AIAA-98-4726

DEVELOPMENT OF A FLEXIBLE DESIGN OPTIMIZATION STUDY TOOL

Dipankar K. Ghosh*, John H. Garcelon*
Vladimir O. Balabanov* and Garret N. Vanderplaats†

Vanderplaats Research & Development, Inc. 1767 S. 8th Street, Suite M210
Colorado Springs, CO 80906

ABSTRACT

A flexible design study tool to couple optimization technology with third-party analysis programs is presented. The main objective of such a design study tool is to provide engineers with the necessary optimization tools so that optimization capabilities could be easily interfaced with analysis programs. This will provide the optimization capabilities where they do not exist. The basic architecture of the design study tool has been developed, and components are being added on an-ongoing basis. The architecture of this study tool is based on an object-relational database system, and object oriented programming methodology. Preliminary interfaces to the third-party software ABAQUS and LSDYNA3D have been developed, and similar interfaces to CAD/CAE packages are being considered. The development of such a study tool will provide the framework for developing multidiscipline design optimization (MDO) capabilities incorporating design information available from different disciplines of interest.

INTRODUCTION

Although optimization technology has received much attention in the engineering community, it has not yet received the degree of acceptance that might be expected considering the power of this technology. Two key reasons have been identified that contribute to this lack of interest. The first reason is the cost of adding optimization technology to existing commercial software when a clear market has not yet been established. The second reason is the fear that optimization is a specialized technology, and thus requires special expertise to use it. The first issue can be addressed by providing the capability to use optimization with existing analysis programs without the need for extensive programming. Although this approach is less efficient, it lets the engineers use optimization in the short term, and demonstrates its value. This may provide the necessary motivation to the analysis program developers to make the investment in a more closely coupled optimization capability in the future. The second issue can be addressed by providing user-friendly software, which includes real time, post-processing information that graphically demonstrates the optimization process The purpose of

* Senior R&D Engineer, Member AIAA
† President, Fellow AIAA

the present development efforts is to address these issues and develop the necessary commercial software tools.

The development of such an optimization study tool has started, and this will allow the coupling of optimization technology with a number of commercially available analysis packages as well as user written analysis software. The development of the overall architecture of this graphics-based design study tool is based on an object-relational database system and object oriented programming.

The development of this design optimization study tool starts with the development of a graphical user interface (GUI) to the general-purpose optimization software DOT (Design Optimization Tools)[1] and DOC (Design Optimization Control)[2]. Herein DOT serves as one of the many functional modules that are planned to be incorporated in the final product. Other functional modules being incorporated are Design of Experiments (DOE), Response Surface (RS) approximations, Genetic Algorithms (GA) and a post-processing module. Interfaces to a number of third-party analysis and CAD/CAE software are also being developed. Preliminary interfaces to ABAQUS[4] and LSDYNA3D[5] software packages have been completed. These interfaces provide users with optimization capabilities while using nonlinear analysis codes. The development of an interface to MSC/PATRAN has also been started.

In this paper, we present the philosophy behind the overall design study system, and a brief overview. Although the components of this design study tool are being developed with single disciplines in mind, this could easily be extended to multiple disciplines, and thus will provide the framework for developing multidisciplinary design optimization (MDO) capabilities.

THE DESIGN PHILOSOPHY

The design study process starts with the definition of a design project. General information about design variables, objective functions, constraints etc. is stored in the object-relational database. Using this design project information, the user specifies which functional module will be used for optimization and the design study tool creates the design data for that particular functional module and stores them in the database. Then the design study tool invokes that functional module to solve the problem. Once the functional module has finished, the results are transferred back to the database for later use. The user may now solve the same design problem using a different functional module with the same or modified control parameters, and thus keep building his/her design database. The user may use the updated design database to redefine his problem attributes, and/or perform post-processing activities. Thus this design study tool not only performs design optimization but also provides insight to the design problem itself.

