K. Liu, T. Ishihara, and H. Inooka., " Multiobjective control systems design by genetic algorithms", Proceedings of the 34th SICE Annual Conference, Jul 26-28 1995, Hokkaido, Jpn, P1521-1526.
11. Y. F. Yin, "Multiobjective bilevel optimization for transportation planning and management problems" *Journal of Advanced Transportation*, v 36, n 1, Winter , 2002, P93-105.
12. C. A. Coello, A. D. Christiansen, and A. H. Aguirre, "Using a new GA-based multiobjective optimization technique for the design of robot arms", *Robotica*, v 16, n pt 4, Jul-Aug, 1998, P401-414.
13. "EASY5 User Guide", The Boeing Company, 2000.
14. "Learning Simulink", www.mathworks.com
15. J. Horn, N. Nafpliotis, D. E. Goldberg, "A Niched Pareto Genetic Algorithm for Multiobjective Optimization", Proceedings of the First IEEE Conference on Evolutionary Computation, 1994.
16. N. Srinivas, K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms", Evolutionary Computation, 2(3), 1994, P221-248

www.manaraa.com

CHAPTER 4. COMPLETELY DOMINANT GENETIC ALGORITHMS

4.1 Introduction

Practical physical dynamical systems are nonlinear and complex. System behavior is difficult to fully understand using traditional mathematical analysis due to nonlinearities and uncertainties. Designers often construct detailed dynamical models to simulate system dynamics and try to vary several design variables to assess the system response. For any changes in different variables, the system response needs to be checked by running detailed simulations. This process can be quite time consuming to find a satisfactory solution for a high dimensional problem. Moreover, a local search has to be performed to make sure that the solution meets robustness requirement.

Genetic Algorithms (GAs), developed by Holland [1], are inspired by natural selection and survival of the fittest. GAs are powerful and robust stochastic search and optimization techniques, which have been applied to many engineering and mathematical areas such as engineering design [2, 3] and stock investment [5]. GAs can solve complex problems that are difficult to solve with conventional techniques, such as gradient approximation methods. In addition, they can help automate search process to find a satisfactory design variable combination without completely understanding interactions between input and output.

For a single objective optimization problem, the optimum is the minimum or maximum of the objective function. Real-world problems often have several criteria that sometimes conflict. The optimality of multiobjective optimization is defined as Pareto optimality [4]. Dominance is an essential concept in Pareto optimality. If a solution **A** is said to dominate another solution **B**, it means that all of **A**'s objectives are better satisfied than **B**'s. The set of non-dominant solutions among all possible solutions is called the Pareto front. The goal of multiobjective optimization is to find a trade-off solution on the Pareto front. Several techniques, such as Pareto ranking and using a weighted-sum function, have been used to solve multiobjective problems [6-7]. Cvetkovic and Parmee have presented several multiobjective optimization methods using in engineering evolutionary design [18, 19].

www.manaraa.com

There are two main directions for solving multiobjective problems using GAs: 1. Converting multiple objectives to a single objective; 2. Using Pareto domination ideas. In the first case, the algorithm is designed to make the population converge to a global optimum. It may represent one point in the Pareto Front. Designers also need to map objectives to a single fitness function. Parmee has applied the weighted sum method into many engineering design preliminary studies [18, 23]. In the second case, the population tends to converge to the Pareto-optimal front. The solutions on the Pareto-optimal front are non-dominated. One of problems using Pareto dominant selection is that non-dominant solutions will increase dramatically as process goes on. Without diversity control, population will often be trapped into local Pareto fronts. Therefore, several GAs, such as Reduced Pareto Set GA [8], Diversity Control Oriented GA [11], keep the diversity of the population well distributed to cover the whole Pareto front space. Crowding and Niche sharing have been used to keep diversity in the population [26].

Parmee has proposed Cluster Oriented GA (COGA) to help engineer do preliminary study [24, 25]. The basic idea is to help GAs converge to high performance (HP) areas quickly, which is defined as an area close to the Pareto front. The high performance clusters can help design engineer further understand system such as input and output interactions and design variables sensitivity. COGA uses a variable mutation rate to allow diversity in the final stage so that it can formulate high performance clusters. In order to prevent low fitness solutions from falling into clusters, filters have to be used in evolutionary process. Filters' functionality is to use a threshold to manage clusters. COGA has been used in many real engineer design problems [24, 25].

COGA's high performance concept is exceedingly practical in engineering study. Engineering design needs to consider many other side factors, such as cost and robustness. A quick way to identify HP areas is helpful to further investigate designs with other factors in mind. COGA uses adaptive filters to direct evolution to HP area. The way to calculate threshold for adaptive filters has many variations and is problem dependent. Adaptive filters still use the fitness idea, which limits its ability in solving multiobjective problems.

This paper proposes a new genetic algorithm method called Completely Dominant Genetic Algorithm to achieve convergence to HP areas. It is intuitive to multiobjective domain because it is based on objective space without mapping them to single fitness function. It relaxes the dominance condition in selection process to allow the genetic

www.manaraa.com

algorithm to explore high performance areas and help to find the robust solution. Detailed description and discussion are presented in the following sections.

4.2 Mathematical Preliminaries

This section gives the mathematical formulation for multiobjective optimization problems and necessary definitions that are used in this paper.

Multiobjective Optimization Problem:Vector $\mathbf{x} \in X$ where X is a subspace of R^n, objective function $\boldsymbol{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \in R^k$. The goal is to minimize function $\boldsymbol{f}(\mathbf{x})$, i.e. minimize all $f_k(\mathbf{x})$ functions, with $\mathbf{x} \in X$. For multiobjective problems, k and n are larger than 1. Each $f_i(\mathbf{x})$ is assumed to be a continuous function.

Robust design: A robust system satisfies the objective specifications for all perturbed cases about the original model up to the worst-case perturbation [12].

Pareto Dominance: For two vectors $\mathbf{A}=[a_1, a_2, ..., a_n]$, $\mathbf{B}=[b_1, b_2, ..., b_n]$ in R^n space, if $\forall$ i, $a_i < b_i$, then **A** is Pareto dominant to **B** [9].

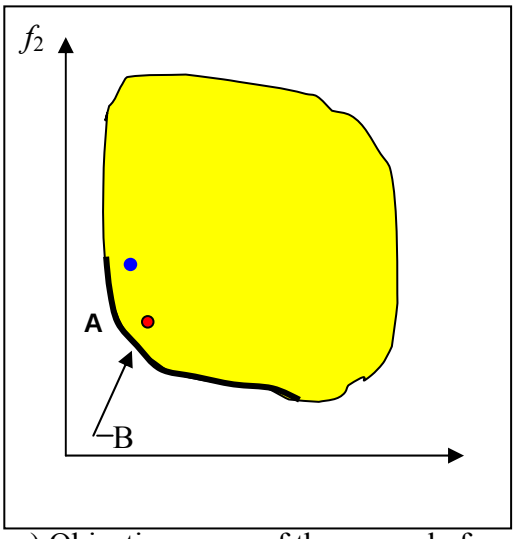
Complete Pareto Dominance: For two vector **A**, **B** in R^n space, given a positive tolerance value ε, let vector $\mathbf{C} = (\varepsilon, \varepsilon, \ldots, \varepsilon)$, a hypercube in n dimension. If **A** is Pareto dominant to **B – C** (+ for maximization), we call **A** is completely dominant to **B** with respect to ε.

4.2.1 Description

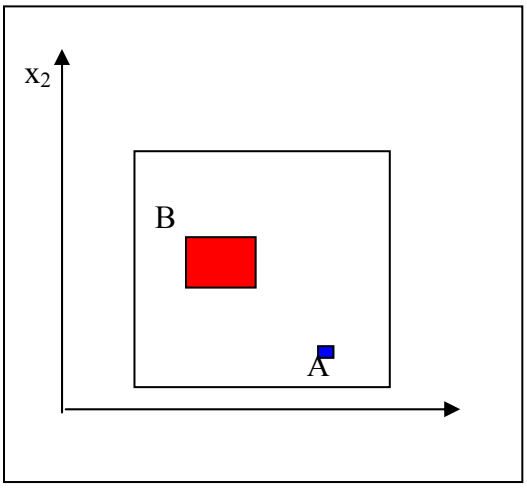
Figures 1 (a) and (b) illustrate a two dimensional (x_1, x_2) optimization problem with two objectives (f_1, f_2). Figure 1a shows two possible solutions, **A** and **B,** which are close to the Pareto-optimal front [12]. They are non-dominant with respect to each other. Assuming that, decision-makers have to make a choice between two with robust consideration. In this case, the corresponding input space of solution **B** has a larger space than the space of the solution **A** given the same tolerance (Figure 1b), i.e. the solution **B** has larger tolerance for uncertainties than **A** does. Therefore, the solution **B** is considered more robust than **A**.

www.manaraa.com

The ideal design procedure is to first pick a HP space, which is close to the Pareto optimal front, and then pick several good solutions from this space and evaluate them with respect to robustness. The key step is locating the HP space. Robustness can be checked by local search in HP space. Our idea is to relax selection pressure in some degree to allow diversity in population and also push the evolution process going to the Pareto Front space.



a) Objective space of the example for solutions A and B



b) Input space of the example for solutions **A** and **B**.

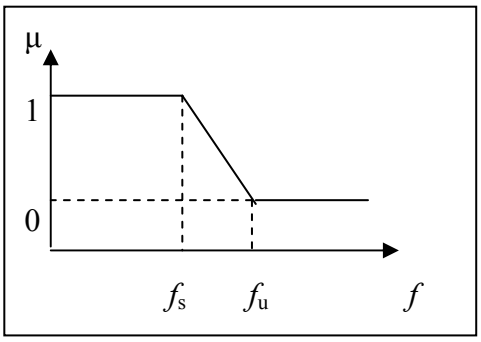


Figure 2 Fuzzy fitness function

www.manaraa.com

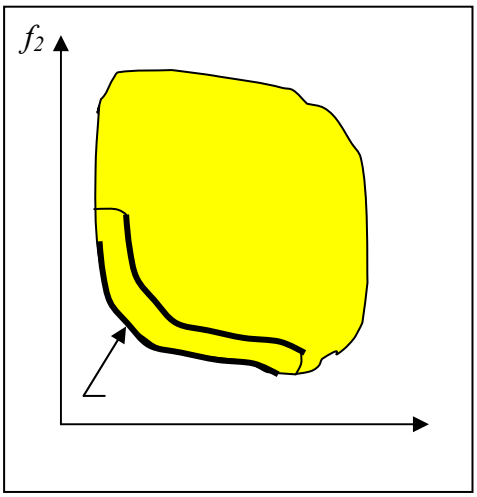


Figure 4 HP space using Complete Pareto Domination

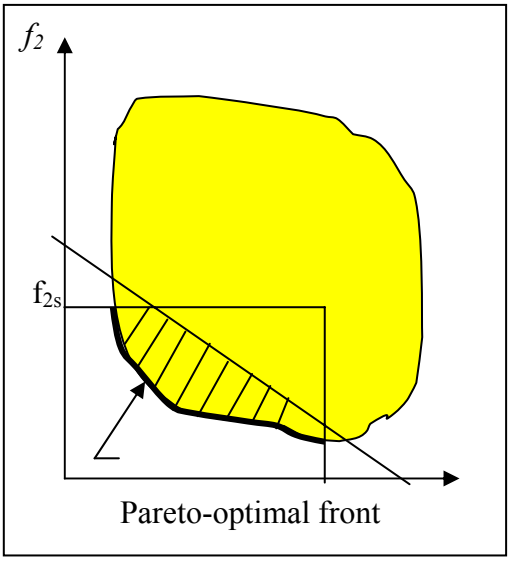


Figure 3 HP space using the fuzzy fitness function

There are two possible selection methods that meet the requirements: fuzzy objective functions and complete Pareto domination. The fuzzy function is a mapping from an objective function to a satisfaction function. f_u represents an unsatisfied value for a particular objective, while f_s is the satisfied value (Figure 2). The goal is to minimize all objectives so as to maximize the minimum membership function for all objectives. In figure 3, the input space between f_{1s} and f_{2s} is considered to contain satisfactory solutions. Adding some fuzzy penalty terms, i.e. $p=\Sigma w_j f_j$, can reduce the size of the subspace, as shown by the shaded area in Figure 3. From the genetic algorithm viewpoint, all of the populations in the shaded area have the same fitness. An elite pool is used to store all the individuals falling into this area, which is considered as HP region, and individuals with the same fitness are selected with equal probability.

Using fuzzy logic allows the GA solutions to converge to the satisfied objective areas. However, the search space expands as problem dimensionality increases. The ideal objective space includes all areas close to the Pareto-optimal front. Using the completely dominant concept ensures that the solution can converge to these areas. If **A** is not completely dominant to **B**, the solutions are considered the same. The consequence of this selection is that all the values in the shaded area of Figure 4 are considered as HP area.

www.manaraa.com

The elite pool generated by the modified GA contains all the individuals with objectives in the HP space. Further local search and exploration in HP area can determine solution's robustness and other features.

4.3 Engineering Multiobjective Optimization

A lot of researches have been conducted to develop multiobjective optimization methods. Genetic algorithms are one of appearing algorithms in Pareto optimization. Engineering design encounters numerous multiobjective optimization problems in the process of designing products. Multiobjective Genetic algorithms have been applied to many engineering design areas [11, 30].

Engineering multiobjective design problems have their own characteristics that differ from general multiobjective optimization problems. First of all, complex system design is not a simple straightforward problem. Design variables and objectives are not well defined sometimes in preliminary design stage. Parmee has suggested using Cluster-Orient GA to locate High Performance areas for preliminary design. Designers have more interests in exploring High Performance areas and find interactions between design variables and objectives than finding optimal solutions in this stage. Designers want to explore interactions of design variables and objectives and sensitivity of design variables. Therefore, multiobjective optimization is desired to explore search space well especially in High Performance areas.

Second, engineering design problems have critical concerns on robustness. Due to manufacturing tolerance and volatile working environment, there are a lot of uncertain factors affecting complex system. Design variables could have disturbance and objectives could be not calculated accurately. For engineering design, most of designs are done on the computer by simulation models. Design engineers have to make sure the final product meet robustness requirement in simulation phase. It requires that multiobjective optimization has to consider robustness besides optimization.

Diversity and robustness of multiobjective optimization results are two basic concerns to engineering design multiobjective optimization. To apply GA in multiobjective algorithm, GAs have to consider these two factors in order to meet optimization goal. Previous researches have proposed various types of ideas to preserve population diversity in evolutionary process and find the most robust solutions. This

www.manaraa.com

chapter will review previous research results and discuss completely dominant GA's performance on diversity and robustness.

4.3.1 Diversity

One of multiobjective genetic algorithm optimization difficulties is keeping diversity of population. The Pareto Front is a hyperplane in the same dimension of number of objectives. As dimensions increase, the non-dominated solutions in the population will increase exponentially. However, most of genetic algorithms are keeping population size stable, i.e. consistent number of children will be produced for each generation. Non-dominated solutions will quickly take control of the population. Without diversity control, non-dominated solutions possibly only cover a small part of the Pareto front.

General requirements for a good multiobjective genetic algorithm are as follows: 1. directing the population towards the Pareto Front; 2. maintaining the diverse non-dominated set; 3. preventing from losing non-dominant solutions. These three requirements are also representing the developing history of multiobjective algorithms. At the first stage, multiobjective genetic algorithms are using fitness assignment to find the Pareto Front. Pareto ranking is the primary technique for multiobjective optimization. At the second stage, more and more researchers realized the importance of diversity in multiobjective optimization. Among numerous diversity techniques, niche sharing is the most popular one that many multiobjective genetic algorithms use. Elitist selection is added into multiobjective optimization at the third stage.

Multiobjective Genetic Algorithm's performance can be measured by two criteria: convergence and diversity. Convergence test is measuring how close non-dominated population is approaching the Pareto Front. Diversity test is measuring how non-dominated set distributes compared with uniformly distribution. As Deb [27] proposed, these two criteria can be measured by two metrics. The convergence metric is defined as γ metric. It is assuming the Pareto Front is known. N uniformly distributed points from the Pareto Front are picked. For each point, the smallest Euclidean distance to GA population is found. The sum of all distances is the γ metric. Even if every non-dominant point is located on the Pareto Front, the metric is not zero. Only if when non-dominant points are uniformly distributed on the Pareto Front, the metric will be zero. It can measure diversity along the Pareto Front in some degree.

www.manaraa.com

Diversity Δ metric is defined as the following:

$$\Delta = \frac{D_f + D_l + \sum_{1}^{N-1} \left| d_i - \bar{d} \right|}{D_f + D_l + (N-1) * \bar{d}}$$

This metric is best to apply to two-objective Pareto Front. D_f and D_l are the minimum distance of non-dominant sets to two extreme points on the Pareto Front. All N non-dominant points are sorted continuously along the Pareto Front. $\bar{d}$ is the average distance of N-1 consecutive distance of non-dominant points. When the distribution is exactly uniform, the numerator will be zero. In other cases, the metric is always positive and can be over one. Small Δ metric value normally represents that distribution is close to uniform.

4.3.2 Robustness

Except convergence and diversity, robustness is also an important factor deciding the performance of GA. For engineering design, there always are uncertainties due to manufacturing tolerance and uncertain operating environment. Design variables may have variations in reality to some extent, while they are fixed in modeling. Any variations may affect system performance (objectives). Robustness can be viewed as sensitivity of variables around optimal areas. Considering two sets of design (Figure 5), design A and B have the same optimality, but design A is less sensitive to design variable. Clearly, designer wants to pick the design A due to its robustness.

www.manaraa.com

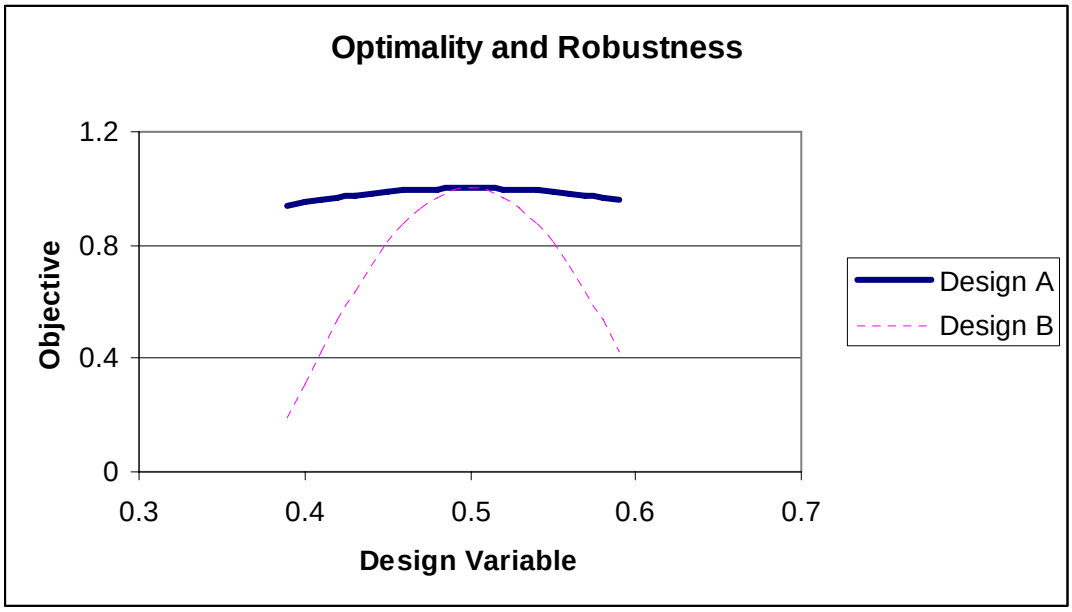


Figure 5 Example of Optimality vs. Robustness

Previous approaches to deal with noise and uncertainty in GAs include using perturbed objective functions [9], averaging objectives or modeling uncertainty as a Gaussian noise [10]. Uncertainties for GA can be characterized into two categories. 1. The same design combination can have different objectives because of uncertain objective functions; 2. The same design combination has the same objectives, but design variables have variations. In Anderson's dissertation, three approaches to deal with robustness in Hydraulic system design have been explored. The first method is to use disturbing design variables. If some design variables have disturbance, objectives or characteristics of system are affected correspondingly. The idea is generating the actual value of design variable based on the distribution for each evaluation. That means for the same design combination, due to disturbance of one design variable, it has to be reevaluated in future generations and can get different objective values.

The second method is to use design of experiment described in Anderson's paper. The basic idea is using regression analysis to find out the polynomial function between objectives and controlled factors and uncontrolled factors. The influence of controlled factors and uncontrolled factors can be estimated through evaluation of objective disturbance. This approach is taken after optimization process. It is actually a post-exploration of optimum. It won't help optimization algorithm to locate robust optimal areas in search progress.

www.manaraa.com

The third method is called metamodel. It uses a second order polynomial without cross term to model objective function in terms of design variables. By studying coefficients of the polynomial, the robustness for each solution can be estimated. It can give a whole picture of interaction between design variables and objectives. The disadvantage is that fitting for the second polynomial function is hard for high dimensional nonlinear system and estimation could be far away the actual system.

In the dynamical modeling design problem, uncertainty characteristics are often unknown since they may come from modeling inaccuracies and/or manufacturing tolerances. Until the actual product is produced, changes in manufacturing and the difference between the actual product and the dynamical model are unknown. If using the distribution approach, it applies that the actual product still has possibility to fail to meet requirement, even though the possibility can be very small. Any product failure could bring financial suffer and damage to company's reputation. In engineering design area, any worst perturbation has to be respectably considered.

For most of engineering design problems, robustness is the first priority over optimality. Designers are willing to sacrifice certain degree optimality to achieve system robustness over perturbation. Transferring to multiobjective optimization problems, it is equal to allow that the final solution is not exactly on the Pareto Front in order to meet robustness requirement. It is hard to make quantitative decision about how much optimality will be sacrificed and how robust is needed to achieve. Measurement for robustness has to been established for robustness comparison.

A robust system satisfies the objective specifications for all perturbed cases about the original model up to the worst-case perturbation [12]. This concept is borrowing from control system design. There are many similarities between controller design and engineer design. The measure of robustness is defined as follows: Given a tolerance ε for objectives, the robustness is measured by the biggest perturbation δ allowed. Given a positive ε, $\max(|\mathbf{f}(\mathbf{x}+\mathbf{\Delta}_\mathbf{n}a)-\mathbf{f}(\mathbf{x})|)<\varepsilon$, for any $\mathbf{\Delta}_\mathbf{n}$ and $a\in(-\delta,\delta)$, vector $\mathbf{x}\in X$. $\mathbf{\Delta}_\mathbf{n}$ is a vector in R^n space and its H_∞ norm [12] is less than 1. The measure of robustness is determined by the original state **x**, i.e. design variables, and the given tolerance ε.

www.manaraa.com

4.3.3 Clustering for robustness

From the definition of robustness, the measure of robustness is determined by δ. In reality, δ is not easy to be found accurately. One of the common ways is using local search to find robustness. The idea is to explore the neighboring area in the input space of a solution and find its minimum distance to over the tolerance. Local search is very time-consuming, because it has to be performed on each good solution. For high dimension engineering design problems, numerous solutions exist in High Performance area. If each solution needs a local search, it is going to take tremendous time, considering that each neighboring combination requires a simulation to get results for engineering design.

The alternative way to compare robustness is to estimate robustness metric based on the existing solutions without additional simulations. It is based on the assumption that existing solutions have the same robustness characteristic as all solutions. Clearly, this way is much easier due to no additional simulation. It generally transfers robustness issues to a clustering problem. If a cluster with the largest size is found in existing solutions with respect to a tolerance, the cluster center will be considered as the most robust solutions. It is a simple search problem comparing with calculating robustness metric for every solution.

Even for complex multiobjective optimization problems, the clustering method is not going to take too long to process, because design engineers are only interested in solutions in High Performance area. The clustering is only needed to be performed for those solutions. The general clustering procedure is as follows:

1. Find the Pareto Front of all solutions

   No matter what Heuristic optimization search techniques are using, a lot of simulation results will be generated. Among them, non-dominant solutions form the Pareto Front. This may be different from the actual Pareto Front. However, simulation results in this set have to be considered as the optimum solutions, since they are the best solution found.

2. Find the High Performance area

   The high Performance area concept comes from Clustering Orient Genetic Algorithm (COGA). It is particularly useful for Engineering Preliminary study. Its original definition is to use a filter to put high performance solutions into populations during genetic algorithm process. In COGA, HP area is defined as an

www.manaraa.com

area close to the Pareto front. Given a vector with the same dimension as objectives, each value in the vector is the tolerance for that dimension's objective, and the tolerance for each dimension can be different with each other. The objectives of all can be scaled in each dimension so that the tolerance for each objective is the same. However, design engineers are more interested in robustness in design space. The scaling should be done in the input space based on robustness requirement. For example, the disturbance distance for design variable A is 0.1 and distance for B is 0.2. The design variable A should scale 2 times to get the same disturbance as B. In this way, the largest hypercube cluster that can be found is the most robust area. If a design variable has no disturbance, it will not be considered in hypercube.

3. Reduce solutions outside of HP area

   The way to finding clusters is to check all clusters formed by every solution as a cluster center. In the actual coding, every solution has to search all solutions to find its maximum distance in given by certain tolerance vector. Solutions in objective space that are too far away from HP area are not necessarily needed to consider because their distance to solutions in HP exceed tolerance vector. Therefore, solutions that are two times tolerance distance away from the Pareto Front do not need to be included in cluster searching. This will reduce a lot of unnecessary searches during the searching process.

4. Search the biggest cluster for each solution in HP area

   After reducing solutions outside of HP areas, searching cluster process will be much more improved. The goal is to find the cluster size for each solution based on tolerance. The search process is that for each solution, finding the biggest hypercube in which all solutions' objectives are in the tolerance range of the solution. The cluster metric is defined as the maximum distance to the centroid for any point in the cluster. The distance metric is using L_1-norm metric.

5. Compare cluster size

   When clusters for all solutions in HP area have been found, the most robust cluster is needed to be picked from them. There are two ways to compare clusters:

www.manaraa.com

by cluster size and by number of solutions. Experiments show that these two methods often match each other. When two methods don't match, one of possible reason is that GA doesn't cover the Pareto Front very well. This results in that an uncrowded area possibly has a bigger cluster size just due to fewer solutions in this area. Using the second method increases the confidence level on robustness. Because even though solutions in HP are not evenly distributed, the cluster has been checked with the most number of solutions. The cluster size found by this method is most likely able to be guaranteed to match with the actual one.

Pseudo code for clustering:

1) Loading Data
2) Finding the dominant objectives

```
FOR i=1 to data row size
        FOR j=i+1 to data row size
        Compare objectives[i] with objectives[j]
        END FOR
IF objectives[i] is NOT dominant by any other objective set
        Objectives[i] belong to the dominant set
End IF
END FOR
```

3) Finding the High Performance Area and NON-HP area

```
FOR i=1 to data row size
        FOR all dominant data sets
        Compare objectives[i] with objectives[j]
        IF objectives[i] is within tolerance distance to any dominant data set
                Objectives[i] belong to the High Performance set
                BREAK
        End IF
        END FOR
END FOR
```

4) Finding the largest cluster

```
FOR i=1 to High Performance Area data size
        FOR j=1 to NON-HP data size
        Compare objectives[i] with objectives[j]
```

www.manaraa.com

IF objectives[i] is within tolerance distance to any dominant data set

Objectives[i] belong to the High Performance set

BREAK

End IF

END FOR

END FOR

FOR all data sets in High Performance Area

Find the size of cluster

END FOR

Example

To minimize two objective functions f when $x_1, x_2, x_3, x_4, x_5 \in [0 \quad 0.65]$

$$f_1 = |g(x_1) + g(x_2) - 1| + 1 - g(x_1) + x_3 * x_3 + x_4 * x_4 + x_5 * x_5$$

$$f_2 = g(x_1) * g(x_1)$$

where

$$g(x) = \begin{Bmatrix} 2 * x & when & x \le 0.2 \\ 0.3 + x/2 & when & 0.2 < x < 0.4 \\ 2 * x - 0.3 & when & 0.4 \le x \le 0.65 \end{Bmatrix}$$

Characteristics of Example Problem

The example is a simple two objective optimization problem with five parameters. Variables x_2, x_3, x_4, x_5 only affect the objective function, f_1. In order to minimize f_1, $g(x_2)$ should be equal to 1-$g(x_1)$ and x_3, x_4, x_5 should be zeros. The Pareto front is easy to calculate and is shown in Figure 6a. $g(x)$ is a continuous, nonlinear function. It is designed such that it is less sensitive in the 0.2 to 0.4 range. Therefore, the objective function is robust when x_1 and $x_2 \in$ [0.2 0.4] (Figure 6).

www.manaraa.com

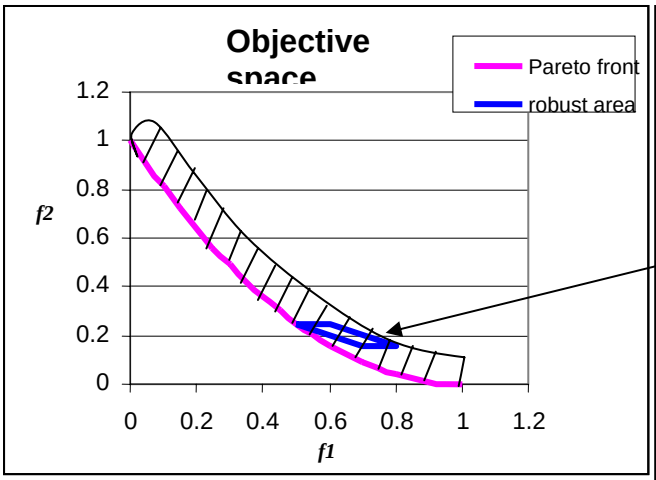


Figure 6a robust area and Pareto-front optimum

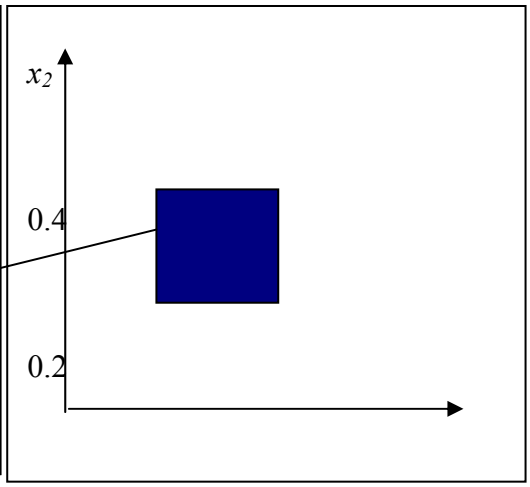


Figure 6b robust design in input space

This example has a very simple Pareto front and has a clear robust design area. If robustness is the priority of the optimization problem, the optimum could be sacrificed a little to achieve robustness. Assume the shaded area in objective space is acceptable to designers. Our question is that if we can use genetic algorithm to find not only Pareto-optimal front, but also detect this robust area.

In this section, several genetic algorithms with Pareto fitness function have been applied to this problem. The only difference between them is how they assign fitness and select the next generation. The genetic operators and parameters are listed in table 1. The non-uniform mutation method selects one variable randomly and sets it equal to a uniform random number generated from a range decreasing with generation [14, 16]. A clustering algorithm is used to find the robust design in GA results. The robust design has the most neighborhood points around it. The objective functions of any its neighborhood points are within the tolerance of objectives of the robust design point. The tolerance used is [0.5 0.5] for two objectives respectively.

Table 1 Genetic operators and parameters

GA type	Floating point
Initial Population	64
Mutation	Non-uniform
Crossover	One point
Selection	Tournament
Termination	50 generations

www.manaraa.com

(1) Pareto Multiobjective Optimization

The selection for the Pareto multiobjective GAs is the Pareto dominance. If gene A dominates another gene B, A's fitness is high than B's. Figure 7 shows only the Pareto multiobjective GA results that fall into the accepted area. The most robust point found by the clustering algorithm is [0.015 0.616 0.147 0.0641 0.063]. It is not in the theoretical robust area, since x_1 and x_2 do not fall into the [0.2 0.4] range. The Pareto genetic algorithm doesn't use diversity technique such as crowding and sharing. The reason for this is to compare with the completely dominant Pareto GA for the same conditions. These techniques prevent the search from getting stuck in a local minimum.

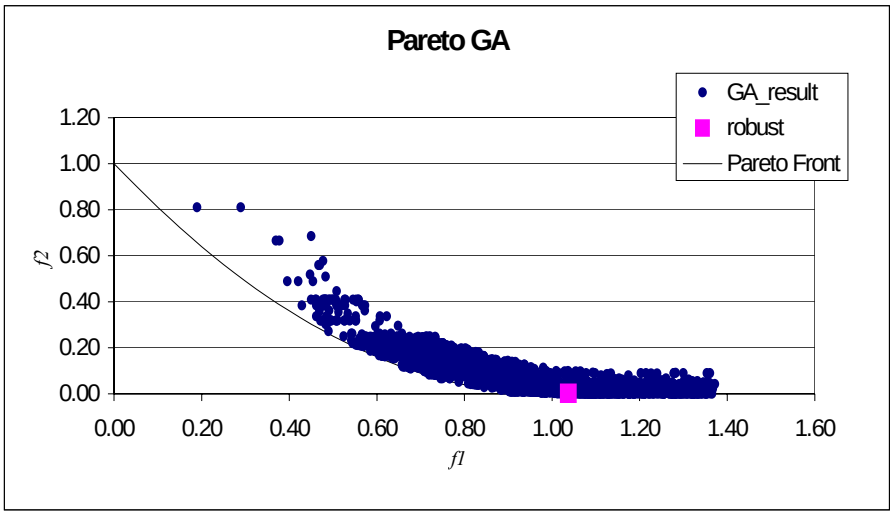


Figure 7 Objective results of the Pareto GA

(2) Fuzzy Fitness Functions

Using the fuzzy fitness function defined in figure 2, the satisfied values for *f1* and *f2* are set to 0.75 and the satisfied sum of *f1* and *f2* (Σf_j) is set to 1. The most robust point found by the fuzzy fitness GA is in the theoretical robust area (Figure 8). The corresponding **x** is [0.336 0.364 0.002 0.032 0.009], which is in the robust area.

www.manaraa.com

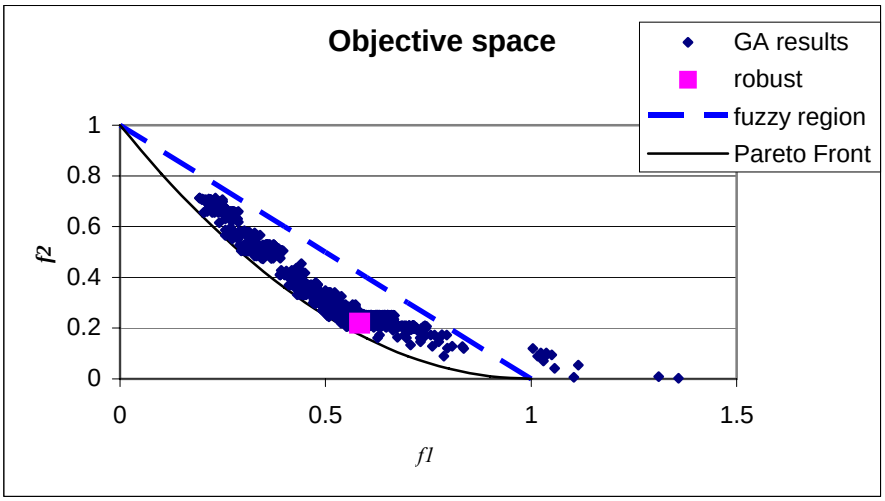


Figure 8 Objective results of the fuzzy fitness GA

4.4 Completely dominant GA

The tolerance in the complete Pareto dominance is set to [0.5 0.5]. If gene A's objectives is Pareto dominant to gene B's objectives minus the tolerance, gene A is completely dominant to gene B so that gene A's fitness is better than gene B. Using the completely dominant Pareto space, the most robust point that completely dominant Pareto GA found is in the theoretical robust area (Figure 9). The corresponding **x** is [0.300 0.336 0.081 0.061 0015], which is exactly in the robust area.

The robust area in design space found by clustering method is shown in Figure 10. It only plots the design space in x_1 and x_2 dimension and x_3-x_5 are very close to zero for high performance area due to the characteristics of the problem. It is clear that the robust area CDGA found is located in HP area and has no overlapping with the Non-HP area. The robust are is the biggest existing cluster that we can find at such tolerance of objectives. If we change the objective tolerance, i.e., the definition for HP area, the robust cluster will be changed. Similarly, if we choose other solutions as our robust solution, the cluster size will not be the largest. Of course, designers can re-modify their requirements for HP area and tolerance for each design variable so that they can get better optimization results. But cluster size may be changed accordingly.

www.manaraa.com

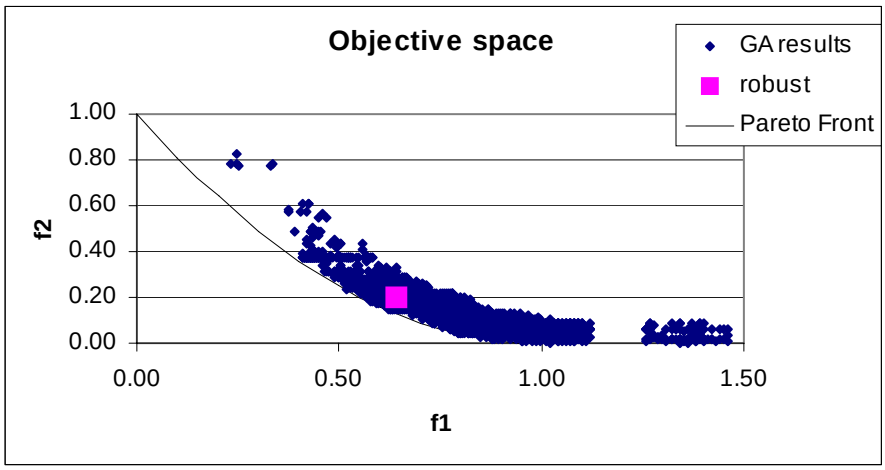


Figure 9 Objective results of the completely dominant Pareto GA

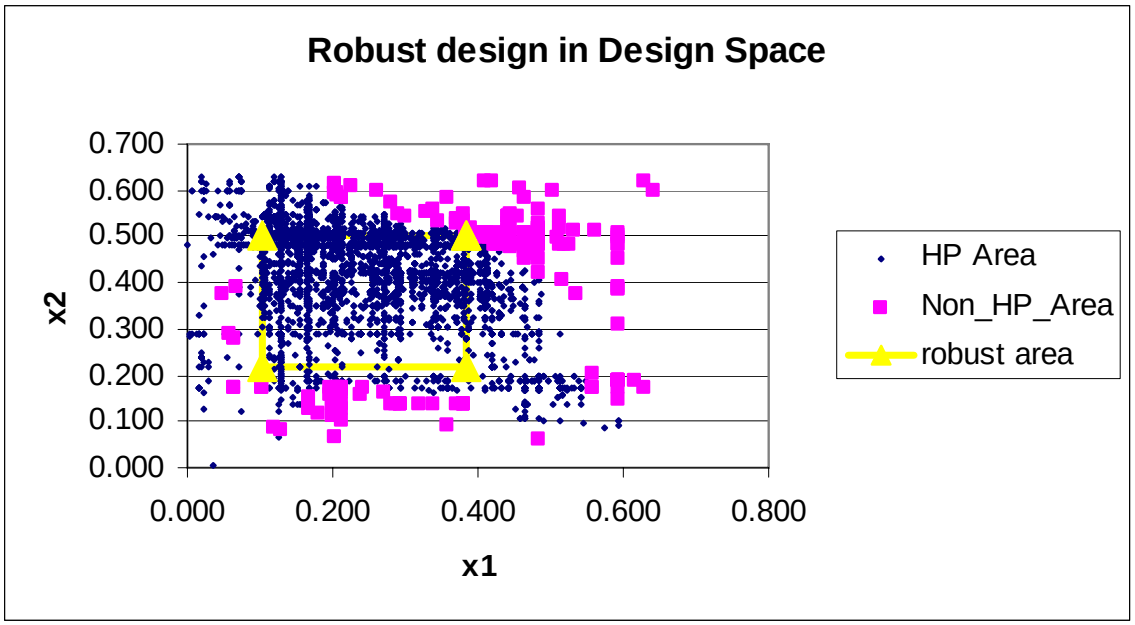


Figure 10 Robust Design Area in Design Space using CDGA

Results Comparison

Table 2 Comparison of GA performance

Algorithm	γ metric		Δ metric	
	Mean	Variance	Mean	Variance
GA real-coded	0.037	0.011	0.77	0.010
CDGA	0.025	0.002	0.60	0.05

www.manaraa.com

Table 2 shows γ and Δ metric results for real-coded Pareto GA and CDGA. CDGA has a better performance on γ metric ($p<0.01$).. The variance is very small for CDGA, which implies that CDGA's convergence performance is consistent.

From the simulation, both the fuzzy fitness GA and the completely dominant Pareto GA were able to locate the robust area and find the robust optimal solution, while the weighted-sum and common Pareto GA failed for the first example. However, the fuzzy fitness function requires that designers are knowledgeable about the system to design a suitable fitness function. It is arguable that designers could have run a number of simulations to become familiar with the system. The main problem that fuzzy fitness will face is a larger objective space to search, which increases with problem dimensionality. The completely dominant Pareto GA can successfully find robust regions and can reduce the search space. It is intuitively appealing to designers. Designers can have a clear picture about the range of objectives of the final design.

Conventional GAs using weighted sum and Pareto multiobjective methods failed to find the robust area, because these algorithms are designed to converge to the high fitness area or Pareto front. To meet designers' robustness criteria, neighborhood of each solution has to be explored until a robust solution is found. This requires a lot of additional work and is very difficult to find the most robust solution in practice. In the completely dominant definition, we can extend the vector $[\varepsilon, \varepsilon, \ldots, \varepsilon]$ to a more general vector $[\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n]$. That means we can have different tolerances for different objectives.

4.5 Conclusion

In this chapter, we have presented the completely dominant Pareto method selection method for genetic algorithms. This method relaxes dominance condition so as to relax comparison selection condition. Several genetic algorithms with other selection methods have been compared with the completely dominant Pareto method for a simple two objective problem. The results show completely dominant Pareto GA makes population quickly convergent to HP areas and keep population well distributed at HP areas. CDGA is proved to be able to help locate the robust solution in HP areas. It is also an easily implemented general algorithm that does not require much preliminary knowledge. It is quite possible to combine fuzzy fitness idea with completely dominant Pareto GA to further cut HP search areas. Further research on applying this technique on constraints and other co-evolutionary algorithms could extend its applicability.

www.manaraa.com

Acknowledgement

This project was supported by John Deere Corporation. We would like to thank Brian Kellogg, Tye Conlan, Lary Williams and Mac Klingler of John Deere Dubuque Works for many helpful suggestions and comments

www.manaraa.com

CHAPTER 5. DIVERSITY AND ROBUSTNESS IN MULTIOBJECTIVE OPTIMIZATION

5.1 Engineering Multiobjective Optimization

Chapter 4 proposed a new Genetic Algorithm – CDGA and used a simple two objective optimization problem to show CDGA's performance on convergence, diversity and robustness. This chapter accesses robustness and diversity performance of CDGA via simulation studies. Three more examples will be presented to do some deep analysis. One of them is a standard multiobjective optimization testing problem and has been applied to different GAs to compare GAs' performance. The second one is an engineering design problem with robustness concern. The third one is the brake system design presented in chapter 3.

The example in chapter 4 is specially designed to demonstrate CDGA's performance on convergence and robustness. This chapter focuses on comparison between CDGA and other GAs and robustness problems met in real engineering design problems. Detailed analysis is provided to demonstrate CDGA's performance.

5.2 Examples

5.2.1 Example 1 – T_6 Problem

To minimize two objective functions f when x_i belong to [0 1], i = 1, …, m. The test function and some empirical results are presented in [21, 22].

$f_1(x_1) = 1-\exp(-4x_1)\sin^6(6\pi x_1)$

$f_2(\mathbf{x}) = g(x_2, \ldots, x_m)h(f_1(x_1), x_2, \ldots, x_m)$

where

$$g(x_2, \ldots, x_m) = 1+ 9*((\sum_{2}^{m} x_i)/(m-1)^{0.25}$$

$$h(f_1(x_1), x_2, \ldots, x_m) = 1 - (f_1/g)^2$$

m=10

www.manaraa.com

Characteristics of Example Problem

The example is a simple two objective optimization problem with ten parameters. Variables $x_2, \ldots, x_m$ only affect the objective function f_2. The Pareto front is non-convex and formed with $g(\mathbf{x}) = 1$. This test problem has been used for comparison of different evolutionary algorithms due to its two difficulties. One is that it has a nonuniformly distributed Pareto Front (it is biased where f_1 is close to 1). The other is that density of solutions is lowest closest to the Pareto Front [22].

Genetic Algorithms

A multiobjective genetic algorithm with three different fitness functions has been applied to this problem. The only difference between them is how they assign fitness and select the next generation. The genetic operators and parameters are listed in table 1. The non-uniform mutation method selects one variable randomly and sets it equal to a uniform random number generated from a range decreasing with generation [14, 16].