THE PROGRAM ARCHITECTURE

The development of the overall program architecture is based on an object-relational database system. The key components of this system are (i) a central graphical user interface (GUI), (ii) an object- relational database management system, (iii) a suit of functional modules and (iv) interfaces to a number of third-party analysis and CAE/CAD programs. This is shown in Fig.1

Graphical User Interface (GUI)

The basic philosophy of the GUI is to create a fully automated, graphical design environment that is easy to use for the novice, but provides the flexibility

needed by the advanced user. Users interact with the program using a spreadsheet metaphor or window based forms. The data consistency checks are performed at two levels namely, (a) at the first level, when the user enters data into spreadsheet/form(s) and (b) at the second level, when this data is transferred to the database. At the pre-processing stage, the design model is defined. For example, the user will specify the design variables, objective functions, and constraints, if any. The user will also specify which functional module to use and any necessary control parameters. At the post-processing stage, the user may specify which results are to be retrieved from the database and graphically displayed.

The GUI stores all design data in the database using an object-relational format. A prototype interface using the DOT functional module and the database has been implemented. This will be further extended and refined for both gradient and non-gradient-based functional modules. The next stage of implementation will involve interfacing the GUI to the database for different functional modules as they are implemented.

An Object-Relational Database Management System

We have developed and tested an object-relational database engine. This database engine is based on a public domain database engine[6], which we have significantly modified and extended.

This database engine uses traditional relational components such as indexed b-trees, linked lists, and random, binary file access. B-trees provide fast lookup (O(log n)) of random data as opposed to sequential access (O(n)). Here 'n' represents the number of relations in the database. Binary files provide compact fast file I/O compared to larger text file access.

The object-oriented features of the database extend the relational capabilities. Traditional basic relational databases use fixed length records; however, the object-oriented database extensions allow variable length records even with relations.

Another advantage of the object-oriented database is that the relations are simply and directly derived from the objects used in the user interfaces. This reduces code complexity, allows for code reuse, and reduces code maintenance.

A Suit of Functional Modules

This design study tool contains a number of functional modules. The basic motivation here is to provide extended design optimization capabilities, and several tools to assist this task. These modules are briefly described next.

DOT[1] serves as the core nonlinear, gradient based, optimization tool. For constrained optimization, it provides three algorithms, namely Modified Method of Feasible Directions (MMFD), Sequential Linear Programming (SLP), and Sequential Quadratic Programming (SQP). For unconstrained optimization, it provides two algorithms, namely BFGS and Fletcher-Reeves.

A Genetic Algorithm (GA) module[7] provides an opportunity for non-gradient based optimization. This module currently exists as a stand-alone program, and will soon be incorporated in the main program.