Table 1 Genetic operators and parameters

GA type	Floating point
Initial Population	100
Mutation	Non-uniform
Crossover	One point
Selection	Tournament
Termination	250 generations

(1) Pareto Multiobjective Optimization

The selection for the Pareto multiobjective GAs is the Pareto dominance. If gene A dominates another gene B, A's fitness is high than B's. Figure 1 shows only the Pareto multiobjective GA results that fall into the area where $f_2<1.6$. Only a few results are located at the Pareto front or close to the Pareto front. The Pareto genetic algorithm doesn't use diversity technique such as crowding and sharing. The reason for this is to compare with the completely dominant Pareto GA for the same conditions.

www.manaraa.com

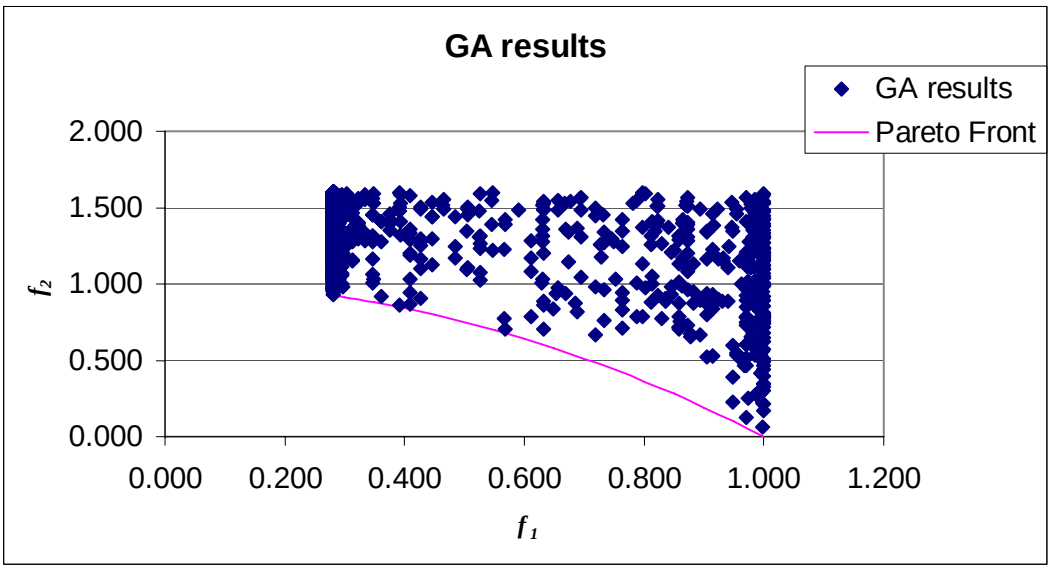


Figure 1 Objective results of the Pareto GA

(2) Fuzzy Fitness Functions

Using the fuzzy fitness function defined in figure 2, the satisfied values for *f1* and *f2* are set to 1 and 1.2 respectively and the satisfied weighted sum of *f1* and *f2* ($f_1 + 0.8\, f_2$) is set to 1.2. This fitness function is chosen based on preknowledge of the Pareto Front so that it is designed to drive the search to the Pareto Front and not lose much diversity. Figure 2 shows the GA results that fall into the satisfied fuzzy region. More solutions are found at the Pareto front than Pareto GA. However, the whole HP area is not well distributed.

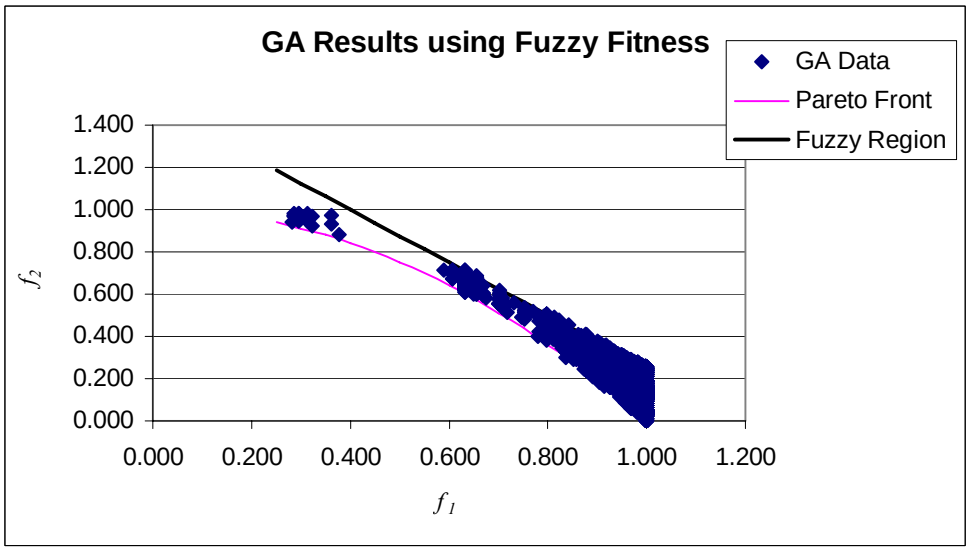


Figure 2 Objective results of the fuzzy fitness GA

www.manaraa.com

(3) Completely Dominant Pareto space

The tolerance in the complete Pareto dominance is set to [0.1 0.1]. If gene A's objectives is Pareto dominant to gene B's objectives minus the tolerance, gene A is completely dominant to gene B so that gene A's fitness is better than gene B. More population are falling into $f_2 < 1.5$ areas than Pareto GA's (figure 3). The population along the Pareto front distributes well. The whole population shows an excellent convergence and diversity. Diversity and convergence measures are shown in the following section.

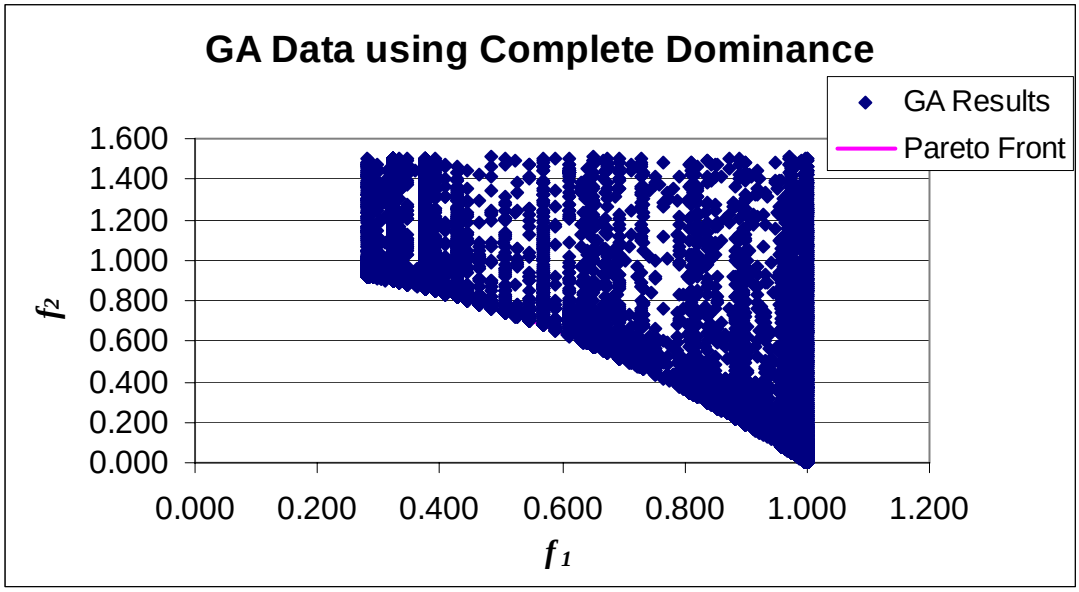


Figure 3 Objective results of the completely dominant Pareto GA

Discussion

Simulation has been repeated five times for each case and similar results appear. From the simulation, both the fuzzy fitness GA and the completely dominant Pareto GA relaxing selection rule are push more population into HP areas. However, the fuzzy fitness function requires that designers are knowledgeable about the system to design a suitable fitness function. It is arguable that designers could have run a number of simulations to become familiar with the system. Main problems that fuzzy fitness will face are a larger HP space to search, which increases with problem dimensionality, and need of diversity techniques. It is hard to design a fuzzy fitness function to push population distribute evenly. Figure 3 shows that population are denser near f_1=1 areas. The completely dominant Pareto GA can successfully make the population converge to

www.manaraa.com

HP area while keeping diversity. It is intuitively appealing to designers. Designers can have a clear picture about the range of objectives of design in HP areas.

Table 2 shows a comparison of GA performance between NSGA II Real_coded with Completely dominant GA Real_coded. Two metrics defined in the paper [21] are used to measure convergence and diversity performance. As table 6.1 shown, CDGA has a better convergence performance (γ metric) than NSGA II for T6 testing problem ($p<0.01$). The diversity metric (Δ metric) is almost in the same level for both algorithms ($p = 0.25$).

Table 2. Comparison of GA performance

Algorithm	γ metric		Δ metric	
	Mean	Variance	Mean	Variance
NSGA II real-coded	3.38	0.13	0.668	0.010
CDGA	1.75	0.25	0.678	0.014

The Pareto GA without diversity control did very poorly on this special problem. [21] has shown that some multiobjective GAs with sharing techniques can improve searching. However, any sharing techniques are computational expensive. Completely dominant selection allows equal producing child opportunity in HP areas at each iteration. In this way, even without any additional diversity control, it shows a good performance on such nonuniformly distributed test problem. It is easy for designers to do additional exploration for design because they can define HP areas and solutions in HP areas are nonbiased.

www.manaraa.com

5.2.2 Example 2 – Simplied Engine Control Design

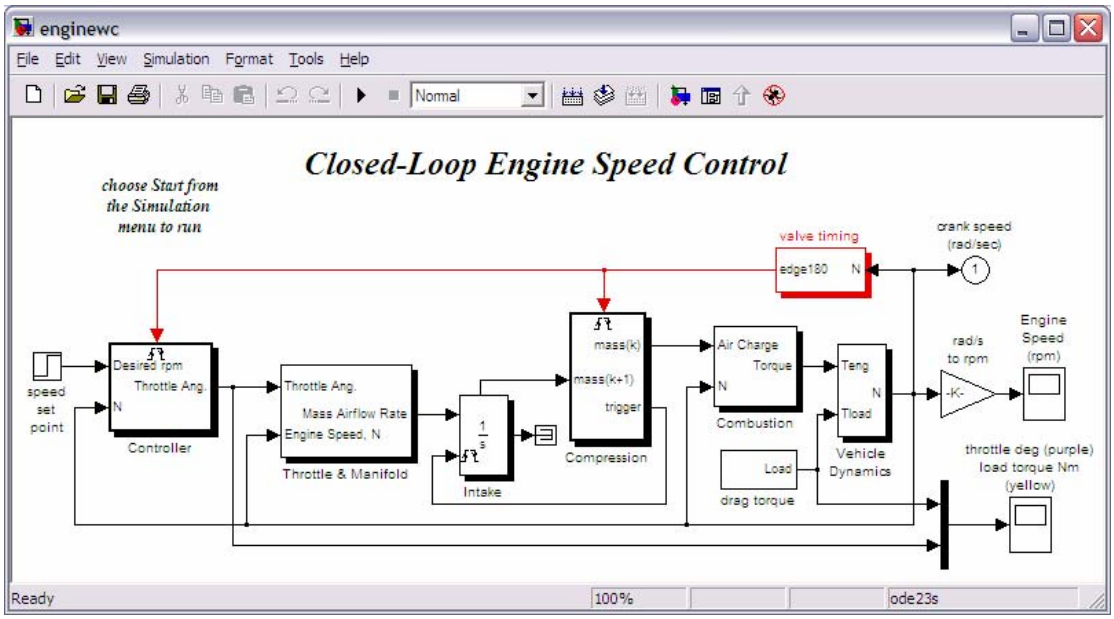


Figure 4. Engine Control Design model

This is a closed loop engine speed control model. The engine dynamics is simplified to a function of throttle angle, start of injection, engine speed, load, and inertia. The three design variables are two control gains and fuel injection time. The objective of the controller design is to minimize two kinds of tracking errors: stable engine speed state tracking and accelerating state tracking. For stable engine speed condition, it is desired that the engine speed control model has a small steady state error. For the second case, it is desired that the engine speed control model has a quick response time to follow acceleration trajectory. Since the engine always has disturbance during running, such as inertia and load changing, the design also wants to meet robustness requirement to overcome any disturbance in torque feedback.

The engine dynamics are modeled in Simulink. The complicated dynamic engine has no explicit function to describe relationship between objectives and design parameters. To get the design objectives for each combination of design parameters, designers have to run simulations for 20 seconds to calculate the tracking errors. If using the traditional design methods, this is very time-consuming to find out the right gains and the fuel injection time given the contradicting objectives, and the robustness concerns. However, with the help of the complete dominant GA methods plus automated software designed in this dissertation, it is easy to solve this multiobjective optimization problem.

www.manaraa.com

Our Engineering Design Genetic Algorithm software (EDGA, see appendix) will manage the evolving process of genetic algorithm by controlling simulation. It automatically runs everything; include Simulink model simulation, population evolving process, and optimization results management after the model is hooked up with the software. All genetic algorithm operators can be set in the software.

This is a multiobjecitve optimization problem with three design parameters and two objectives. Since there is a need to meet the robustness requirement, another uncertainty parameter – torque disturbance is included in the design parameter set, assuming that the torque disturbance is changing in [0 2] range. In addition, since there is no uncertainty for control gains, we multiply a scale 100 to the control gains so that any uncertainty in [0 1] range is ignored. For injection time, engine system can only achieve one-degree accuracy, so we add half-degree uncertainty in the search of optimal design. The GA setting is listed in table 3.

Table 3 Genetic operators and parameters

GA type	Floating point
Initial Population	32
Mutation	Non-uniform
Crossover	One point
Selection	Tournament
Termination	100 generations

The tolerances for two objectives are set as 0.12 and 0.07 respectively. The robust point found by Completely dominant GA is 0.18 for Proportional gain, 0.12 for differentia gain, 35 for fuel injection angle. This design point is located on the Pareto Front and has the most robust resistance to the torque disturbance. If the torque disturbance is changing between 0 to 2 KN*m, the two objectives are bounded by [±0.12 ±0.07] respectively. If we want to further minimize the objective variances, the torque disturbance range has to be reduced. In another word, the disturbance range directly affects the variance of the objectives. For designers, it is clear that what sacrifice in objective values they have to bear to overcome the disturbance. If the accepted variance for the objectives could be larger, there would be more robust solutions that could be picked.

www.manaraa.com

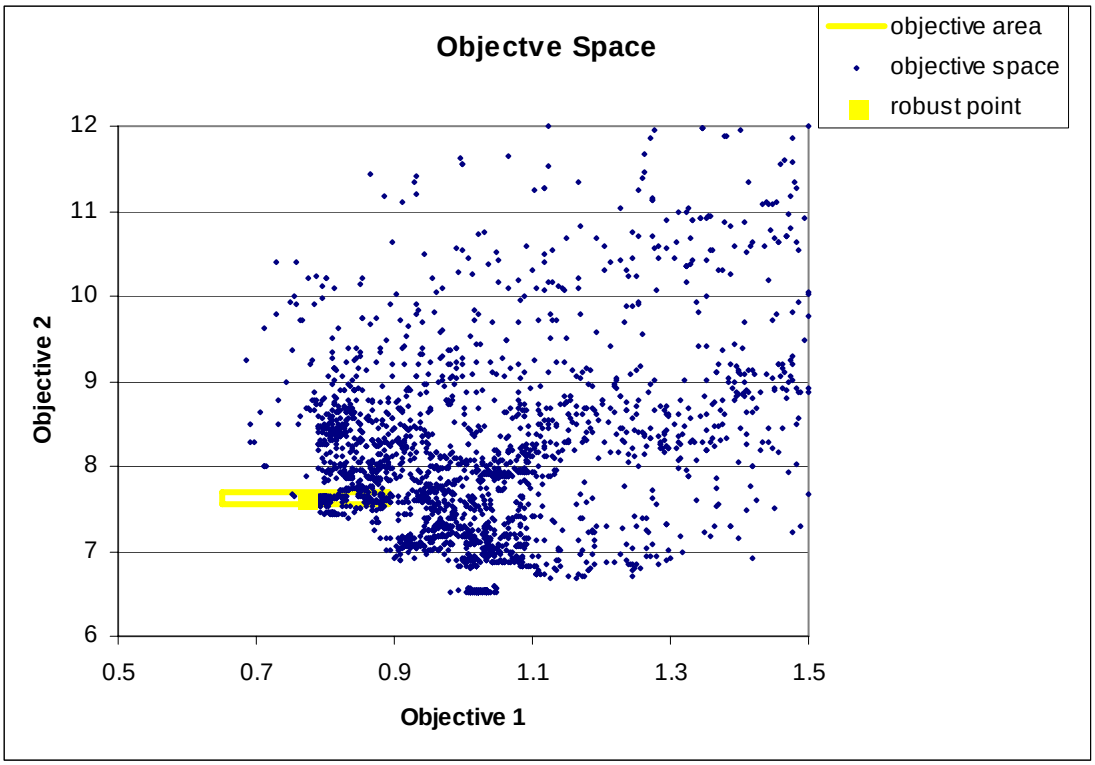


Figure 5. Objective Space

For the same engine optimizastion problem, if engine characteristics are changed, i.e. the optimization function is changed while objective functions are still the same. Applying the same setup for CDGA, figure 6 shows the objective space for the new engine type. The Pareto Front is moved to a different area due to engine model's change. Optimal PID gains and SOI are also changed to different sets. To achieve the same objective tolerance target, the optimal and robust PID gains and SOI are 0.32, 0.47, and 0, respectively. However, for this type of Engine, our disturbance to torque is only under 0.25 kN*m range if the same objective tolerance is used. That means we can not achieve robustness requirments for this type of Engine. To realize the robustness requirement, we need to either loose robustness requirement or loose optimum target. In another word, it is possible to sacrifice optimum to achieve robustness for this engine controller design.

www.manaraa.com

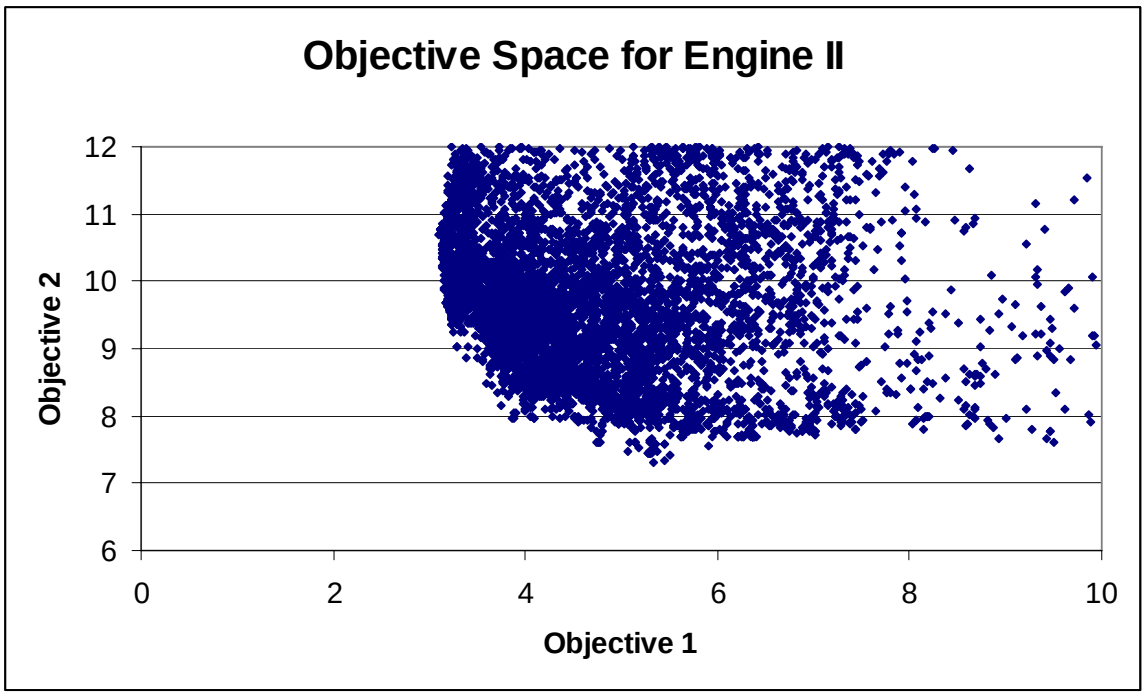


Figure 6 Engine II Objective Space

5.2.3 Example 3 – Brake System Design

CDGA is also applied to the Hydraulic Brake Design problem mentioned in Chapter 3 to study design robustness. As complicate the problem is, there are many robust designs along the Pareto Front. Figure 7 shows two different robust solutions found by CDGA. Some design variables are located in the same range such as design variable 2, 5, and 7, while some of design variables are far away. These two solutions are located in the Pareto Front and have almost the same robustness performance, which tolerate 10% disturbance for any design variable in 0.1 objective range. In another word, if any less than 10% disturbance occurs in any design parameters for these two design combinations, the objectives are guaranteed to be varied less than 0.1. Using the first design variable as an example, the design table can have 10% disturbance and do not cause large objective variation. The table is mapped to one design parameter using ratio between the lower bound and the upper bound. Figure 8 shows the curve of the first design table and its robust design range.

In a word, CDGA finds two robust solutions for such complex hydraulic design problems. If the worst disturbance for design parameters is less than 10%, the robust solutions can guarantee the objectives of design to be vary in less than 0.1 in its Pareto Front area.

www.manaraa.com

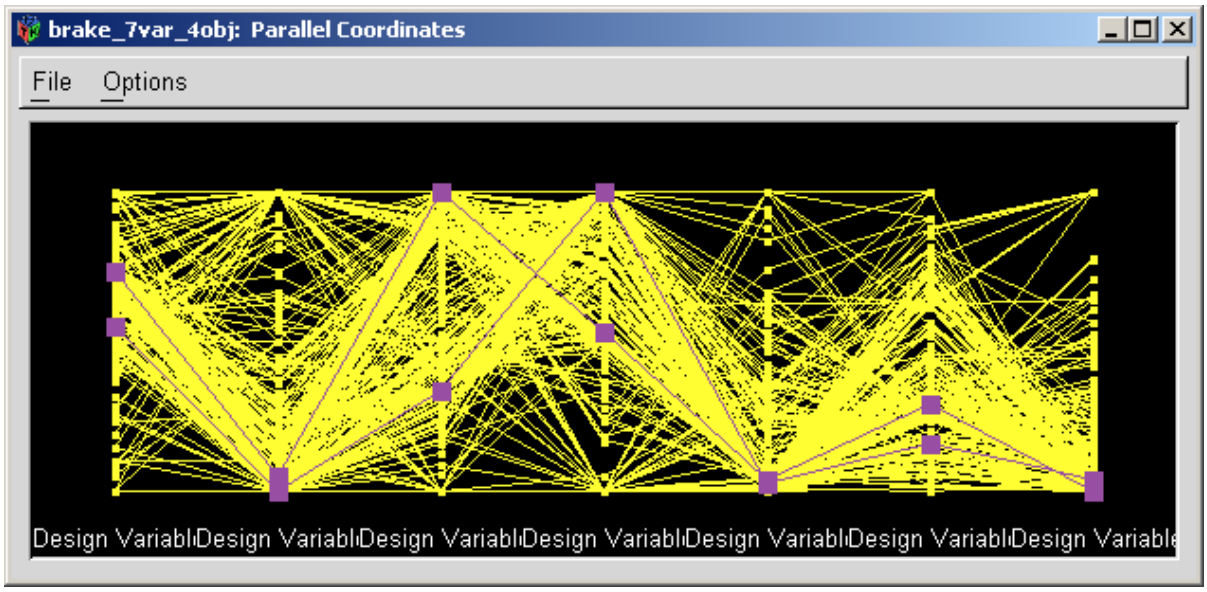


Figure 7. Parallel Plot showing robust design

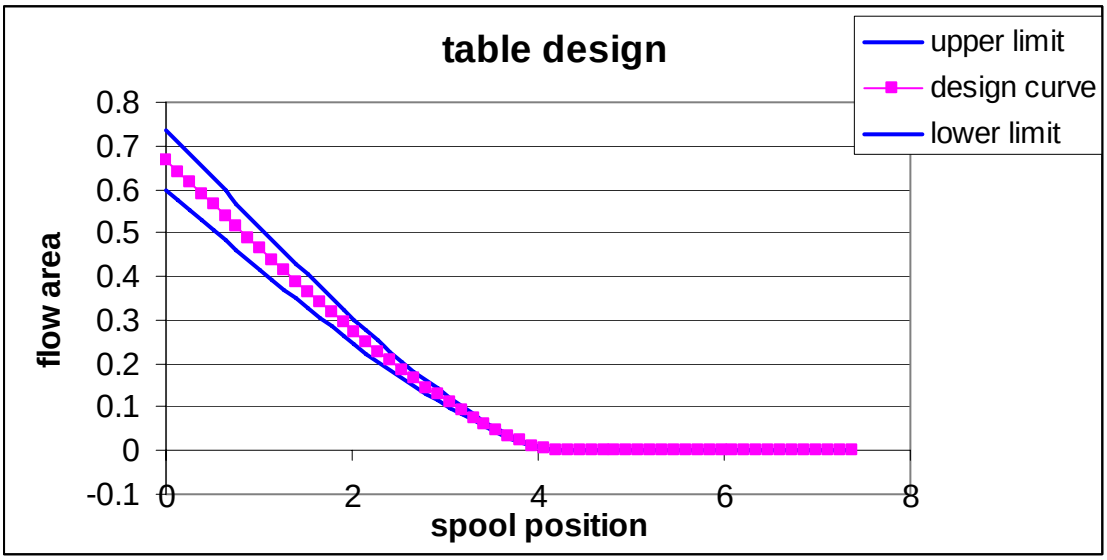


Figure 8. Design Table 1 range

5.3 Discussion

The first example is a standard testing problem for multiobjective optimization. CDGA has shown significant improvement in convergence and achieved the similar diversity comparing with NSGA-II. The other significance of CDGA is its simplicity. It does not have a complicated diversity control techniques, which normally require tremendous additional search. Combining with certain diversity control techniques is a

www.manaraa.com

possible future research area. However, since CDGA is mainly for multiobjectiv optimization problems with robustness concern. How diversity techniques affect locating robustness requires more deep studies.

The second example shows a practical engine control design problem with robustness concern. Completely Dominant GA has successfully searched the design space and located a set of design parameters that not only makes the objectives closed to the Pareto Front, but also gives a minimal boundary for objective variances during disturbance. The clustering search verifies the boundary is minimal and it could be smaller in reality. This is very important for designers because they will have confidence on their controller design. The objectives for the worst case are bounded in a certain tolerance. CDGA does not need complicated robustness estimation techniques and automatically makes the final population converge to the robust area.

The third example is the restudy of Brake System presented in Chapter 3. We focus on robustness study of Brake System optimization, especially that this example involves curve design. CDGA has shown that it is able to locate robust design area for such complex system and is capable to deal with curve design as normal parameters. Future research on applying CDGA on constrained optimization problems especially constrains in parameterizing curves.

This chapter has shown CDGA's performance on convergence, diversity, and robustness using different testing problems. CDGA has a superior performance on convergence and robustness. Its diversity performance is comparable with other Pareto-based Genetic Algorithms with diversity control techniques. Since the examples used here are all two objective optimization problems, high dimensional objective problems are needed to further explore diversity in the future.

www.manaraa.com

CHAPTER 6. INTERACTIVE GRAPHICS FOR ENGINEERING DESIGN INVOLVING DYNAMIC EQUATIONS AND GENETIC ALGORITHMS

6.1 Introduction

The general engineering design process involves modeling, simulation, and evaluation [4]. Complex engineering design problems are normally high dimensional and multiobjective and are modeled by complicate non-linear dynamic equations. It can be very time-consuming sometimes impossible to find optimal designs by solving dynamic equations. Several optimization methods have been applied to the engineering design problem to find the optimal design automatically, such as simulated annealing and Tabu search [1]. Of these optimization algorithms, evolutionary methods have become popular because they are gradient-free and generally produce good results [13]. Genetic algorithms (GAs) have become a commonly-used optimization tool for design engineers especially for multiobjective optimization problems. GAs can extensively and efficiently search the design space to find an optimal and robust design combination that meets design objectives.

Multiobjective optimization means that the optimization problem has several objectives to meet. Sometimes, objectives are conflicting each other, i.e. if one objective becomes better, another one will be worse. The optimality of the multiobjective optimization is defined as Pareto optimality [11]. Dominance is an essential concept in Pareto optimality. If a solution **A** is said to dominate another solution **B**, it means that all of **A**'s objectives are better satisfied than **B**'s. The set of non-dominant solutions among all possible solutions is called the Pareto front. The goal of multiobjective optimization is to find a trade-off solution on the Pareto front.

Genetic algorithms (GAs), developed by Holland [8], are one of the most common evolutionary algorithms. They are inspired by natural selection and survival of the fittest. GAs are powerful and robust stochastic search and optimization techniques, which have been applied to many engineering and mathematical areas such as engineering design [4, 6] and stock investment [3]. GAs can solve complex problems that are difficult to solve with conventional techniques, such as gradient approximation methods. To find an optimal design or locate the Pareto front requires that GAs extensively explore feasible design hyperplane regions. The difficulty with using GAs in multiobjective optimization

www.manaraa.com

is understanding the solutions as design dimension increases. The Pareto front will form a complex hyperplane. Furthermore, nonlinear complex dynamics of system will result in complicated interactions between input and output. It is hard for designers to have an overall picture of the Pareto Front.

GAs are beneficial for engineering preliminary design because they are capable to explore High Performance area completely and quickly [5]. In this design stage, design engineers are more interested in interactions between input and output than finding an optimal design. Data analysis and visualization in high performance regions will help engineers better understand multivariable interaction between design variables and objectives.

Visualization is the visual representation of information (data sets, geometry models) using graphics, image, or animations. Human's perception and cognition to visual effect enable them to rapidly obtain insights in data, such as relationship, clusters, and trends. Visualization can help engineers to understand the evolutionary process and underlying relationship between input and output. Traditional visualization methods for GAs [14, 15] generally show simple graphs displaying fitness or objectives versus generation time as shown in figure 1. These graphs show overall convergence information. However, design engineers need to know more about the relationship between input and output, relationship between different objectives, and robustness of design, etc. This work explores methods for visualizing data from evolutionary. Background information for engineering evolutionary design is provided in section 2. Section 3 presents the modular structure of visualization software and visualization techniques. Section 4 focuses on using visualization to present useful information for design engineers, and section 5 concludes the paper

www.manaraa.com

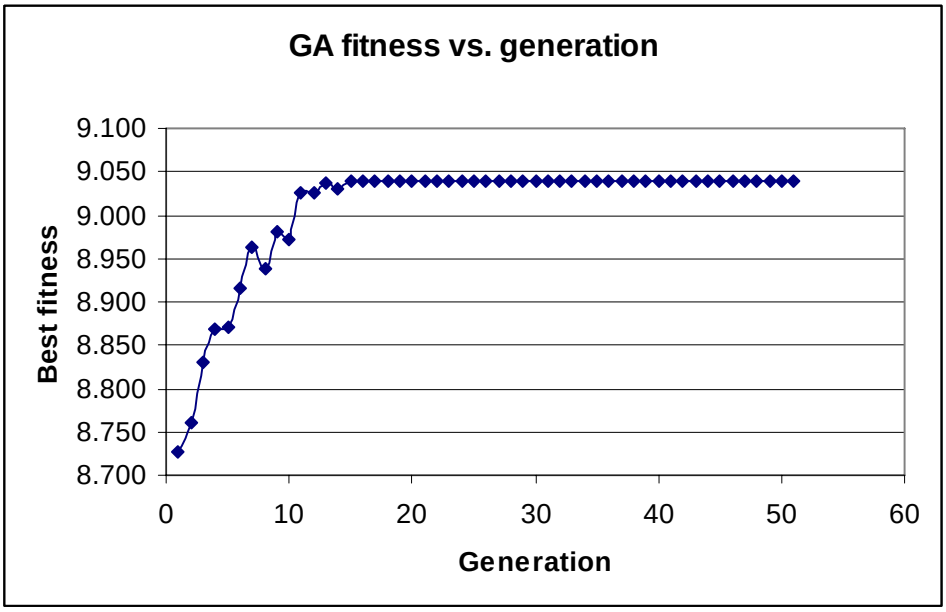


Figure 1 GA fitness evolution process

6.2 Engineering Evolutionary Design using Genetic Algorithms

GAs start from a set of initial population. At each generation, the weaker ones in the population (low fitness) tend to die and the stronger ones (high fitness) tend to produce children by crossover and mutation [7]. As generation goes on, the final population will contain a lot of high fitness individuals, i.e., optimal design combinations. The general procedure for GAs is as follows:

Set up binary or real number data structure (chromosome) representing design combination

Create an initial population

Evaluate solutions in the population

Repeat

Select solutions to produce offspring

Produce new solutions from parents' chromosome by mutation and crossover

Evaluated new solutions and put them into the population

Until stopping condition met

GA uses fitness to evaluate solutions and select the next population. Fitness is a function of objectives. It reduces high dimensional objectives into one dimension. Some common fitness functions are weighted-sum fitness function, fuzzy fitness, and

www.manaraa.com

dominance rank [1, 5]. The example in this paper is using weighted-sum fitness function, which is a linear mapping from objectives to fitness $\sum weight_i * objective_i$.

Parmee has proposed Cluster Orient GA (COGA) to help engineer do preliminary study. The basic idea is to keep GA converge to high performance (HP) area quickly, which is an area close to the Pareto front. The high performance clusters can help design engineer further understand system such as input and output interactions and design variables sensitivity. COGA uses variable mutation rate to allow diversity in the final stage so that it can formulate high performance clusters. In order to prevent low fitness solutions from falling into clusters, filters, which are using a threshold to filter low fitness solutions, have to be used in evolutionary process. Engineers face bigger challenge when using COGA for preliminary study because more results are needed to be studied.

6.3 Visualizing GA Output

Designers build models to simulate system performance. Genetic algorithms (or other evolutionary algorithms) control models to generate a lot of simulation runs in evolutionary process. The results from the genetic algorithm are stored in a database along with the search parameters for the potential solutions. Any visualization can be constructed through database connection.

The normal data from GAs are chromosomes, objectives, fitness, generations, parents, and life time. For engineering design, the chromosomes are formulated from design variables. There are two kinds of design variables: single parameters and look up tables. Single Parameter is a numeric value such as PID gains. Look-up tables are 2D curves or 3D surface to model nonlinear characteristics of a particular system. Each design parameter is one chromosome in an individual gene. For tables, there are several techniques to construct a curve such as interpolation and parameterization. For example, if using interpolation method, a table can be represented by one ratio variable. The table values can be recovered by a conversion as follows:

Table=min_table+(max_table-min_table)*ratio

where min_table and max_table are tables on low boundary and high boundary, respectively. No matter what technique uses, a table can be represented by several real number chromosomes.

www.manaraa.com

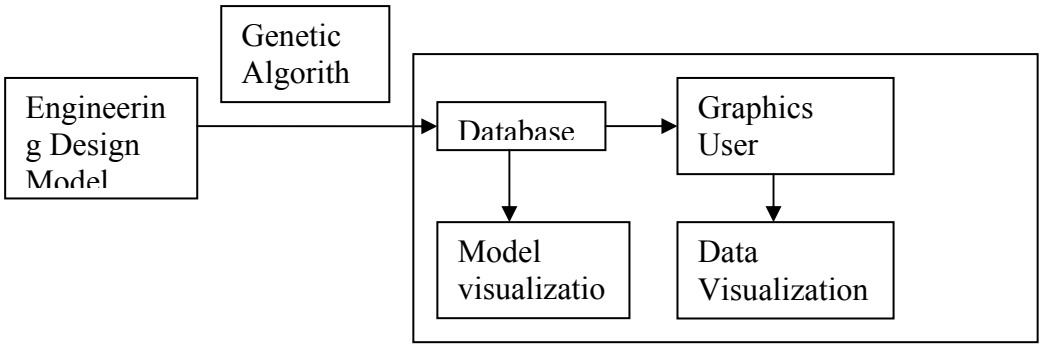


Figure 2 Visualization Software Structure Diagram

Data Visualization is normally showing high dimensional GA results into 2D/3D dimensions. Typical plots include time series plot, scatter plot, and parallel coordinate plot. Time series plot generally is used to show GA's convergence progress such as fitness vs. generation. Scatter plot shows correlation of two variables. For high dimensional data, scatterplot matrix is able to show data in pair dimensions. Projection method is able to change high dimension data into 2 dimensions so that data can be viewed in scatter plot. Parallel plot presents a series of data with line connections between data. The x axis is a set of parallel axles for each variable. It is highly effective to show input combinations. Most of the existing visualization software packages support all plot types. Some of them support dynamic visualization, which means that users can interact with visualization and look at the same data area in different plots. In the following paragraphs, GGobi is used to show how to use dynamic visualization to present GA results. The unique feature in GGobi is its Grand Tour method that will be described later.

For GA results, we are trying to visual multivariable relationships, such as:

1) Input space: How well did GAs cover the possible inputs? Are there big empty spaces in the inputs, combinations that were not run at all? Maybe this suggests doing a few manual runs of these combinations to check that it is not important for getting good output.

2) What is the relationship between multiobjective function and the fitness functions? Are all the objective functions optimized when fitness is optimized, or what are the trade-offs between objective functions.

3) What is the relationship between fitness and inputs? Does good fitness correspond to small neighborhoods in the input space?

www.manaraa.com

Exploring mulit-variable relationship is important to understand complex system dynamics and nonlinear system characteristics. For example, if a multiobjective optimization problem is converted to a single fitness function problem, the choice for the fitness function is critical to optimization. If an undesired fitness function is chosen, the GA optimization will be pushed to a wrong direction so that the final optimum may be away from the Pareto Front. Visualization is capable to help design engineers to explore the design space so that it will help design engineers to choose a right fitness function.

6.4 Examples

In order to present these techniques, a real industrial backhoe design problem is used as an example. This design problem has 7 variables to design and four objectives to meet. Detailed problem description can be found in paper [6]. The design variables are marked as Design Variable 1 to 7 and the objectives are marked as Objective 1 to 4 respectively. The fitness is the function of four objectives ($5 - \sum objectives$). The design goal is to maximize the fitness or minimize all objectives in multiobjective sense. GA has been used to optimize the design and GA results have been put into a database.

1) Conventional Visualization

Conventional visualization, which uses 2D plots, can be easily constructed through database such as best individual vs. generation and objective convergence. Common techniques that have been used in presenting high dimensional evolution data include using Dimension Reduction, Color graph, Glyphs, Parallel coordinate, and Projection [1, 16]. Most of the existing techniques use the static plot to present evolutionary data, which is good to show evolution process but hard to explore multidimensional relationship. For engineering designers, the underlining relationship between design variables and design objectives is more important than evolutionary process. Figure 3 is a typical static plot suggested in [12]. It shows High Performance points of Objectives 2 and 3 in hyperplane Design Variable 3 and 6. Objective 2 and Objective 3 are not exclusive in Design Variable 3 and 6 dimensions because they have overlapping areas. However, conventional static plots cannot show the multivariable relationship efficiently. It is hard to know relationship of Objective 2 and 3 in 7 design variable dimension and variable sensitivity in full dimensions. In addition, all combinations of any two design variables

www.manaraa.com

have to manually set up to fully explore interactions. The following visualization techniques presented are all dynamical and interactive. Designers have more freedom to look at the data in high dimension and find the information much more quickly.

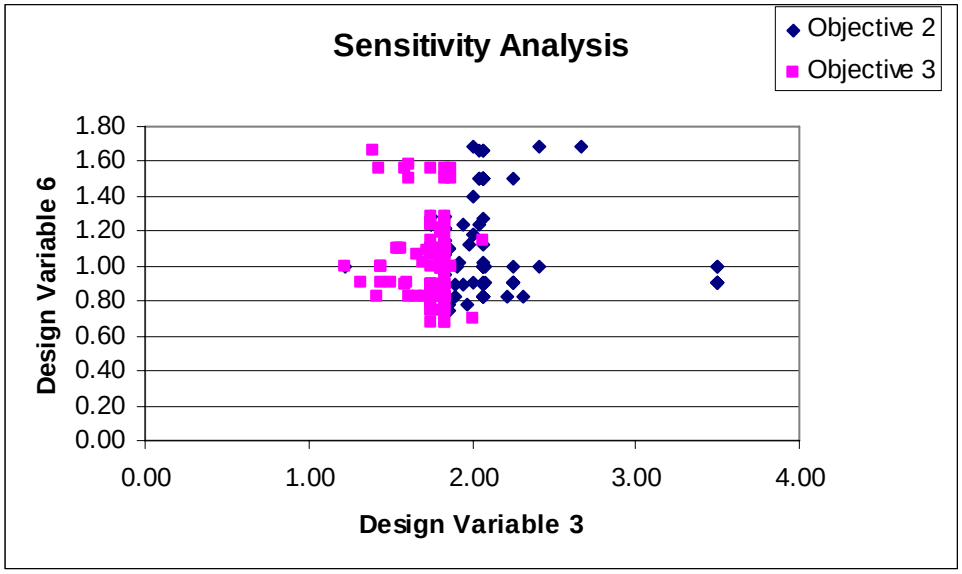


Figure 3 Sensitivity Analysis of Design Variable 3, 6

T6 Optimization Example:

T6 optimization problem is a popular testing case for evolution algorithm [18]. It is to minimize two objective f_1 and f_2 defined as follows.

$f_1(x_1) = 1\text{-}\exp(-4x_1)\sin^6(6\pi x_1)$

$f_2(\mathbf{x}) = g(x_2, \ldots, x_m)h(f_1(x_1), x_2, \ldots, x_m)$

where

$$g(x_2, \ldots, x_m) = 1 + 9*((\sum_{2}^{m} x_i)/(m-1)^{0.25}$$

$$h(f_1(x_1), x_2, \ldots, x_m) = 1 - (f_1/g)^2$$

m=10

If we assume the objective functions are not known, it is hard to interpret the relationship between objectives and inputs using static plots due to high dimension. However, dynamic visualization techniques are able to easily present the relationship. As shown in figures 4 and 5, as we brush along the Pareto Front in objective space, the input combinations are highlighted in the Parallel plot. When obj1 (f_1) is close to 1 in the Pareto Front, $x_2 - x_{10}$ are all close to the lowest value (0). X_1 is spreading along its range but forms several clusters, which suggests us that the function between x1 and f_1 is

www.manaraa.com

multimode. If we brush the points along the f_1=1 area, it is easy to identify that x_1 is independent with f_2.

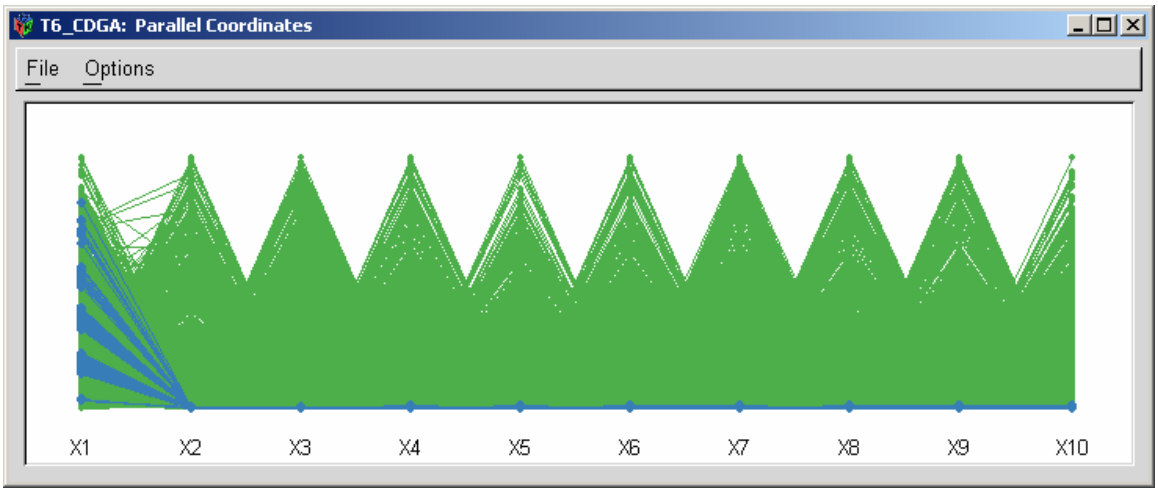


Figure 4 Parallel Plot for inputs

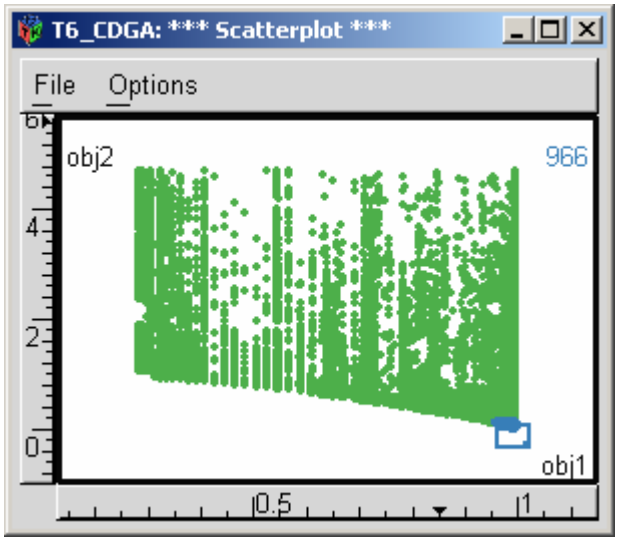


Figure 5 Scatterplot for objectives

2) Interactive visualization

Design engineers not only want to optimize system performance, but also want to understand the relationship between input and output in order to improve system design. Typical questions design engineers want to study are how each input affects system performance, what design combinational is optimal and robust, and what design combinational results in bad performance.