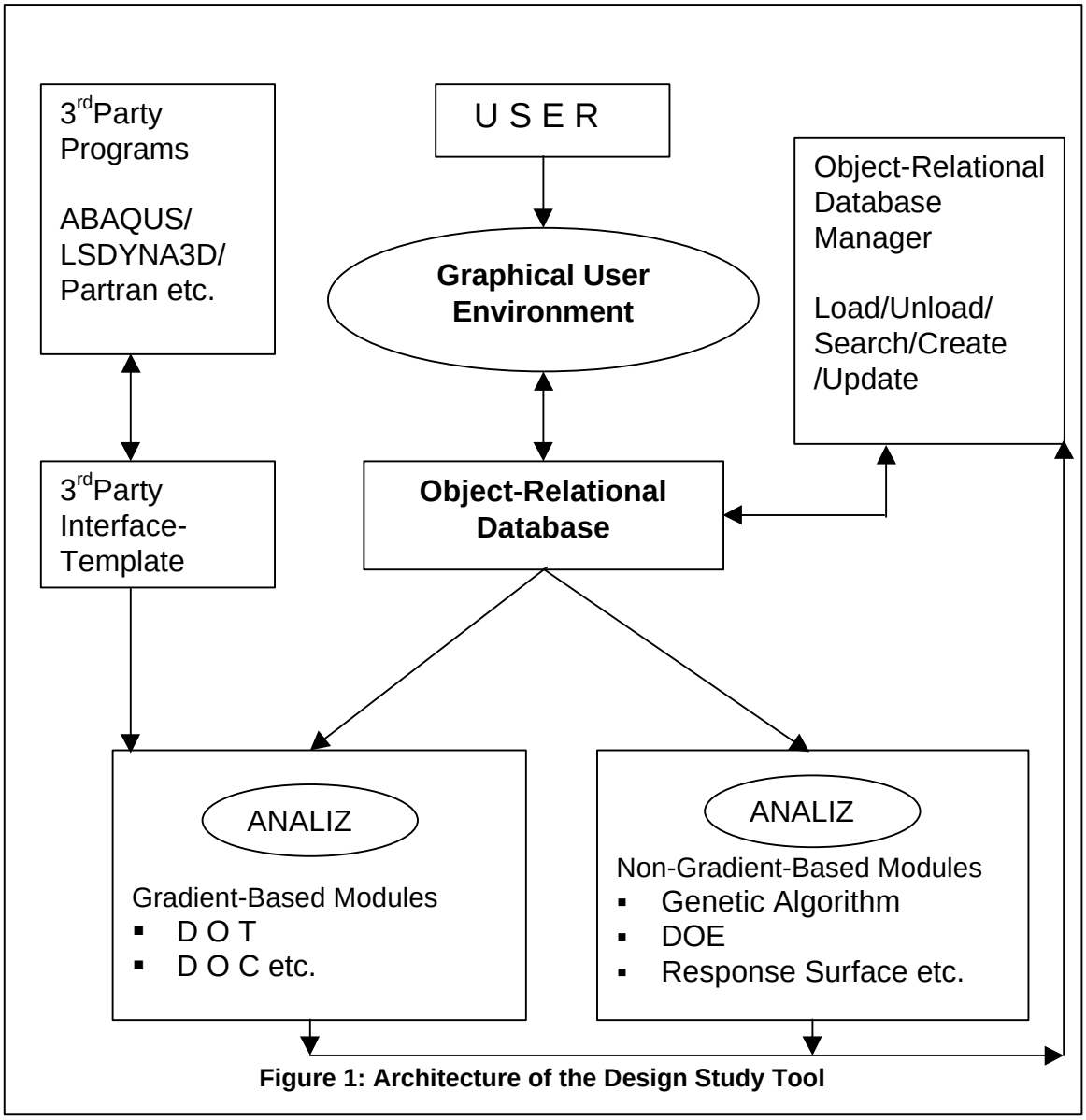

Figure 1: Architecture of the Design Study Tool

Design of Experiments (DOE) and closely related Taguchi methods can be used as tools to provide improved product quality and assist in correcting approximate models, and thus complement the formal optimization methods. The development of DOE module is complete and presently is available as separate program.

Response Surface (RS) methodology for approximately calculating the design responses is also provided as a separate functional module. This is available in the DOC program, and will be further enhanced and improved.

Using the object-relational database, we developed and tested a prototype design database and interface it to a prototype functional module. This prototype demonstrated the feasibility and basic functionality required of the database engine, of the database itself, and of an interface between functional modules and the database. The prototype functional module was based on DOT. We are currently expanding the scope of