Parallel and scatter plots can be used to show the connections among input variables and high dimensional output. Figure 6 is the parallel plot of three design variables. In a parallel plot, each line represents a connection between two variables. Figures 7 and 8 are the scatter plots of two output variables. A technique called brushing

www.manaraa.com

is used to help engineers to link these plots and answer above questions using visualization. Brushing - is the process of selecting multiple data points, shown by a box, where the corresponding points in other plots identify themselves by changing color. As in figure 8, certain combinations of Objective 2 and Objective 4 are brushed. The corresponding points are highlighted in other two plots. The visualization shows that these combinations have very narrow range for Design Variable 1 and the other input variables have wide ranges. It suggests Design Variable 1 is sensitive to Objective 2 and 3 while Design Variable 2 and 3 are nonsensitive in this hyperplane area. It is easy to move brushing areas and look at variable sensitivity in other regions. Parallel plots provide a possible way to study high dimension interactions by putting several design variables in and also the input space covered by GA. The dynamical visualization provides quick and easy interactive method to look at input-output relationships.

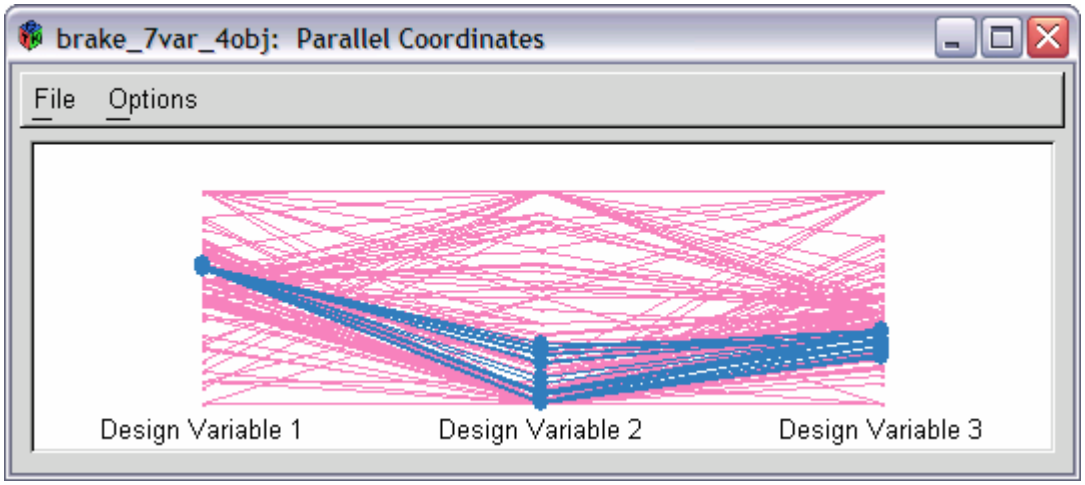


Figure 6 Parallel plot of Input design variables

www.manaraa.com

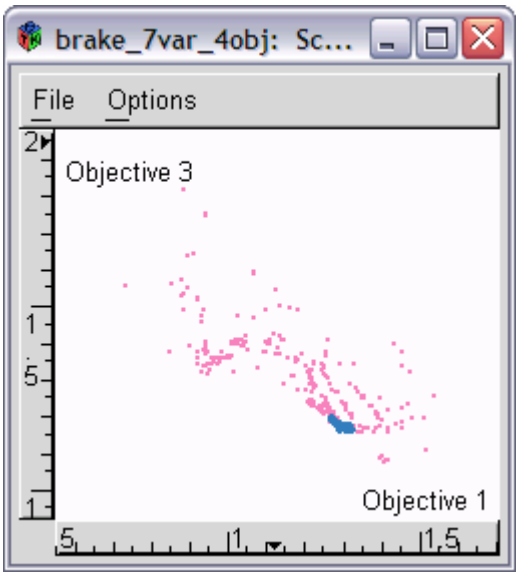


Figure 7 Scatter plot of output variables

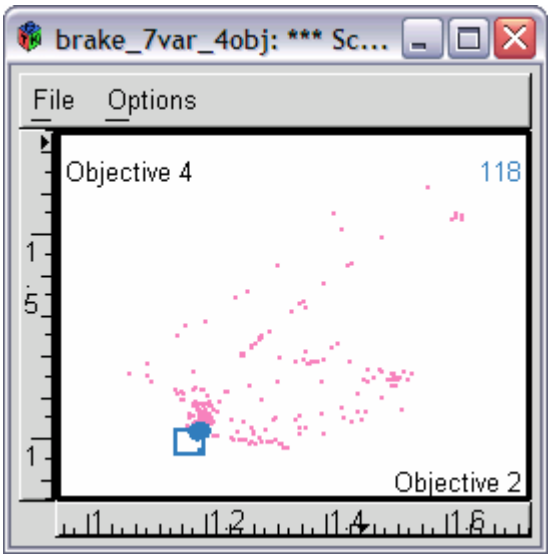


Figure 8 Brushing

3) Multiobjective Exploration

For multiobjective optimization problems, it is difficult for design engineers to design a desired fitness function to trade-off multiple objectives. For example, if using a weighted sum method in GA [7], choosing different weights will guide GA to different directions. Design engineers are interested in the Pareto front, non-dominant sets. If they have ability to explore the Pareto front, they can choose a trade-off solution themselves.

In our example problem, there are four objectives that can be represented in two 2D linked plots. Figures 9 and 10 are scatter plots of four objectives. They show some non-dominated solutions formulating part of the Pareto front in high dimension. Figure 10 is the fitness distribution plot. By brushing a small non-dominated area in objectives 1 and 3 space, we can see its corresponding fitness and its position in other objective space (figure 11). In such plot setup, we can not only have an understanding of interaction between multiple objectives, but also have an insightful knowledge of fitness vs. objectives. It is easy to remap objectives to a different fitness function and compare results in the same setup. Linked scatter plots provides an easy way to look at multiple objectives and study High Performance regions.

www.manaraa.com

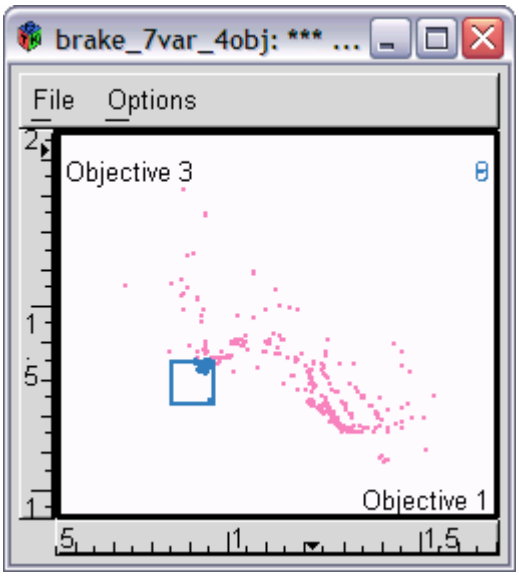


Figure 9 Scatter plot of Objectives 1 and 3

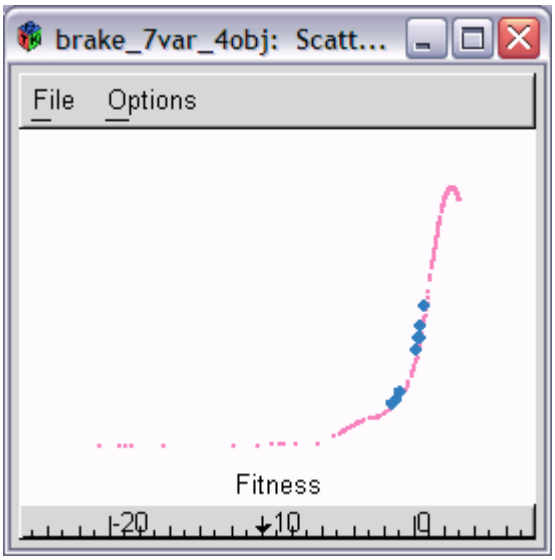


Figure 10 Fitness distribution

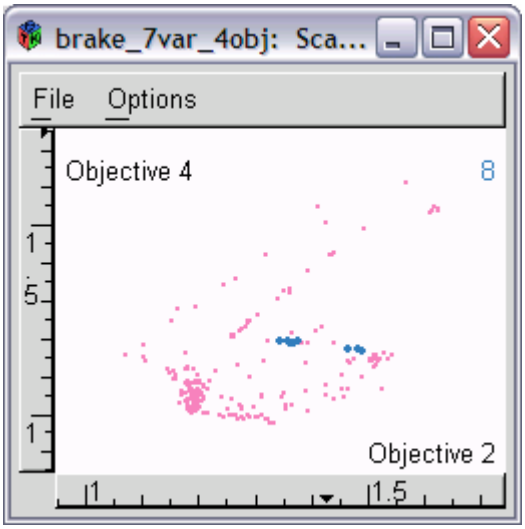


Figure 11 Scatter plot of Objectives 2 and 4

4) High dimensional visualization

For multiobjective optimization problems, high dimensional visualization can help understand multiple objectives' evolution. Traditional high dimensional visualization is using multiple scaling to map high dimension data to 2-D plot. The disadvantage is that multiple scaling [2] is itself an optimization problem and it rarely exists an optimal solution. Even if it showed the distance roughly correctly, it could not clearly show what solution is better and essentially what the Pareto front is.

The Grand tour is a dynamical view of high dimensional data [14] that maps high dimensional data into two-dimensional plot while changing the projection matrix continuously. Users see is an animation of data from different projection angles. The

www.manaraa.com

Grand tour not only shows the distance between points in a dynamical way and but also shows the high dimensional relationship between points. Appendix 1 shows continuous snapshots from Grand tour for best individuals in a four-variable and seven-variable space respectively. Combining with brushing techniques, they show that in four-variable space best individuals form a solid cluster while in seven-variable space they scatter away. It suggests additional dimensions (Design Variable 2, 4, and 6) are not sensitive as other four design variables. Grand tour is a powerful tool to show high dimensional projections and clusters.

5) Pareto Front

The Pareto Front is the set of non-dominant solutions. Since designers can only choose one solution as the final solution, the final optimal solution is often picked from the Pareto Front or High Performance Area. Visualization greatly helps designers to explore the Pareto Front and make the final decision. Multiple scaling and Grand Tour are not very effective in presenting the Pareto Front. The most effective way found is using dynamic visualization either in Parallel plot or in scatterplot matrix. The Pareto Front can be separated from all solutions first and then presented in visualization. Figures 12 and 13 show four-dimension Pareto Front in Parallel plot and scatterplot matrix respectively. Users can brush any area in one objective and find out the solutions' positions in other objective space.

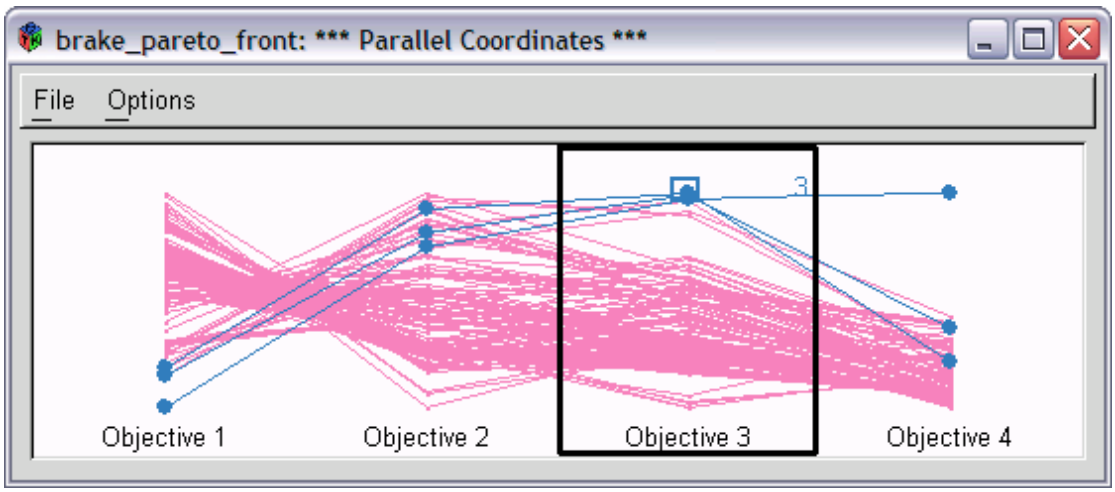


Figure 12 Visualization of the Pareto Front in Parallel Plot

www.manaraa.com

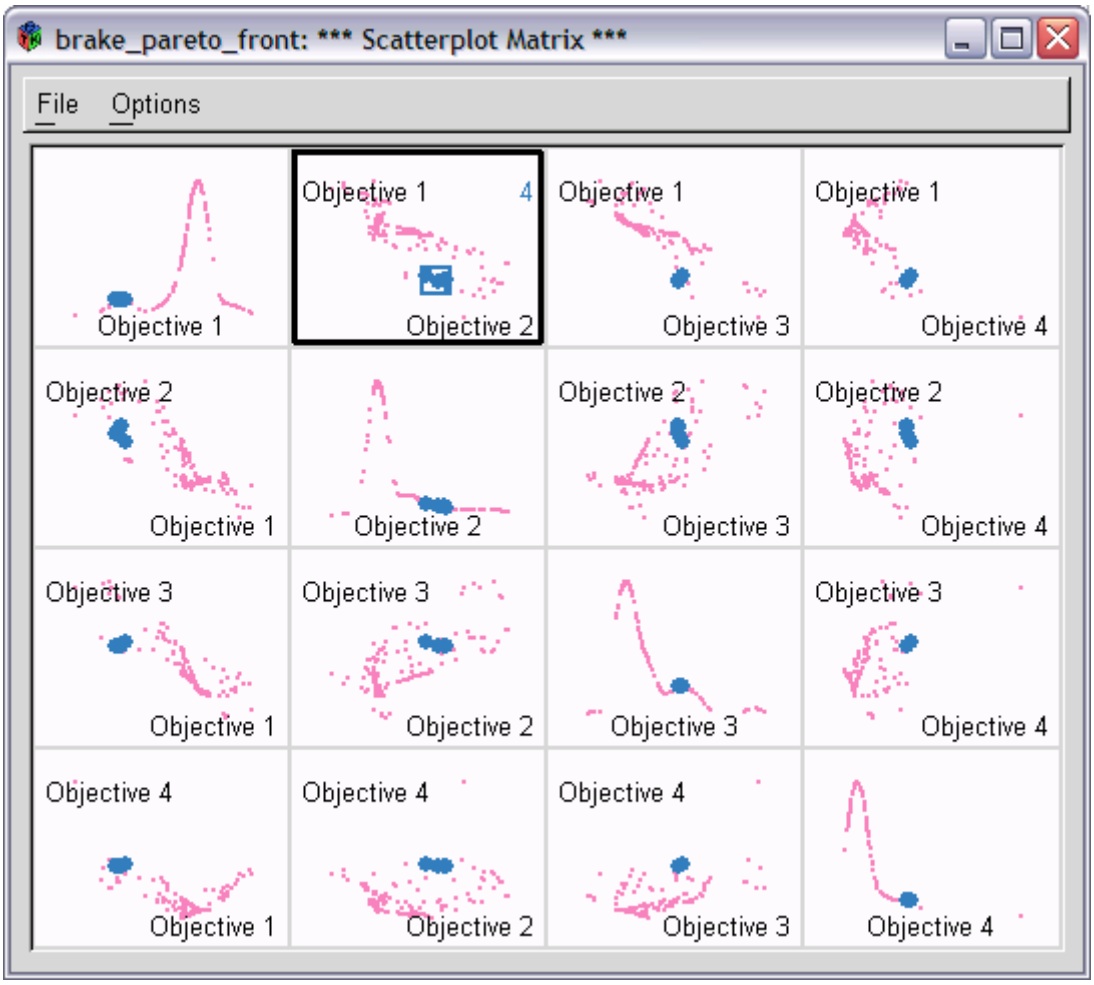


Figure 13 Visualization of the Pareto Front in Scatterplot matrix

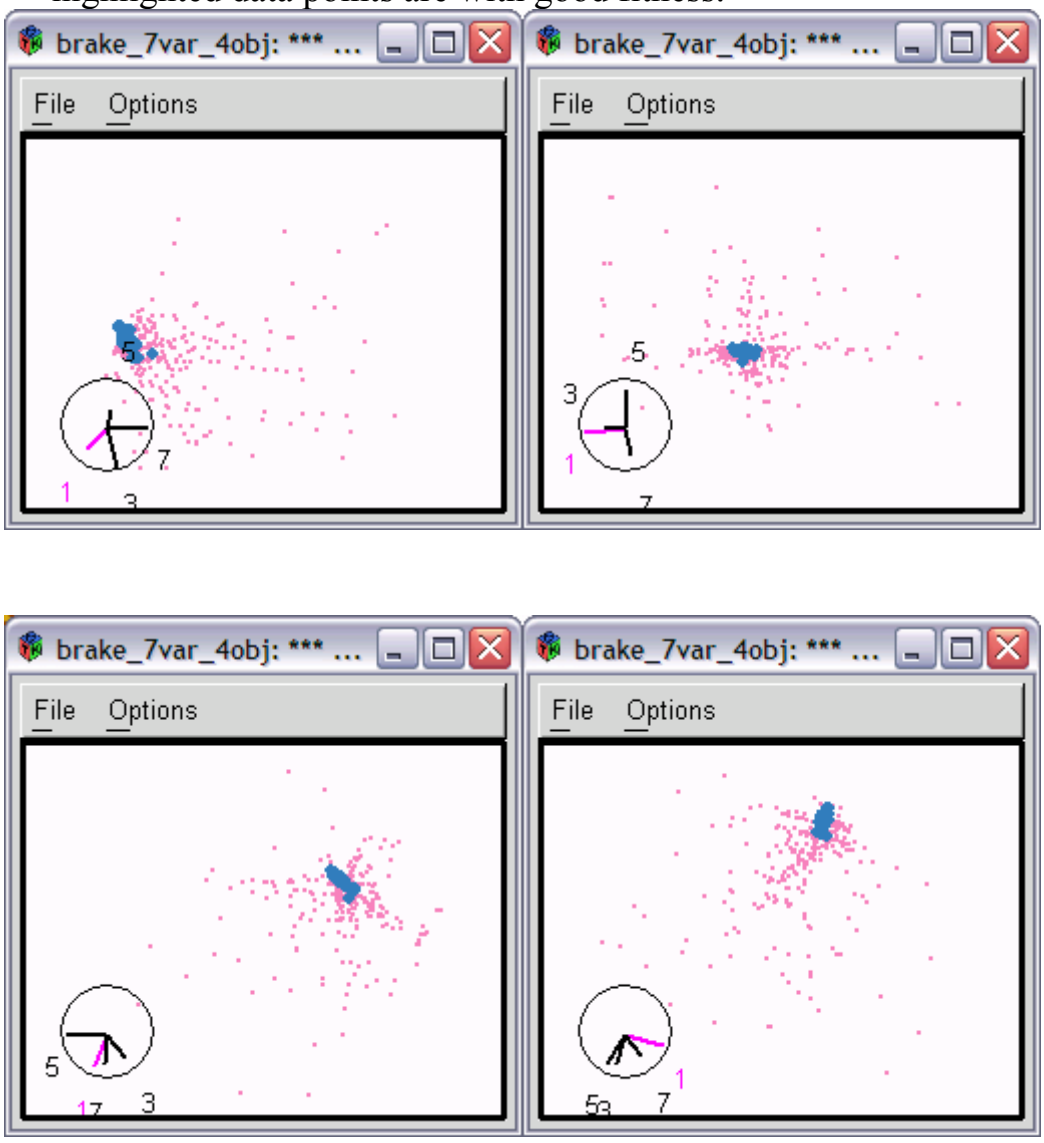
6.5 Conclusion

Engineering design data has fixed format because of its natural characteristics. It makes possible to design a generic interface for visualizing engineering design data generated by genetic algorithms. The generic software structure in the paper can be extended to any engineering design models and other evolutionary algorithm data. The multivariable relationship of design variables and design criteria is needed to be fully explored for designing a complex system. The dynamical and interactive visualization techniques presented in this paper can help engineers explore the data quickly and identify unusual pattern easily.

www.manaraa.com

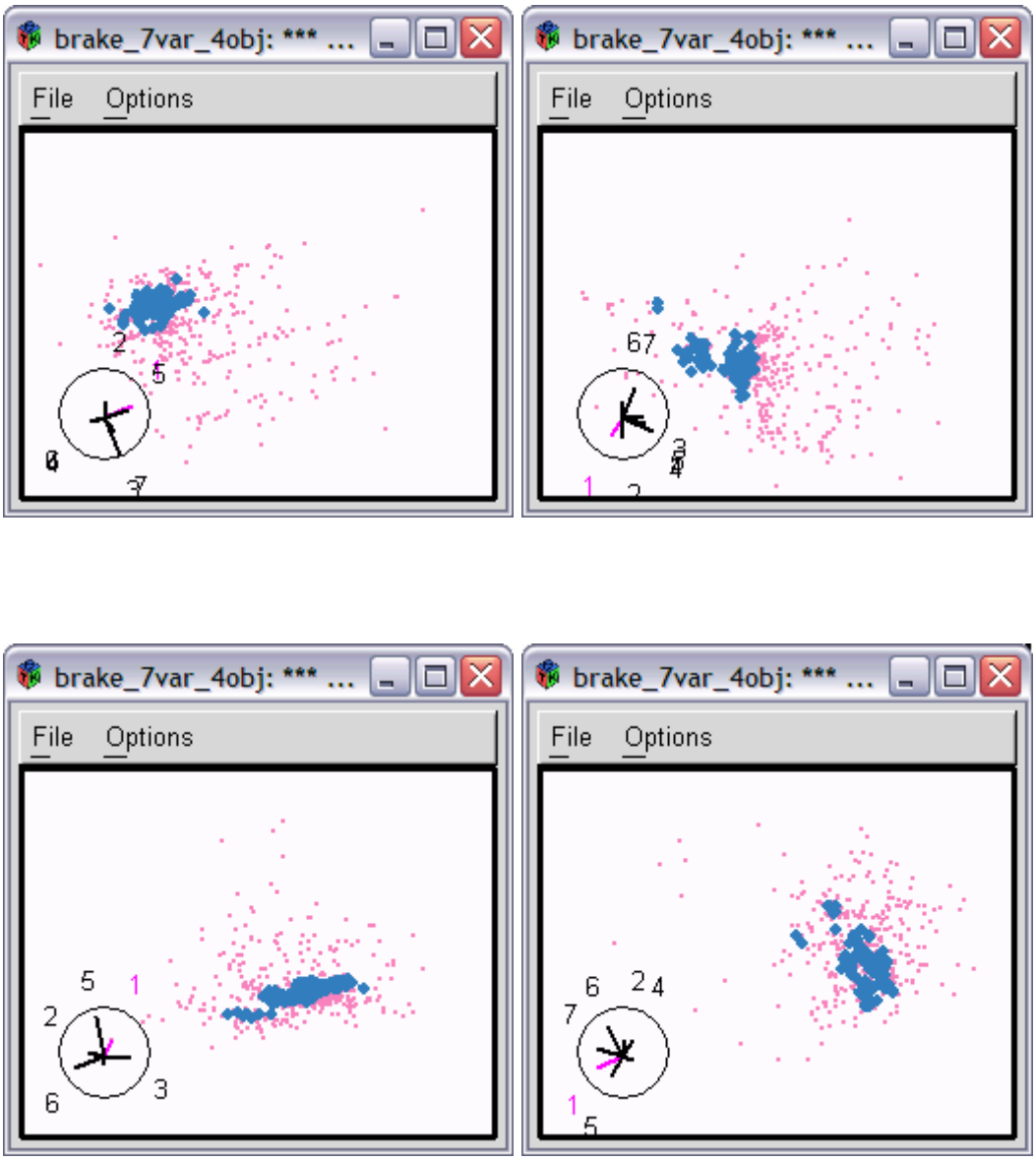
Appendix I Snapshots from Grand tour

Figures 14-17 are snapshots from grand tour of GA data. They are presented in four dimensions with Design Variable 1, 3, 5, and 7. The highlighted data points are with good fitness.



Figures 18-21 are snapshots from grand tour of GA data. They are presented in 7 dimensions with all design variables. The highlighted data points are with good fitness.

www.manaraa.com



www.manaraa.com

CHAPTER 7. DISCUSSIONS AND CONCLUSION

The thesis focuses on solving difficulties existing in multiobjective optimization problems for Engineering Design. First, engineering design problems are modeled as in complex dynamical models, which are hard to apply traditional optimization methods to. Second, lack of visualization and data analysis methods make it difficult to understand multivariate relationship in engineering multiobjective design. Third, robustness in engineering design is a fundamental problem. To balance robustness, optimality, and diversity is a multiobjective problem itself. Few existing multiobjective optimization algorithms are aimed at improving robust design for Engineering Design.

For the first issue, we propose applying genetic algorithms to Engineering Design Optimization. Genetic algorithms are very general optimization techniques that do not require calculating gradients and having explicit function description of the system. In particular, genetic algorithms work very well on different types of optimization problems (continuous and discrete). They are less susceptible to getting stuck' at local optima than gradient search methods because of its multi-directional search. Engineering design problems sometimes are so complex that no closed-form transfer function is able to represent the relationship between outputs and inputs. Design engineers normally build complex dynamical models in modeling software tool to study system dynamics and control characteristics. For example, the hydraulic problems presented in the thesis are typical problems that engineers are facing in current industry. They are high dimensional design problems and involve non-linear dynamics and uncertainty in disturbance. Without good optimization tools, currently engineers have to design the system based on their own experience. Existing commonly used optimization methods such as simulate annealing and Tabu search, etc, have difficulties to deal with multiobjective optimization problems. Applying GAs to these high dimensional optimization problems has shown very good results in the dissertation. This dissertation suggests using real-value GAs for engineering design problems because it is natural for designer to treat each design input as a gene. This also makes that crossover operation in GAs will not cut in the middle of a design input as in the binary format. Existing researches have shown that real-value GAs have the similar performance as binary ones for real value optimization problems.

There are many varieties of multiobjective Genetic Algorithms dealing with multiobjective optimization. This dissertation has evaluated different techniques in fitness

www.manaraa.com

design for multiobjective GAs. Combining multi-objectives into one fitness function is a common way to simplify optimization problems and easy for design engineers to understand. But this method can not explore the Pareto space and will only return one optimization solution as directed by the fitness function. It is suggested by this dissertation that this method should be only be applied to some simple problems or the problems that designers have preliminary knowledge about optimal solutions. Fuzzy objective functions and dominance based objective functions have been explored and have shown that they are capable to explore the Pareto Space more thoroughly. But fuzzy objective design requires that designers have some preliminary knowledge on the Pareto Front. Different fuzzy functions will result in different search scheme for Genetic Algorithms. Therefore, the dissertation recommends using dominance based multiobjective GAs to solve high dimensional and complex optimization problems in engineering design areas.

In multiobjective optimization, GAs require several design cycles: modeling, optimization, evaluation. Designers have to explore the design space to decide the final optimal solution. As Parmee suggested, GA has more impacts on preliminary study. It is important for GAs to quickly explore the design space and find the High Performance area. Designers can further explore HP area through clustering or visualization. With this idea in the mind, we proposed a new Completely Dominant Selection method. The idea is to relax dominance rule and allow non-dominant solutions to have chance going into the population. It still pushes searching converge to the Pareto Front or areas close to the Pareto Front, but prevents stuck into locally dominant solutions. Compared with common GA's selection method, it adds a tolerance to dominance concept, i.e., a solution is dominant to another solution only when the difference is larger than the tolerance. Since the new selection method does not change the essential dominance idea, many Genetic algorithms still can be coupled with this idea. We used a standard testing problem for Genetic Algorithm to show the powerfulness of this new idea. From the convergence and diversity aspects, the new idea shows a great improvement on performance comparing with Genetic algorithm without diversity control techniques. The new GA has better convergence performance than standard NPGA and achieves the same level diversity performance. Furthermore, the new GA doesn't need to use complex diversity control techniques which are computation intensive. Its natural characteristics results in diversity in the population.

www.manaraa.com

There are many research areas that can be explored in multiobjective Genetic Algorithms. All multiobjective Genetic algorithm optimization share the disadvantage of extensive searching. It is unavoidable that GAs will search a lot of unrealistic areas in the view of design engineers. How to guide GAs using expert knowledge is a potential research area to further improve efficiency of multiobjective genetic algorithm. Other future research areas include applying this new idea to more existing multiobjective Genetic Algorithms and solve multiobjective optimization problem with constraints.

We developed a software tool called EDGA for design engineers to use GAs for engineering design problems and applied EDGA on a hydraulic design problem from real industry. It shows that industry design problems are always involved with multiobjective optimization. The interaction between design variables and objectives is hard to understand. GA is a good way to explore the problem and find out the Pareto Front without profoundly knowing relationship between input and output. With the help of interactive visualization, we are able to present the high dimensional Pareto Front set, robustness of design, and sensitivity of design variables in high dimension using EDGA. The application has shown that our software and design concepts help engineers solve real industry problems.

For the second issue, we presented various visualization techniques for analyzing GA results. What makes GA data unique is its complex interaction between genes and fitness (Pareto dominance) and the existence of the Pareto Front. Existing visualization methods for GA data can only explicate evolving process and multivariable relationship in low dimension. We used a real industry design problem as an example to illustrate how to explore GA data using dynamic visualization. Dynamic visualization allows users interact with visualization. It can link several different plots together so that users select an area in one plot and the corresponding areas in other plots will be highlighted. Another feature of dynamic visualization is to project high dimensional data into 2 dimension plots and show them continuously by randomly changing project matrix. These techniques are very powerful to show design variable sensitivity, robustness of design, and the Pareto Front. For engineering design problems, another visualization linkage can be established by linking design combination with physical models. In John Deere Backhoe design project, backhoe model movements are displayed in a virtual reality environment. Designers can select any design combination saved in database interactively and look at Backhoe's movement. It is more impressive for designers to look at system

www.manaraa.com

performance before production than look at data. Even though building system graphical simulation needs additional work, it is worth the effort for designing a complex system.

Some of high dimensional visualization techniques are hard for designer engineers to understand, such as projection and grand tour. It will take some training time to get familiar with these techniques. Researches have proved that there is no significant improvement if the high dimensional data are projected to 3-D space. Engineers still like to see normal 2-D plots. In the future, more industry application cases are needed to present to show how to understand the high dimensional data using dynamic visualization and how to present GA data using high dimensional visualization techniques.

For the third issue, we discussed some important issues of multiobjective genetic algorithms thoroughly in the dissertation. Diversity and convergence are two conflicting factors in multiobjective optimization. How to balance them is an essential part for different GAs. For engineering design problems, designers are concerned about design robustness. Products operated in real environment are susceptible to disturbance and their manufacturing process is easy to bring tolerance as well. A practical good design should be satisfactory for optimality and be robust to uncertainty. We suggested using the completely dominant Genetic algorithm for balancing these two key issues. CDGA is capable to locate the high performance area without destroying the robustness information. After CDGA has found the high performance area, we applied clustering method to find out the most robust area in the high performance area. The simulation results have shown that CDGA has a great performance on convergence and diversity. In addition, it shows excellent ability to locate the robust area. CDGA shows significant performance on convergence and diversity even without any diversity control techniques. In addition, CDGA is intuitively designed for design robustness consideration. Designers are able to put their requirements for objectives optimal level and design variable tolerance into CDGA. CDGA is capable to handle these constraints to find the robust and optimal solution. Its simplicity makes it suitable for quickly setting up preliminary study of engineer design as well.

Different examples have been used to prove the concept of CDGA. Not only the convergence and diversity performances of CDGA have been studied, but also the robustness performance has been shown with the examples. The dissertation has described how to use CDGA to meet designers' robustness requirement and how to use clustering algorithm to analyze CDGA's result to find the robust solution. Simulation

www.manaraa.com

results have proved that CDGA is capable to locate the robust solution and determine the maximum disturbance that system allowed.

In the future, CDGA need to apply to more complex design optimization problems. Some future research areas include applying CDGA into multiobjective optimization problems with complex constraints, applying CDGA with diversity control in high performance area,

To conclude, this dissertation is dedicated to engineering design optimization using Genetic Algorithms. It discusses several important issues from GA software for dynamic modeling, Multiobjective Engineering Design Optimization to Data Analysis. It has been preliminarily used in industry for designing hydraulic system. Future additional exploration of software is needed and more complex industry problems are needed to be tested. The new complete GA has its exceptional advantage over classical multiobjective GAs and will be applicable to numerous engineering design problems, especially ones concerning about robustness.

www.manaraa.com

Appendix: ENGINEERING DESIGN SOFTWARE

1. Software Overview

Engineering Design Genetic Algorithm (EDGA) software is designed to assist design engineer to optimize system using Genetic Algorithm. Design engineers face varieties of different design tasks in current rapid prototype competition environment. All design tasks involve optimization in certain design stage. A user-friendly optimization tool is necessary to help design engineer quickly solve optimization problems.

It is no surprise that many commercial and non-commercial software packages have provided various optimization algorithms. For example, there are many different algorithms in Matlab© Optimization Toolbox, and there are many free Matlab based optimization software packages [4, 16]. However, design engineers need to implement their problems into software packages, which could be the hardest challenge.

Due to design complexity, design engineers always need to transfer their engineer design problems into computer language so that they can use computer to do complicate and tedious calculations and simulations. In design industry, many commercial software packages are available for modeling complicated engineering system. For example, EASY5© is a graphics based dynamics modeling tool and is developed by Boeing Company. It has been extensively used in hydraulic control modeling areas such as airplanes and tractors. Our software is designed to be able to integrate with various modeling software packages. The integration is independent with modeling software so that it can be generally applied to optimization problems without limitation.

Our software is intended to assist design engineers to solve optimization using Genetic algorithms. The goal is that the software is so easy to use that engineers can use it in daily basis. It has easy-to-use features and includes many user-friendly interfaces providing easy interactions. The complicated fitness design part has been simplified by using interactive graphics design.

Our software not only provides GA optimization tools, but also provides various methods to assist engineer to understand system. EDGA has Design by Experiment functionality and Visualization tools to help engineers do preliminary study and explore the system. These tools are extremely useful because GA can only optimize system to where it is designed to search. The preliminary study will help to study multiobjective relationship and relationship between inputs and outputs.

www.manaraa.com

Software structure is shown in figure 1. The whole structure is database core-based. Any components' communication is through database. This structure allows people work coordinately and share their design experience and results optionally. The core system is developed by using Perl language. The main reason to use Perl is that its excellent cross-platform transformability and good efficiency. Its object design ability and easy GUI development, of course, are its advantage over some other program languages. Many other tools are not developed in Perl. They are either because certain algorithms have already been well developed by other packages or many required libraries have been provided by other packages. Since these tools do not directly communicate with Perl program, it won't increase software-using complexity. It is good for extensibility because many design engineers are not expert on Perl but are familiar with Matlab and C and capable to develop their own functionalities.

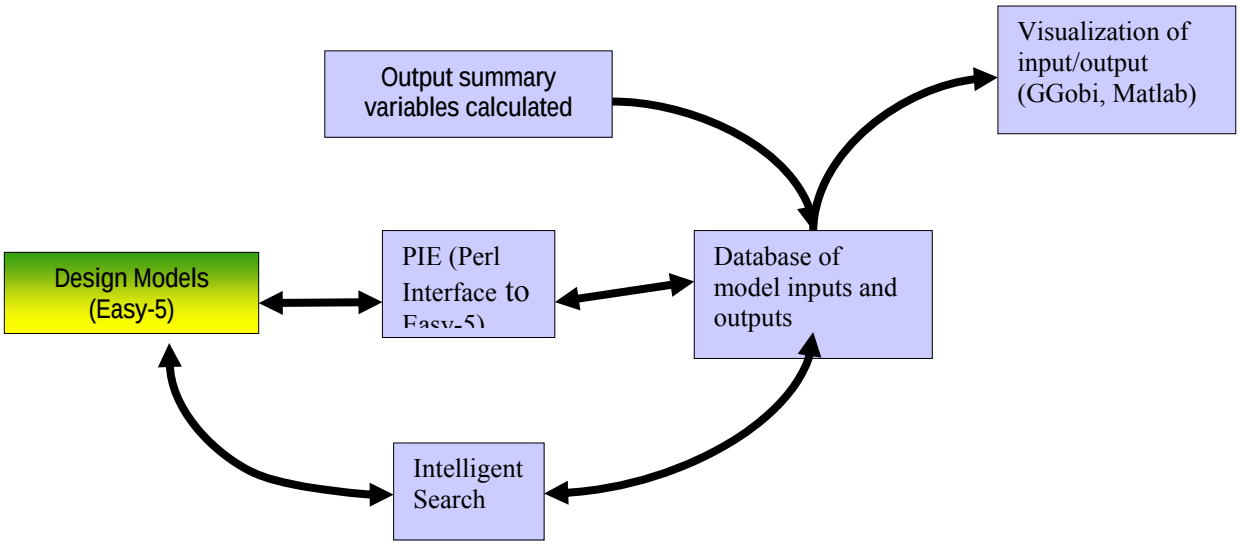


Figure 1 software infrastructure

The integration of modeling software requires an input file, an executable file, and an output file. The input file specifies parameters and tables users are interested in. Values of these variables can be changed through PIE interface. The output file contains the simulation results from the modeling software. Normally it is a time-series file. Users are able to select multiple outputs they are interested in and generate the output file in the

www.manaraa.com

modeling software. The executable file is the model file generated by modeling software. In this seamless design structure, any modeling software can be coupled with our software system.

Designing a high dimensional system requires running many simulations. The management for simulation data is difficult for engineers. Due to large number of runs, it is hard to know what combinations have been run and what combinations have not. Database is used to provide storage and management for large sets of simulation data. The standard structure of the database makes the whole software easily extensible.

The output file generated by modeling software contains time series output data. Designers are not normally interested in the whole time response, but are interested in some summary variables, such as maximum value and overshoot. Our software provides a data-filter tool to process time series data based on users' preference. The results will be stored into database as well.

Simulation results for a high dimensional complex system are very huge. Designers have difficulty in understanding multi-parameter relationship. Our software provides high advanced visualization technologies to help designers to understand the system. Except conventional 2D and 3D plots are provided, high dimensional visualization has been included in our software. Dynamic and interactive visualization and Grant Tour (36) provides a new way to look at multivariate data.

Intelligent search tools are to optimize the high dimensional system automatically. Designers only need to specify what to optimize. Then intelligent search tools use genetic algorithm (GA) to find the optimal solution in immense design space. Database and visualization are applicable to GA results as well.

2 Software Integration

The software is constructed and tested with EASY5 modeling software. However, its seamless integration method makes it extend to most of modeling software. The integration structure divides system modeling into three parts: input file, models, and output file. It requires modeling software should allow this structure. Fortunately, most of modeling software is compatible with the structure.

Input file defines a number of input variables. In any modeling software, there are a number of different variables with certain initial values. They are stored into a separated file (For example EASY5) or can be put into a file and load it before simulation (For

www.manaraa.com

example, SIMULINK). To make software compatible with any modeling software, there are only two choices to handle input file. Either a universal compatible input file format has to be created or each input file format has its own input file processing method. The first way doesn't exist in reality. Each modeling software has its own input file format, which makes it impossible to convert between each other. The universal format is not going to happen in modeling software industry in the near future. So the second way is the only possible way, which is exactly what EDGA employs. This way sounds complicated because it has to provide input file processing functions for almost each modeling software. Due to EDGA's modular design, it is actually very easy to implement different processing methods. EDGA has separated processing function into an independent library. For different modeling software, EDGA only needs to load the corresponding library. From users' usage point view, there is no actual difference so that EDGA is easy to extend to other modeling software packages.

Models contain actual design information. In dynamical modeling, complicated differential and integration equations with nonlinear functions are constructed to simulate input and output relationship in models. It has no surprise that each dynamical modeling software has its own design methods. However, this will not affect EDGA. EDGA only calls simulating models through modeling software. For example, if using SIMULINK, user can run SIMULINK models through MATLAB command *sim*. To run SIMULINK models from EDGA, a MATLAB engine dll file has to be set up to call the MATLAB command *sim*. Even though it seems that EDGA goes through a long way to run models, this structure makes it independent with modeling software. It only needs to build connections with command to run models. This setup is also good for compatibility.

To check if system meets requirement, system response has to been recorded. As dynamical modeling simulates system dynamics and use integration to solve differential equations, the output from modeling software is always time series response. Dynamical modeling software generally provides powerful capability to handle output files. Users can set up any format they want. Therefore, it is possible to set up a standard for output file format so that EDGA does not need to be modified for different modeling software packages. The general output format that EDGA accepts is the first line of output file is variable names separated by spaces. The first column is simulation time. The rest columns are output variables users are interested. Any output file processing in EDGA is based on this format.

www.manaraa.com

Figure 2 shows the main interface for EDGA. All the functionalities can be found in interface's menus. The interface is designed following general GUI characteristics. Perl does a good job on keep interface same on different platforms.

3 Software Description

3.1 Design by Experiments

Design engineers have to do many trials to test their design. For example, if they are design 5 parameters to optimize certain system, they normally choose three values for each variable: minimum, median, and maximum. In this approach, they have a basic feeling how each variable affect system so that they can decide if design is worth to do further investigation. Five variables will make total simulation number to be 243. Without a good tool to manage running such many runs, no design engineer is willing to run them manually. First of all, it will take so much time. People have to wait with sitting before computer till it finishes one run, then sets another combination and runs it again. It is going to be a nightmare for such large runs. Secondly, it is impossible to remember what combinations have been run and what ones have not. People may have to write down combination information into a file themselves and search for combinations have not been run in the file. It is not going to be easy in this way. Third, it is hard to match output files with input combinations. If users want to see a particular run, they have to find the filename in record and then plot them. It is not easy to switch different output displays.

Our system is capable to provide easy management and control for multiple runs. Users can select any number of design variables and set any number of values for each variable. Then all the combinations will be stored into database and simulation will be handled by software. Users do not need to interact with system at all. EDGA will automatically manage running all the runs and store simulation runs into database.

After users select design variables defined in the input file, the main interface changes as figure 3. Users can define number of values and value for each version. EDGA provides an interpolation function to fill with large number of values easily. In addition, it allows several design variables are linked together. The linkage between multiple variables means that these variables have certain relationship. For example, if a backhoe is loaded, not only its mass is changed, but also system inertia is changed. It makes sense to change these two variables together in designing backhoe.

www.manaraa.com

In designing complex system, it is very common to modeling some system characteristics in 2D table interpolation. 2D table represents a nonlinear relationship between an input and output, which is hard to be modeled in math. It often comes from experiments. For example, in hydraulic system, a control valve has its characteristic pressure vs. flow property. This curve shape in the valve is what design engineers want to design. In modeling, the 2D table is represented in a 2D array. If engineers want to redesign the array, they have to manually type in values for 2D array in modeling software traditionally. EDGA provides graphically interactive method to design curves as figure 4. It is easy for users to rescale or reshape curves. Users can use interpolation to generate a set of curves of which boundaries are defined.

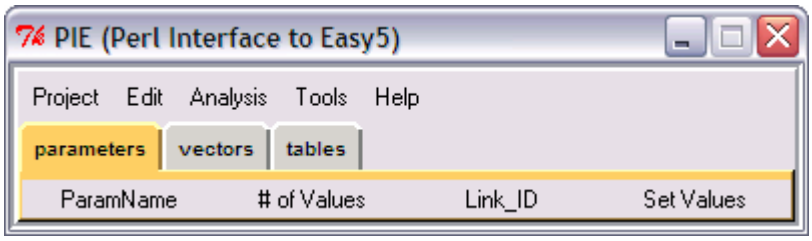


Figure 2 EDGA main interface

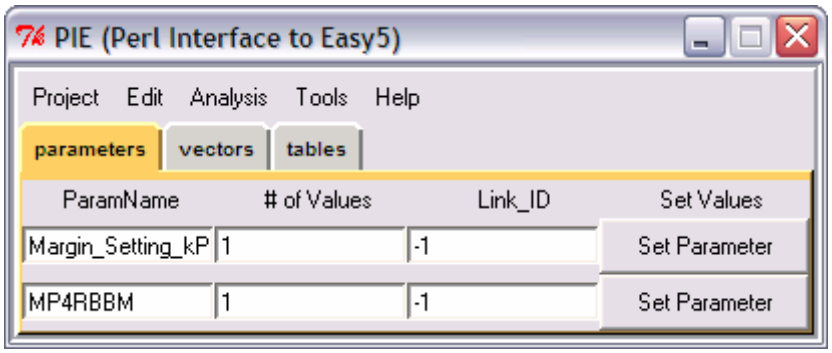


Figure 3 EDGA interface after adding parameters

www.manaraa.com

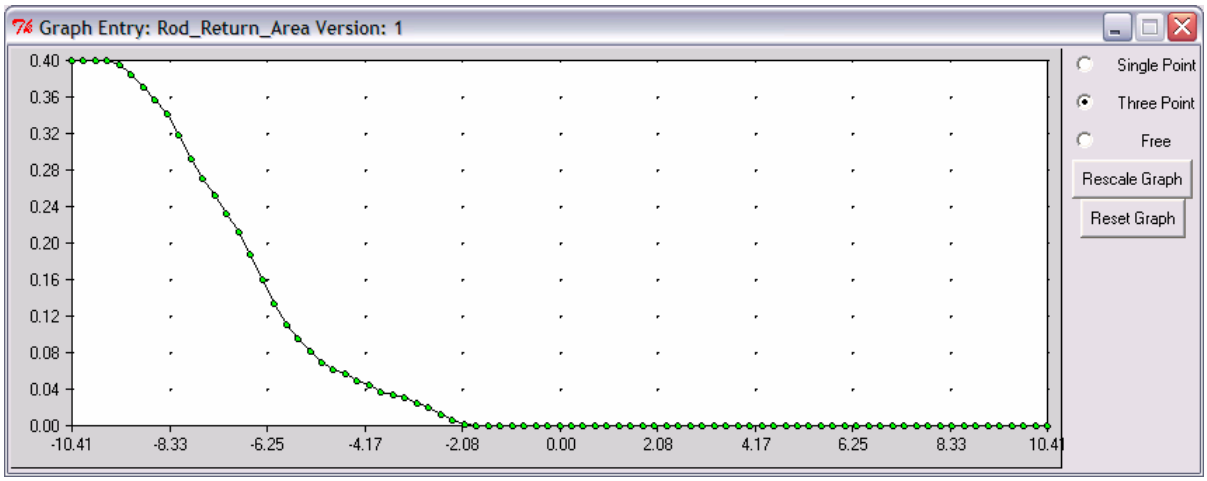


Figure 4 Table Graphical Design Interface

As users set up all parameters and tables, EDGA will take over control and manage to run all the combinations. EDGA has a multi-thread software structure so that it can handle different tasks' communication and show running status of simulation. It will tell users how many runs are left and what is the estimated running time required. It is so convenient that users can make running experiments fit their schedule and not affect their other works. Because database is used to store combinations, EDGA has ability to check input combinations easily so that ones already in the database do not need to run again. If Design By Experiment matrix is too big, the simulation of EDGA can be stopped any time and resume again.