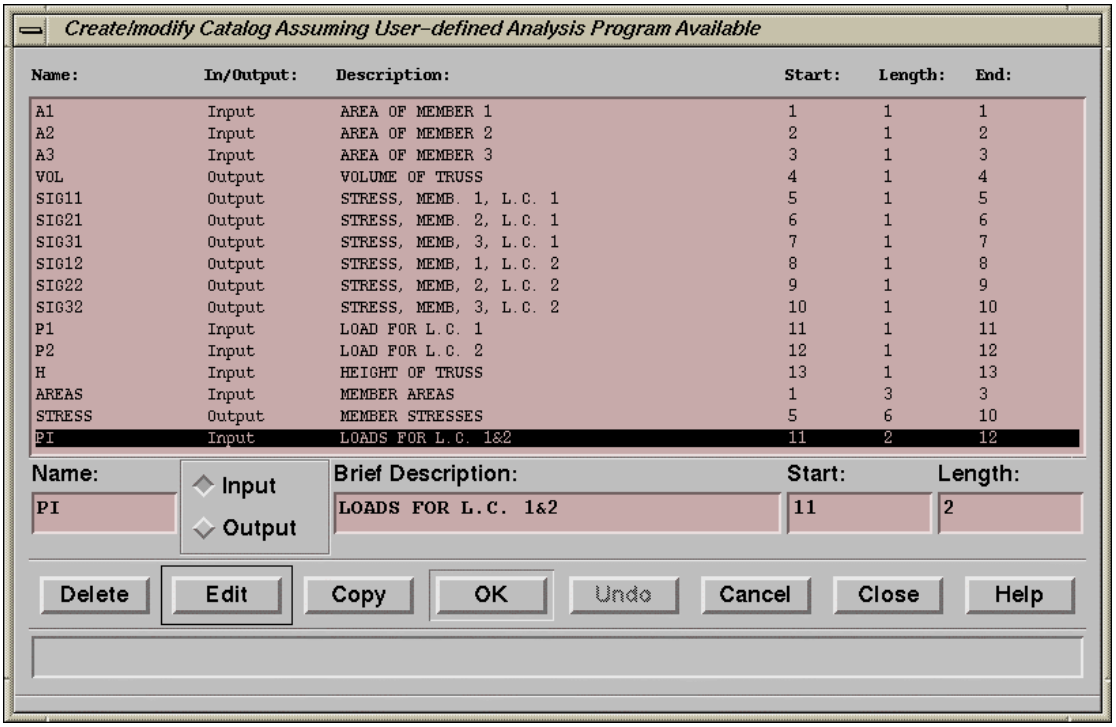

Figure 2: Create a Catalog of Input and Output Parameters

this gradient-based functional module to incorporate additional features like synthetic and linked variables, multiobjective optimization, discrete and integer variables etc. These are parts of the existing DOC product.

Interfaces To Third-Party Software

The design study tool provides the facilities to link analysis programs with the functional modules. Both user supplied analysis programs where source code is available, and commercially available analysis programs where source code is not available may be linked to the functional modules.

When using user written analysis codes, it is possible to modify the source, and link it directly with the functional module. Currently, these links are performed by creating an analysis subroutine called ANALIZ (in DOC terminology) and link it directly with the functional module to create the most efficient executable code.

For users who either do not want to modify their code or who do not have access to the source code, the functional modules must have access to the data file(s) read by the analysis program and output file(s) created by the analysis program. This is the most flexible approach, but it is less efficient due to the fact that the analysis program must be repeatedly loaded and executed.

Developments of interfaces to a number of third-party analysis programs are in progress. Preliminary interfaces to the nonlinear structural analysis codes ABAQUS[4] and LSYDNA3D[5] have already been developed. More closely coupled interfaces to these two programs will follow soon. A number of analysis programs from various disciplines like computational fluid dynamics (CFD), mold flow analysis etc. are considered for implementation in near future. The works on interfaces to CAD/CAE tools are also in progress. Using these interfaces, users