All the simulation data will be stored in a local directory users specified and can be put into database as well. The simulation output files are marked by a unique number, which corresponds to input combinations with same id in the database. This setup provides a linkage between input and output files, which is very important for further development. It makes possible to present data visually without searching linkage information.

3.2 Output Analysis

After running experiments, a lot of simulation runs recording system outputs are stored. The actual goal of running tedious experiments is to look through data and find interesting results. Each simulation file contains time series data for several variables. It will take a long time to visually examine each time series data set. It is almost impossible to clearly present graphs showing all the runs for large runs. EDGA provides output file

www.manaraa.com

processing techniques to capture unique characteristics from time series data so that design engineers can do prescreen quickly. It often helps them to better understand relationship between input and output.

A set of time series data is a data sequence. Designers are not interested the whole sequence but some unique characteristics. These characteristics are used to evaluate system design and compare with other designs. For example, in designing a controller, to decide a controller whether good or bad can be evaluated by steady state tracking error and system response rising time. Clearly designers want to get zero tracking error and the quickest response. These two characteristics of course can be pulled from looking at time series data. However, it would be nicer to generate exact results by processing output data.

Figure 5 shows a typical step response from an actual system. Some features that might interest design engineers are overshoot, peak value, and settling value. EDGA provides a derived data function, which is processing output files to generate derived values. Currently EDGA derived function supports Peak Value, Peak Time, Overshoot, Decay Ratio, Settling Value, Settling Time, Minimum, and Maximum. Users can select any derived values for each output variable. EDGA will process all the simulation files and store derived values into database. The derived values are stored in the same order of file id order so that they are easy to be connected with input combinations. The data processing function can be easily extensed.

3.3 Visualizations

Design by experiments will generate a large number of data. Design engineers often have trouble in exploring the data. Data needed to be studied contain input variables, which formulate input combinations, output variables derived from time series data, and time series data files (figure 5). It forms a network of data. The most effective way to quickly explore data is using visualization. Graphics contains the deepest and most comprehensive information. Most of the important, it is easy for engineer to understand and share information. Designers want to be able to walk among these pieces of data freely. For example, they need to find out system response for a special combination or need to find the combination with the quickest response. Sometimes, designers want to see a set of combinations' response to compare their results. Therefore, software should

www.manaraa.com

be designed to let users easily switch visualization views for any variables and any.simulation data.

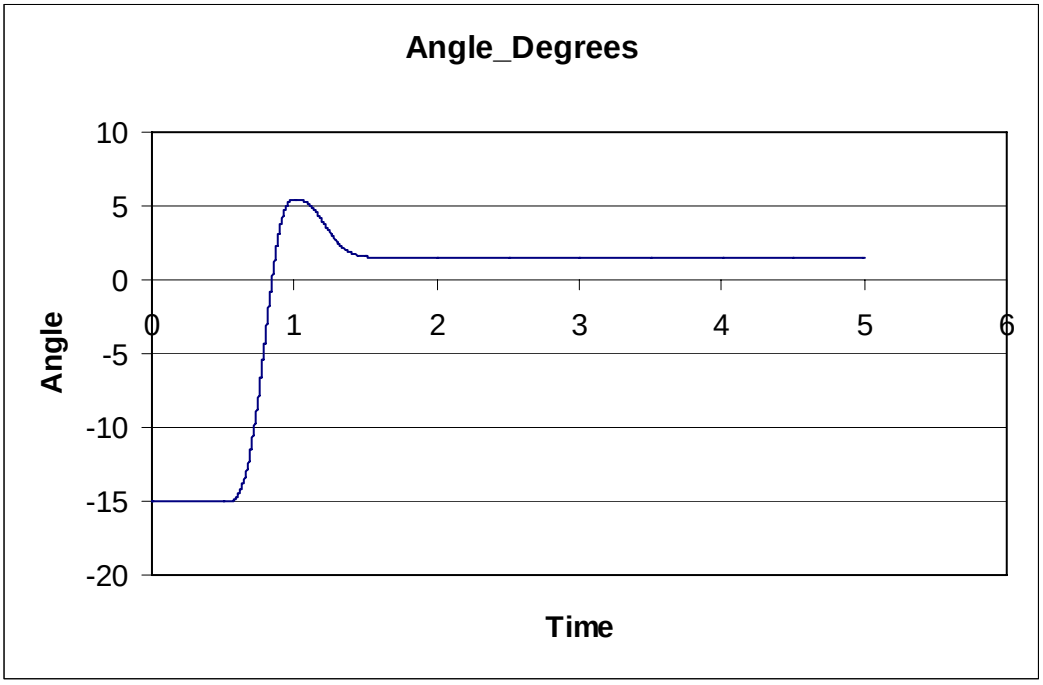


Figure 5 Typical time series response

Database makes it possible to easily pull data from any point in the net. All the data from the same simulation run have the same id in the database so that they are linked together. Database has a powerful searching and sorting algorithm so that visualization software can interact with database quickly. In addition, people can share the data through the database server. Because of its standard format, people have no problem to use them and are able to use any visualization software based on MYSQL database.

EDGA doesn't provide actual visualization functions. Its main function is to put data into database and manage the database. Because of database's popularity, a lot of software has functionalities to communicate with database. For example, Matlab is popular software using in engineer community. Engineers are familiar with Matlab environment and be able to develop programs using it. It has ability to communicate with database through c mex functions, and even the latest version has its own library for database communication. It is so easy to develop some programs to plot certain graphs. Figure 6 is one matlab GUI provided auxiliary by EDGA. It is able to plot time series data from any number of files and put them together. EDGA has many other matlab GUIs such

www.manaraa.com

as 3D plot and sensitivity analysis plot. All these plots are for special purpose. Users can develop their own special purpose analysis tool using simulation data or data from database.

EDGA has a special connection with high dimensional visualization software GGobi. GGobi is evolved from XGobi and is a high dimensional statistical visualization software package. Besides that it is able to provide common plots such as scatter plots, time series plots, and parallel plots, most of unique features are its interactive and dynamic visualization. Each plot in the GGobi can be linked to other plots. It has a special selection method called brushing, which is that the corresponding points selected in one plot are highlighted in other plots. Through dynamic brushing, it is easy to see connections between several plots. For example, if one plot is set as a parallel plot to show input combinations, and the other plot is set as a scatter plot to show two output variables, brushing can easily tell that which combinations have the best response for these two outputs and what response of each combination has. Another of nice features is that GGobi has a projection method to show high dimensional data in a 2D plot. It is using a continuous random projection animation to show high dimensional data. Experienced users can detect high dimensional relationship between each dimension and clusters of data in high dimension.

www.manaraa.com

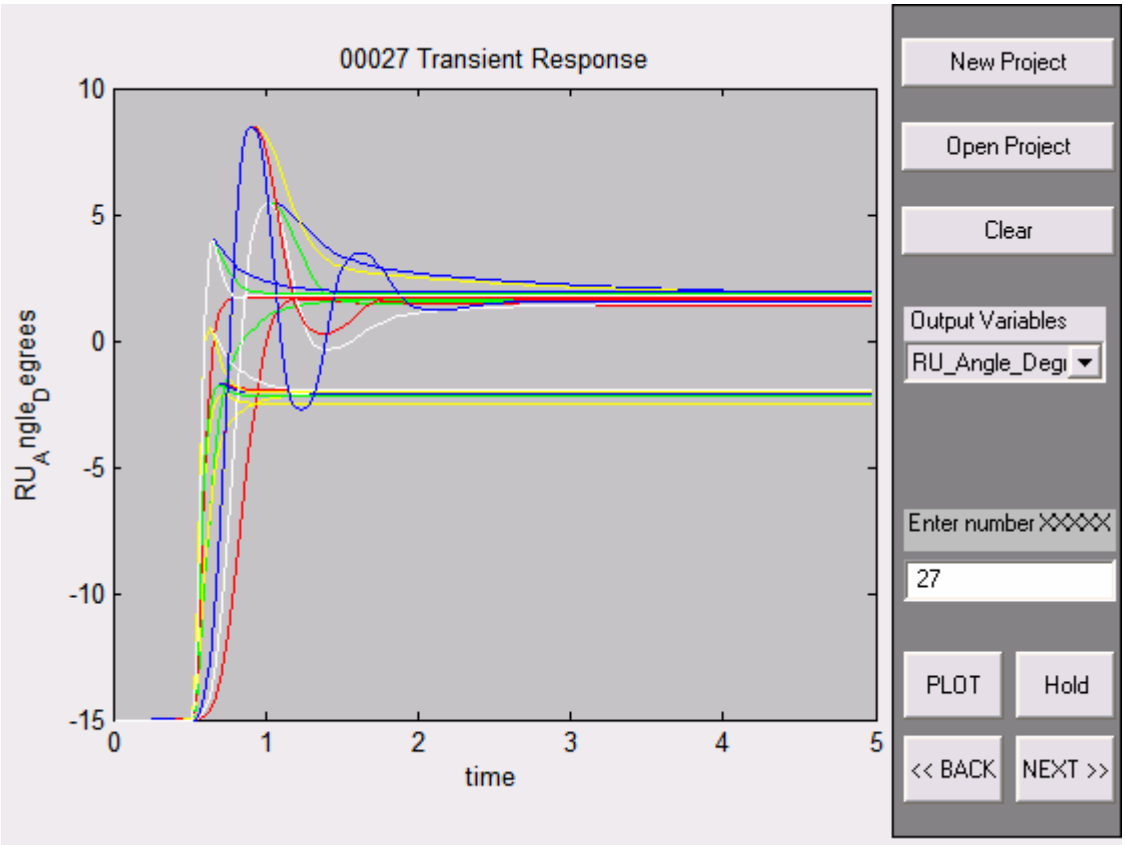


Figure 6 Matlab 2D plot interface

Even though GGobi supports many data formats, database communication has not yet been fully supported. EDGA provides an GUI interface to load data from database and pass data into GGobi. It is called Data Visualization User Interface (DVUI). It is not a simple bridge between database and GGobi, but also has many data handling methods. Database stores tremendous sets of data. They are all useful for each analysis task. Designers may only be interested in a subset of data. It is also important to reduce data size for better visualization purpose. Figure 7 shows interface of DVUI. Users can select parts of variables from long list of variables in the data and set boundary for each variable. The data control strategy is very useful to visualize large data sets such as automobile warranty data and weather records.

www.manaraa.com

4. Evolutionary Design Interface

Evolutionary Design Interfaces are designed to assist engineers to optimize system performance by using Genetic Algorithm. Interfaces include Gene design interface, GA operator design interface, and Objective design interface (figures 7-9).

In Gene design interface, designers can select parameters and tables from the input file. The boundaries and resolutions can be set for each parameter or table. The interface provides ways to distinguish design variables with condition variables. Condition variables are the ones that the system is required to run on different values. For example, if a system is required to run on three different engine speeds to check performance, the engine speed variable is considered as a conditional variable. Design variables forms actual genes in GA.

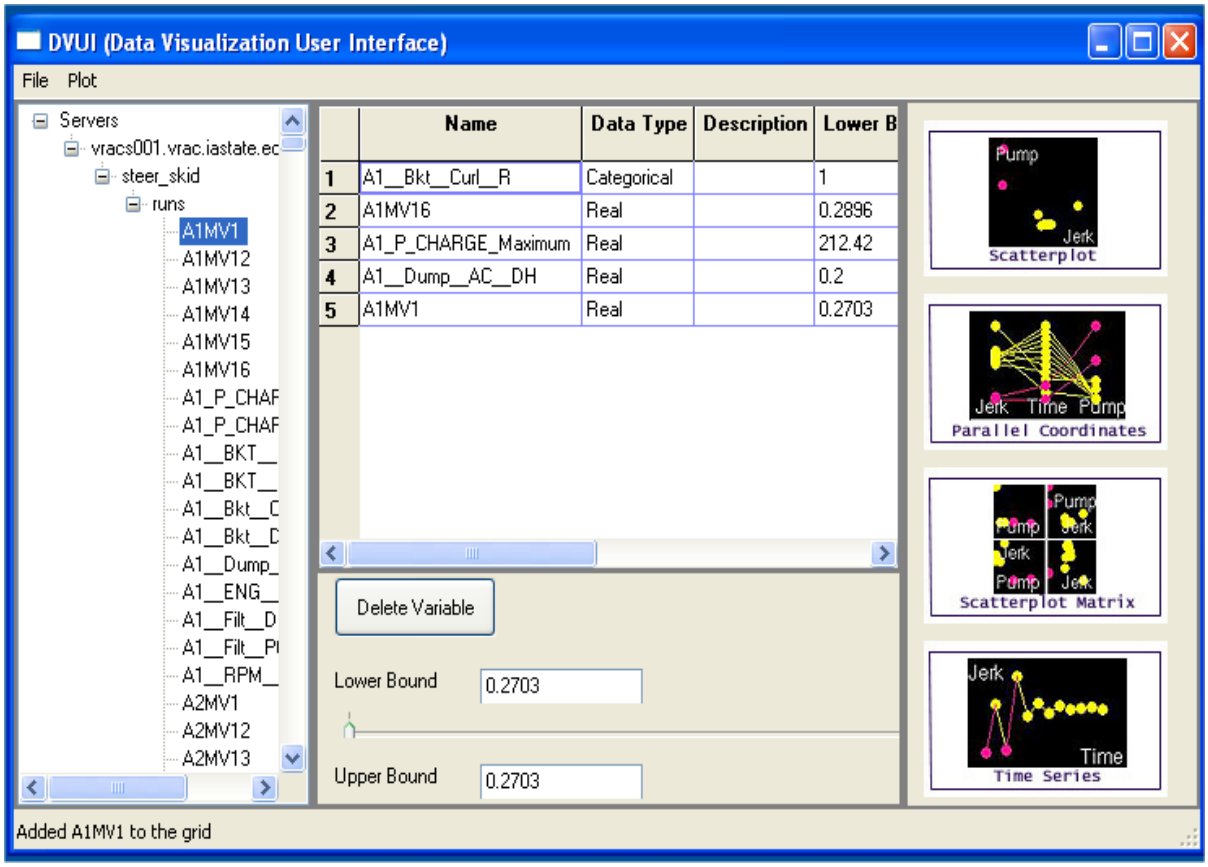


Figure 7 Data Visualization User Interface

www.manaraa.com

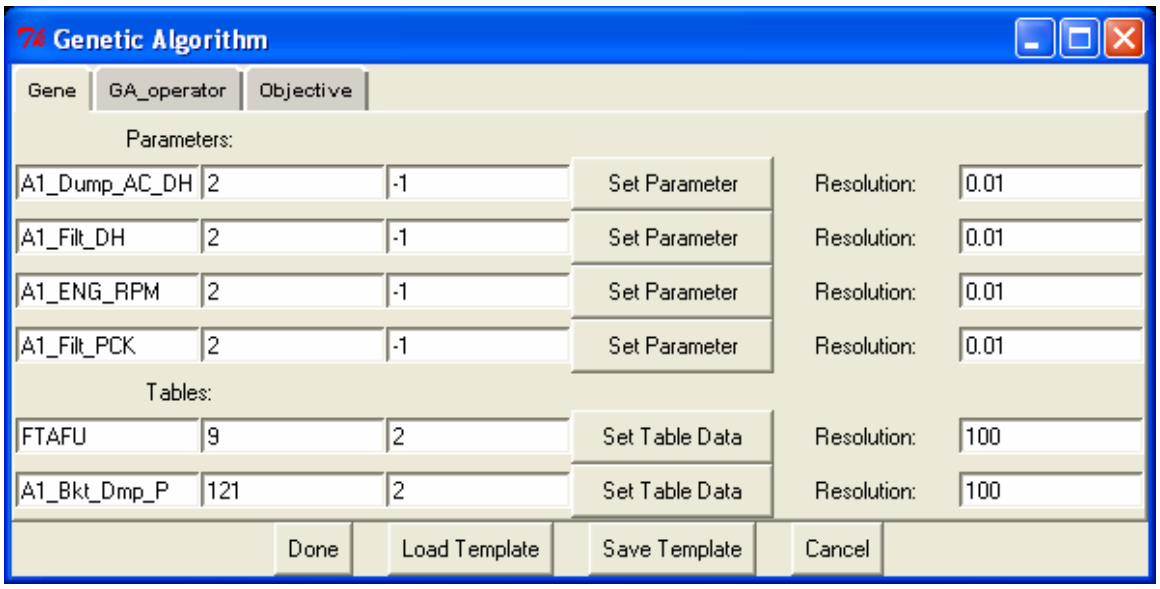


Figure 8 Genetic algorithm Gene Design Interface

.

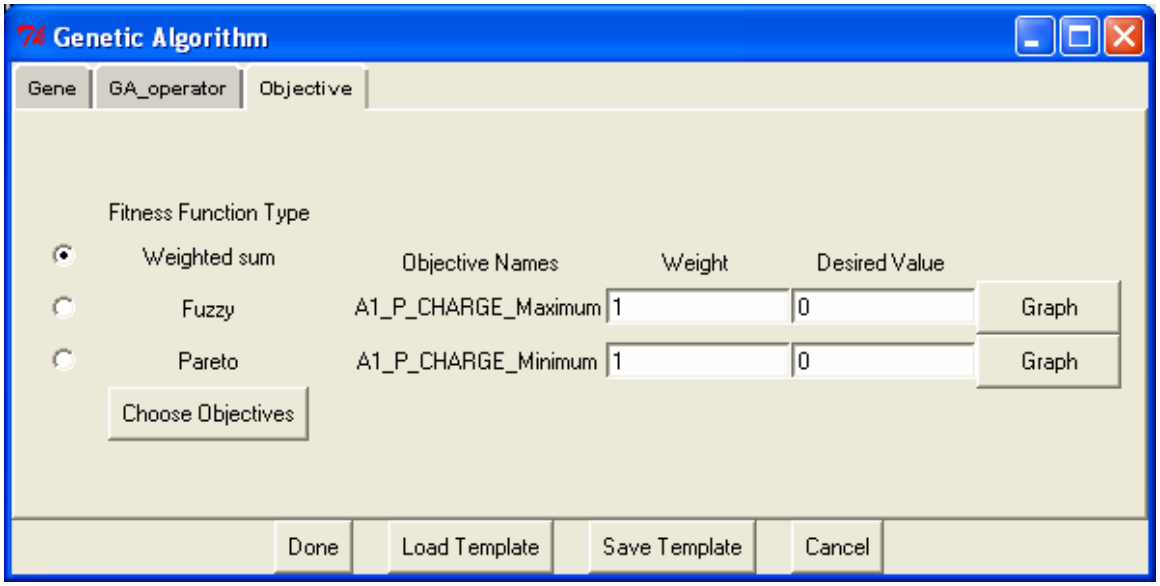


Figure 9 Genetic Algorithm Fitness Design Interface

GA operator interface is used to set GA's characteristics, such as initial population, mutation and crossover methods, and selection methods. The interface is designed as simple as possible to meet engineers' requirement. A lot of detailed algorithm characteristics have been predefined.

Objective design interface is a key element. Designers use this interface to specify what to optimize. As real system always has multiple objectives to optimize, the interface is designed intuitively for multiobjective system. Three different multiobjective

www.manaraa.com

optimization methods: Weighted sum, Fuzzy fitness, and Pareto front optimization, are provided in the interface. Designers are able to select summary variables from a list of output variables and select what type of fitness function to use. Software allows users to see the fitness design for each objective graphically. GA will find the optimal solution based on users' fitness setting.

All mentioned visualization techniques can be used in analyzing GA results because GA data are stored into database as well. Except that, EDGA provides additional visualization tool to show convergence of GA.

4.1. Genetic Algorithm

One of the problems met in applying GA into engineering design is how to handle tables. A table is defining a nonlinear relationship between two variables. In practice, it is using an x-y 2D table to represent the relationship. Any value not in the table can be found by interpolating two neighboring x values. It is clear that one design parameter is represented by one chromosome. There are three basic ways to represent tables. The simplest way uses the interpolation between lower bound and upper bound tables to represent a table. The advantage of this method is that only one chromosome is used to represent one table. The disadvantage is that the shape of designing table is predefined. Another easy way is using the whole table points as genes. For example, table in figure 11 is defined by 10 x-y points. Ten chromosomes will be used to represent y values of these ten points. In this way, designers have much more freedom to design a curve, but the dimension increases dramatically.

The trade-off solution is to use a parameterized curve. A curve can be separated into several parts and each part is represented by a parameterized curve. For example, if using spine curve, the whole curve can be parameterized into four points. If the starting point of the curve is fixed, then the whole curve can be represented by three points. For more complex curves, several parameterized curves can be used. The disadvantage of parameterized curve is that mutation and crossover operators can easily generate curves not meeting curve constraints such as table boundaries and monotonous. Rejection techniques have to be used in Genetic algorithm to meet constraints.

www.manaraa.com

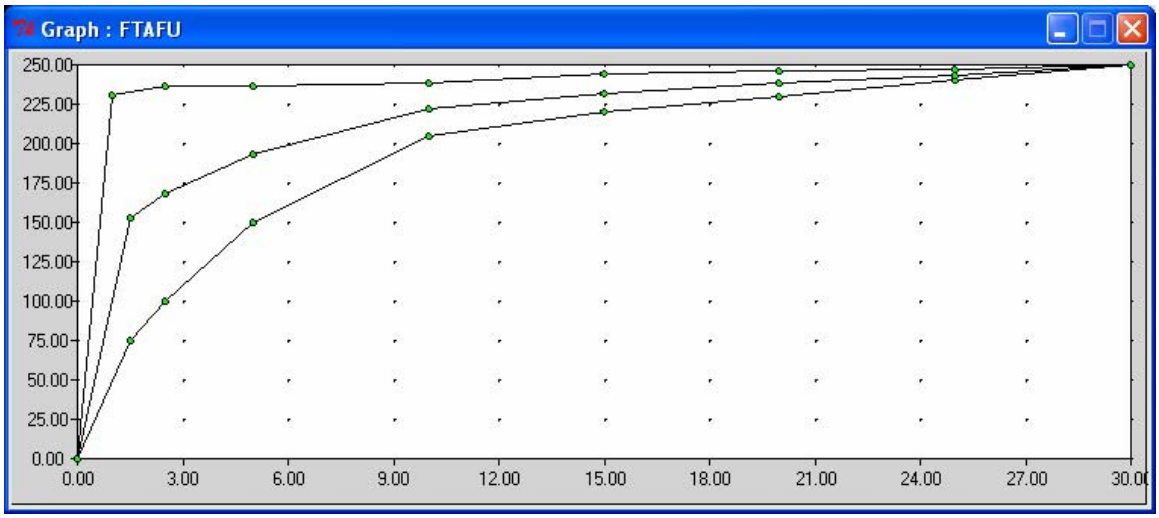


Figure 11 Design tables

Current GA interface only supports the first way to deal with curves. The parameterized curve method requires that users should know how to program parameterize curves and implement constraints. It is hard to implement in a general form. Additional table parameterizing function can be established to support this feature.

5. **Conclusion**

Current industry is lack of a software tool to apply genetic algorithms to dynamical modeling problems. We developed a complete set of software tool, called EDGA software for Engineering Design using dynamical modeling software. EDGA's generic design enables it to be easy to couple with any models developed in dynamical modeling software. EDGA not only provides functionality to optimize system using GA, but also provides Design by Experiments and visualization features to further help explore the system. Designers are able to use Design by Experiments to do preliminary study of complex system and use visualization to explore simulation results. Then designers can use GA to optimize system performance on the basis of preliminary study results. GA's results can also be analyzed using visualization. The relationship between input and output and the Pareto Front shown in visualization will help designers to pick an optimal solution from the optimal solution set located in the Pareto Front. This process can be repeated until designers make the final decision. EDGA is the first Design Automation Software for dynamical modeling software. The advantages of EDGA can be summarized as follows:

www.manaraa.com

1. Software generic structure allows easy coupling with any models;
2. Easy-to-use interface enables users to design easily, especially for designing tables;
3. Multiobjective Genetic algorithms allows different fitness design;
4. Database makes data storage and management easily.
5. Visualization extends software optimization capability.

One of the possible future developments for EDGA is implementation of interactive Genetic algorithms. Interactive Genetic Algorithm [69] has caught recent attentions because real system optimization is very complex and it is impossible to state how to optimize system using GA without studying the system extensively. The idea is to separate a complex problem into several subsystems and optimize them separately. Designer can interact with design process and connect these subsystems to get a global picture of the original picture. Another possible extension for EDGA is implementation of various Genetic Algorithms. It has been found that different GAs may have huge performance difference on diverse problems. It would be nicer to have more GA options for designers.

www.manaraa.com

REFERENSES

1. Abraham, J.A.R., Parmee, I. C., “User-centric Evolutionary Design Systems – the Visualization of Emerging Multi-Objective Design Information”, In Proceeding of Xth International Conference on Computing in Civil and Building Engineering Weimar, June 02-04, 2004
2. Acharjee, S. and Zabaras, N., "A gradient optimization method for efficient design of three-dimensional deformation processes", in the proceedings of NUMIFORM, the 8th International Conference on Numerical Methods in Industrial Forming Processes (edt. S. Ghosh), Columbus, Ohio, June 13-17, 2004
3. Aizawa, A. N. and Wah, B. W., “Dynamic control of genetic algorithms in a noisy environment,” in *Proc. Conf. Genetic Algorithms*, pp. 48–55, 1993
4. ANDERSSON J. AND KRUS P., “Metamodel Representations for Robustness Assessment in Multiobjective Optimization”, accepted publication in *Proceedings of the 13th International Conference on Engineering Design, ICED 01*, Glasgow, UK August 21-23, 2001.
5. Andersson, J, “Multiobjective Optimization in Engineering Design”, Doctoral Dissertation, Linkongping University, Sweden, 2001
6. Arnold, D., “Evolution strategies in noisy environments—A survey of existing work,” in *Theoretical Aspects of Evolutionary Computing*, L. Kallel, B. Naudts, and A. Rogers, Eds. Heidelberg, Germany: Springer Verlag, pp. 239–249, 2001.
7. Ashlock, D., “Evolutionary Computation for Modeling and Optimization”, 2004
8. Bäck, T., and U. Hammel, “Evolution strategies applied to perturbed objective functions,” in IEEE World Congress on Computational Intelligence, vol, 1, pp.40-45, IEEE, 1994.
9. Basalaj, W, “Proximity Visualization of Abstract Data,” Doctoral Dissertation, University of Cambridge, UK, 2001
10. Bauer, J. R. “Genetic Algorithms and Investment Strategies,” John Wiley & Son, Inc. 1994
11. Beaty, S., J., "Genetic Algorithms Versus Tabu Search for Instruction Scheduling," Artificial Neural Nets and Genetic Algorithms, 1993
12. Berenji, H.R. and P. Khedkar, “Learning and Tuning Fuzzy Logic Controllers Through Reinforcement,” IEEE Transactions on Neural Network, 1992.

www.manaraa.com

13. Bernard, J., J. Gruening and K. Hoffmeister, "Evaluation of Vehicle/Driver Performance Using Genetic Algorithms," SAE 980227, 1998
14. Bonham, C. R., Parmee, I.C., "Improving the Performance of Cluster-oriented Genetic Algorithms (COGAs)," In Proceedings of IEEE Congress on Evolutionary Computation, Washington D.C., pp 554 - 561,1999
15. Bonham, C. R., Parmee, I.C., "An Investigation of Exploration and Exploitation in Cluster-oriented Genetic Algorithms,"In W. Banzhaf and et al., editors, GECCO--99: Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, USA; pp 1491 - 1497; 1999
16. Bosman, P. and Thierens, D., "The Balance Between Proximity and Diversity in Multiobjective Evolutionary Algorithm," IEEE Transactions on Evolutionary Computation, VOL 7, NO.2, P174-188, April 2003.
17. Bosman, P. and Thierens, D., "Multi-objective optimization with diversity preserving mixture-based iterated density estimation evolutionary algorithms," *Int. J. Approx. Reasoning*, vol. 31, pp. 259–289, 2002.
18. Branke, J., "Efficient evolutionary algorithms for searching robust solutions," ACDM, Page 275 – 286, 2000.
19. Brake, J. and Schmidt C., "Faster Convergence by means of fitness estimation," In Soft Computing, 2000.
20. Carlos Manuel Mira da Fonseca, "Multiobjective Genetic Algorithms with Application to Control Engineering problems," Doctor Dissertation, The University of Sheffield, 1995
21. Cheng, R. and M. Gen., "A survey of genetic multiobjective optimizations" Technical report, Ashikaga Institute of Technology, 1998.
22. Cobb, H.G. and Grefenstette, J.F. " Genetic Algorithms for Tracking Changing Environments," Proceeding of 5th International Conference on Genetic Algorithms, P523-530, 1993
23. COELLO C., *An empirical study of evolutionary techniques for multiobjective optimization in engineering design*, Dissertation, Department of Computer Science, Tulane University, 1996.
24. Coello, C. A., "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques," Knowledge and Information System, An International Journal, l(3), 269-308, 1998

www.manaraa.com

25. Coello, C. A., "Evolutionary Multi-Objective Optimization: A Historical View of the Field," IEEE Computational Intelligence, pp. 28 – 36 , 2006
26. Curotto, E. "Applications of the Structural Comparison Algorithm for Simulated Annealing Optimizations and Stochastic Simulations," Technical Proceedings of the 2001 International Conference on Computational Nanoscience and Nanotechnology, P1-P4, 2001
27. Cvetkovic, D. and Parmee, I. "GENETIC ALGORITHMS BASED SYSTEMS FOR CONCEPTUAL ENGINEERING DESIGN," Proceeding of the Congress on Evolutionary Computation, P29-36, 1999
28. Cvetkovic, D. and Parmee, I. "Evolutionary Design and Multi–objective Optimisation," Proceedings of the sixth European Congress of Intelligent Techniques on Soft Computing", P397-401, 1998
29. De Jong, K.A, "An Analysis of the Behavior of a Class of Genetic Adaptive Systems." Ph.D. thesis, University of Michigan, Ann Arbor, MI, (1975)
30. Deb, K. and Goel, T., "Controlled Elitist Non-dominated Sorting Genetic Algorithms for Better Convergence", First International Conference on Evolutionary Multi-Criterion Optimisation, 67-81, Zurich, 2001.
31. Deb, K., Pratap, A., et. al. "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 6, NO. 2, APRIL 2002.
32. Deb, K., "Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems," *Evolutionary Computation*, vol. 7, pp. 205-230, 1999
33. Deb, K. and Gupta, H., "Introducing robustness in multiobjective optimization," Kanpur Genetic Algorithms Lab. (KanGAL), Indian Inst. Technol., Kanpur, India, Tech. Rep. 2 004 016, 2004.
34. K. Deb and T. Goel, "Controlled elitist nondominated sorting genetic algorithms for better convergence," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization—EMO 2001*, E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello, and D. Corne, Eds. Berlin, Germany: Springer-Verlag, pp. 67–81, 2001
35. Dennis Jr., J.E. and More, J.J., Quasi-Newton Methods, Motivation and Theory, SIAM Review 19, 46-89, 1977.

www.manaraa.com

36. Dick, G., "An Explicit Spatial Model for Niching in Genetic Algorithms," The 15th Annual Colloquium of the Spatial Information Research Centre, 2003
37. "Expanding Users for Dynamical Modeling," http://www.fwc.com/publications/heat/heat_pdf/9601-020.pdf, 1996
38. Fang, X. and B. Kellogg, et al., "High Dimensional System Design Using Genetic Algorithms & Visualization," American Control Conference, 2003.
39. Fleming, Peter and R.C. Purshouse. "Evolutionary algorithms in control systems engineering: a survey." *Control Engineering Practice*, vol.10, p.1223-1241, 2002
40. Fonseca, C. and Fleming, P., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalisation," Fifth International Conference on Genetic Algorithms, 416-423, California, 1993.
41. Fonseca, C.M. and Fleming, P.J., "An Overview of Evolutionary Algorithms in Multiobjective Optimization," Evolutionary Computation, 3(1): 1-16, 1995
42. Forouraghi B., "A Genetic Algorithm for Multiobjective Robust Design", Applied Intelligence, vol. 12, no. 3, pp. 151-161, 6 May 2000
43. Gaspar-Cunha, A. "RPSGAe – Reduced Pareto Set Genetic Algorithm: A Multiobjective Genetic Algorithm with Elitism," The Second Workshop on Multiobjective Problem Solving from Nature, 2002.
44. Gen, M. and R. Cheng, "Genetic Algorithms & Engineering Optimization," Wiley, New York, 2000.
45. Glover. F., "Tabu Search - Part I," *ORSA Journal on Computing*, vol. 1, pp. 190-206, 1989.
46. Goldberg, D.E. and Richardson, J.E., Genetic Algorithms with Sharing for Multimodal Function Optimization,*Proceedings of the Second International Conference on Genetic Algorithms,* 41-49, 1987.
47. Goldberg, D. and J. Richardson, "Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, Reading, MA, 1989.
48. Hart, E. and Ross, P., "GAVEL – A new Tool for Genetic Algorithm Visualization," IEEE Transactions on Evolutionary Computation, VOL 5, NO.4, P335-348, August 2001.
49. Hayashi, S. "NONLINEAR PHENOMENA IN HYDRAULIC SYSTEMS" http://www.fluid.power.net/techbriefs/hanghzau/1_3.pdf, 1999

www.manaraa.com

50. Herreros A, Baeyens E, Peran JR., "Design of PID-type controllers using multiobjective genetic algorithms," ISA Trans., 41(4):457-72, Oct. 2002.
51. Holland, J., "Adaptation in Natural and Artificial Systems," University of Michigan Press, Ann Arbor, MI, 1975; MIT Press, Cambridge, MA, 1992.
52. HESTENES, M., AND STIEFEL, E. Methods of conjugate gradients for solving linear systems. *Nat. Bur. Stand. J. Res. 49*, 409–436, 1952
53. Horn J. and Nafpliotis N., Multiobjective optimization Using Niched Genetic algorithm, Technical Report IlliGA1 93005, University of Illinois, Urbana Champaign, Urbana, Illinois, 1993
54. Horn, J., Nafpliotis, N., and Goldberg, D. E., "A niched pareto genetic algorithm for multiobjective optimization," in *Proceedings of the FirstIEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, J. J. Grefenstette, Ed. Piscataway, NJ: IEEE Press, pp. 82–87, 1994
55. Houck, C. Joines, J, and Kay, M. "A Genetic Algorithm for Function Optimization: A Matlab Implementation," NCSU-IE TR, 1995
56. Hughes, E.J. " Evolutionary Multi-objective Ranking with Uncertainty and Noise," EMO 2001, P329-343, 2001
57. HUGES E., "Evolutionary Multi-objective Ranking with Uncertainty and Noise", in *Proceeding of the 1st International Conference on Multi Criteria EvolutionaryOptimization*, Zurich, Switzerland, March 7-9, E. Zitzler et al. (eds.), Lecture Notes in Computer Science No. 1993, Springer Verlag, Berlin, 2001.
58. Jin, Y, and Sendhoff, B., "Tradeoff Between performance and robustness: An evolutionary multiobjective approach," In EMO 2003, P237-251, 2003.
59. Jin, Y. and Branke, J., "Evolutionary Optimization in Uncertain Environments – A Survey," IEEE Transactions on Evolutionary Computation, VOL 9, NO.3, June 2005, P303-317
60. Karaboga, D., "A Simple and Global Optimization Algorithm for Engineering Problems: Diferential Evolution Algorithm," Turk. Journal Elec Engin, 53-60. P1080-1091 , 2004
61. Katangur, A.K., Pan, Y, and Fraser, M.D., "Simulated annealing routing and wavelength lower bound estimation on wavelength-divisionmultiplexing optical multistage networks," Optimal Engineering, Vol 43 No. 5, May 2005

www.manaraa.com

62. Kokolo, I., Hajime, K., and Shigenobu, K., "Failure of pareto based MOEAs: Does non-dominated really mean near to optimal?," In Proceedings of the Congress on Evolutionary Computation 2001 (CEC 2001), vol. 2, pp. 957 – 962, Piscataway, New Jersey, IEEE Service Center, May 2001
63. Kirkpartrick, S., Gelatt, C. D., and Vecchi M.P. "Optimization by simulated annealing," Science, 220:671-680, 1983
64. Knowles, J. D., "Local Search and Hybrid Evolutionary Algorithms for Pareto Optimization," Doctoral Dissertation, The University of Reading, 2002.
65. Knowles, J. D. and Corne, D. W., "Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy ," Evolutionary Computation 149-172, 2000
66. Laumanns, M., Thiele, L., et al., "Combining convergence and diversity in evolutionary multi-objective optimization, " Evolutionary Computation, vol. 10, no. 3, pp. 263-282, Fall 2002
67. Lohn, J.D., Kraus, W.F., and Haith, G.L. "Comparing a Coevolutionary Genetic Algorithm for Multiobjective Optimization," Proceeding of the 2002 IEEE Congress on Evolutionary Computation, pp. 1157-1162, May 2002.
68. Lu, H. and Yen, G., "Rank-Density-Based Multiobjective Genetic Algorithm and Benchmark Test Function Study," IEEE Transactions on Evolutionary Computation, VOL 5, NO.4, P335-348, August 2001.
69. Luengo, K. and Raydon, M. "GRADIENT METHOD WITH DYNAMICAL RETARDS FOR LARGE-SCALE Optimization problems." Electronic Transactions on Numerical Analysis, Volumn 16, pp. 186-193, 2003
70. Man, K. F., Tang, K. S., and Kwong. S. *Genetic Algorithms: Concepts and Designs*. Springer, New York, 1999.
71. Marrison, C. I. and Stengel, R. F. "Robust Control System Design Using Random Search and Genetic Aglorithm," IEEE Transactions on automatic control, VOL 42, NO 6, P835-839, 1997
72. Michalewicz, Z. "Gene Genetic Algorithms + Data Structures = Evolution Programs," Springer-Verlag Berlin Heidelberg 1994.
73. Mostaghim, S. and Teich, J., "The role of ε- dominance in multi objective partical swarm optimization methods," In Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003), vol. 3, pp. 1764-1771, Canberra, Australia, IEEE Press, Dec. 2003

www.manaraa.com

74. Neelamkavil, F.. "Computer simulation and modeling," John Wiley & Sons Inc, 1987
75. S. R. Nassif, "Modeling and Forecasting of Manufacturing Variations",*Proc. ACM/IEEE ASP-DAC*, 2001.
76. NEELAMKAVIL F., *Computer Simulation and Modeling*, John Wiley & Sons Inc, 1987.
77. NILSSON K., ANDERSSON J. AND KRUS P., "Method for Integrated Systems Design – A Study of EHA Systems", in *Proceedings of Recent Advances in Aerospace Hydraulics*, Toulouse, France, November 24-25,1998.
78. Oliveira, L.S., et al, "Feature Selection Using Multi_Objective Genetic Algorithms for Handwritten Digital Reconition," in 16th ICPR, P568-571, 2002
79. Oppacher, F. and M. Wineberg, "The Shifting Balance Genetic Algorithm: Improving the GA in a Dynamic Environment," in Proceedings of the Genetic and Evolutionary Computation Conference, Vol. 1. pp504-510, 1999.
80. Packham, I S J and Parmee, I. C., "Data analysis and Visualization of Cluster-Orient Genetic Algorithm Output," Proceedings of the International Conference on Information Visualization pp173-178, 2000
81. Pareto V., "*Cours D'Economie Politique*. Lausanne, " Switzerland: F. Rouge, vol. I and II, 1896
82. Parmee. I.C., et al., "Interactive Evolutionary Conceptual Design Systems,"In Proceedings of International Conference on Artificial Intelligence in Design, Worcester, Mass. USA; June 2000
83. Prasad, T. D., "Multiobjective Genetic Algorithms for Design of Water Distribution Networks," *Journal of Water Resources Planning and Management*, Vol. 130, No. 1, pp. 73-82, January/February 2004
84. Pohlheim, H "GEATbx: Genetic and Evolutionary Algorithm Toolbox for use with MATLAB", http://www.systemtechnik.tu-ilmenau.de/~pohlheim/GA_Toolbox/ 1997
85. Pohlheim, H. "Visualization of Evolutionary Algorithms – Set of standard Techniques and Multimensional Visualization," Proceedings of the Genetic and Evolutionary Computation Conference, P533-540, 1999
86. Powell, M.J.D., Nonconvex Minimization Calculations and the Conjugate Gradient Method, Lecture Notes in Mathematics, Vol. 1066, pp. 122-141, 1984

www.manaraa.com

87. Qi, R., Vittal, V., Kliemann, W., and D., C., Visualization of Stable Manifolds and
88. Multidimensional Surfaces in the Analysis of Power System Dynamics. Journal of Nonlinear Science, 10:175-195. 2000.
89. Raymer, M. L., et al., "Dimensionality Reduction Using Genetic Algorithm," IEEE Transactions on Evolutionary Computation, VOL 4, NO 2, P164-171, July 2000
90. Reardon, B. J., "Fuzzy logic versus niched Pareto multiobjective genetic algorithm optimization," Modeling Simul. Mater. Sci. Eng. **6** (1998) 717–734, 1998
91. Raghuwanshi, M.M. and Kakde, O. G., "Survey on multiobjective evolutionary and real coded genetic algorithms," Asia Pacific Symposium on Intelligent and Evolutionary Systems 2004, P 115 – 126, 2004
92. Ray, T., "Constrained robust optimal design using a multiobjective evolutionary algorithm," in *Proc. Congr. Evol. Comput*, pp. 419–424, 2002
93. RAO S. S., *Engineering Optimization*, John Wiley & Sons Inc, 1996
94. Roy. R and Sevick-Muraca, E. , "A numerical study of gradient-based nonlinear optimization methods for contrast enhanced optical tomography," Opt. Express **9,** 49-65, 2001
95. Roosenburg, N. and Eekeks, J.. "Product Design: Fundamentals and Methods," John Wiley & Sons Inc, 1995.
96. Shimodaira, H., "A Diversity Control Oriented Genetic Algorithm (DCGA): Development and Experimental Results," In Proceedings of the Genetic and Evolutionary Computation Conference, Vol. 1. pp 603-611, 1999.
97. Singh, A., Minsker, B. S., and D. Goldberg, "Combining Reliability and Pareto Optimality - An Approach Using Stochastic Multi-Objective Genetic Algorithms," American Society of Civil Engineers (ASCE) Environmental & Water Resources Institute (EWRI) World Water & Environmental Resources Congress 2003 & Related Symposia, Philadelphia, PA, 2003.
98. Shine, W. B. and Eick, C. F., "Visualizing the Evolution of Genetic Algorithm Search Processes," IEEE International Conference on Evolutionary Computation, April 1997.
99. Simon H.. *The Sciences of the Artificial*, MIT press, 1969.
100. Skogestad, S. and L. Postlethwaite, "Multivariable Feedback Control," Wiley, England, 2001.

www.manaraa.com

101. Srinivas, N. and Deb, K., Multiobjective Optimization Using Non-dominated Sorting in Genetic Algorithms, *Evolutionary Computation*, volume 2(3), pp. 221-248, fall 1994.
102. Shipton, S. "Bifurcation Phenomena in a Channel with an Expanded Section," 2000 ASME Fluids Engineering Summer Conference, 2000
103. Spears, W., "An overviewof multidimensional visualization techniques," in *Proceedings of the 1999 Genetic and Evolutionary Conference Workshop Program*, A.Wu, Ed. San Mateo, CA: Morgan Kaufmann, pp. 104–105, 1999
104. Taylor, J. T., "From Artificial Evolution to Artificial Life," Doctor Dissertation, University of Edinburgh, 1999.
105. Teich, J., "Pareto-Front Exploration with Uncertain Objectives," Evolutionary Multi-Criterion Optimization, First International Conference, pp314-328, 2001.
106. Ursen, B.K, "Diversity-Guided Evoluationary Algorithm," In: Proceedings of Parallel Problem Solving from Nature VII (PPSN-2002), p. 462-471, 2002
107. Venkataraman, P., "Applied Optimization with MATLAB Programming," John Wiley & Sons, Inc, 2001
108. Weeks, R. W. and Moskwa, J.J., "Automotive Engine Modeling for Real-Time Control Using MATLAB/SIMULINK," SAE 950417, 1995
109. ZIMMERMANN H.-J. AND H.-J. S., "Intelligent system design support by fuzzymulti-criteria decision making and/or evolutionary algorithms," in *Proceedings of IEEE International Conference on Fuzzy Systems*, Yokohama, Japan, 1995
110. Zitzler, E., Deb, K. and Thiele, L. "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," Evolutionary Computation, 8(2), pp. 173-195, Summer 2000
111. ZITZLER E. AND THIELE L., "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transaction on evolutionary computation*, vol. 3, pp. 257-271, 1999
112. http://www.mathtools.net/MATLAB/Optimization/Genetic_algorithms/ Mathworks.
113. Zitzler, E., Laumanns, M. and Thiele. L., SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.

www.manaraa.com