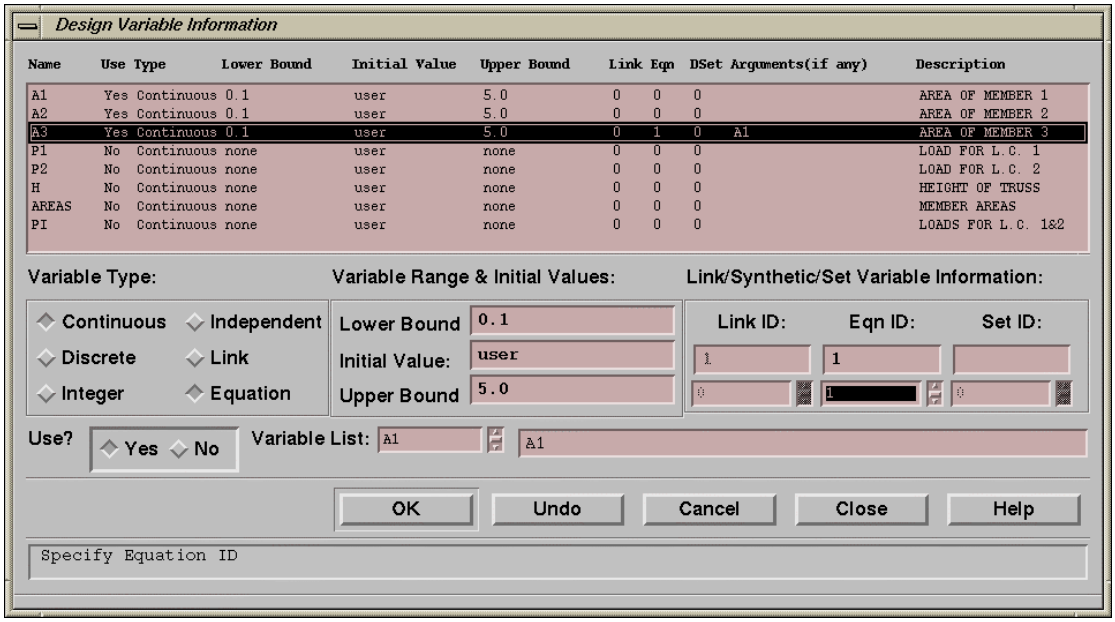

Figure 3: Create Design Variable Information

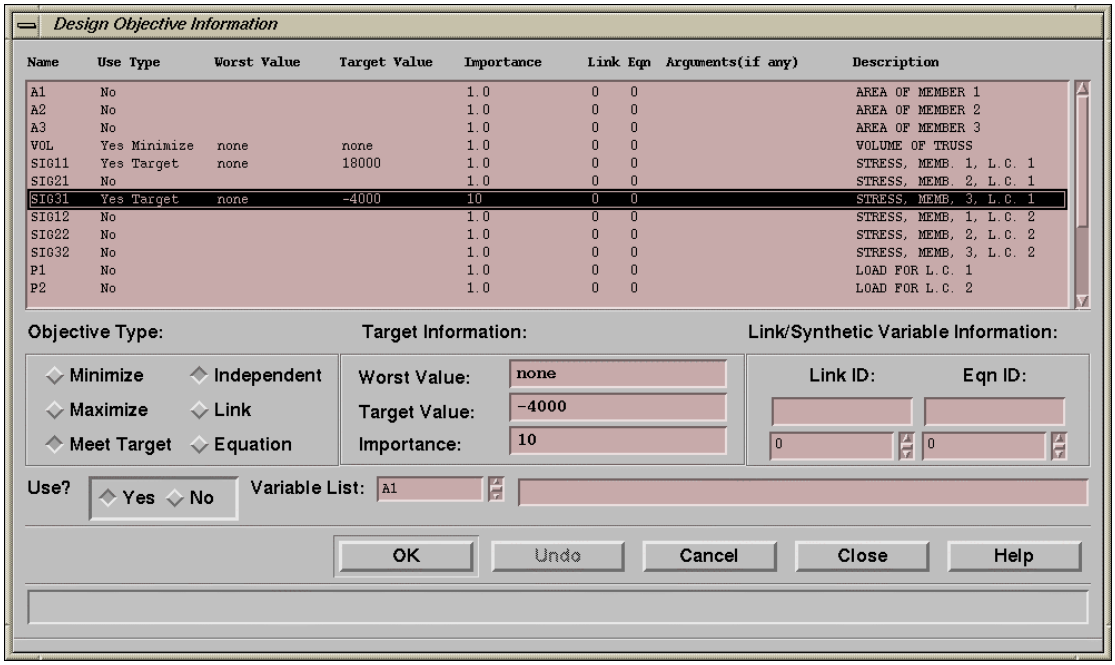

Figure 4: Create Objective Function Information

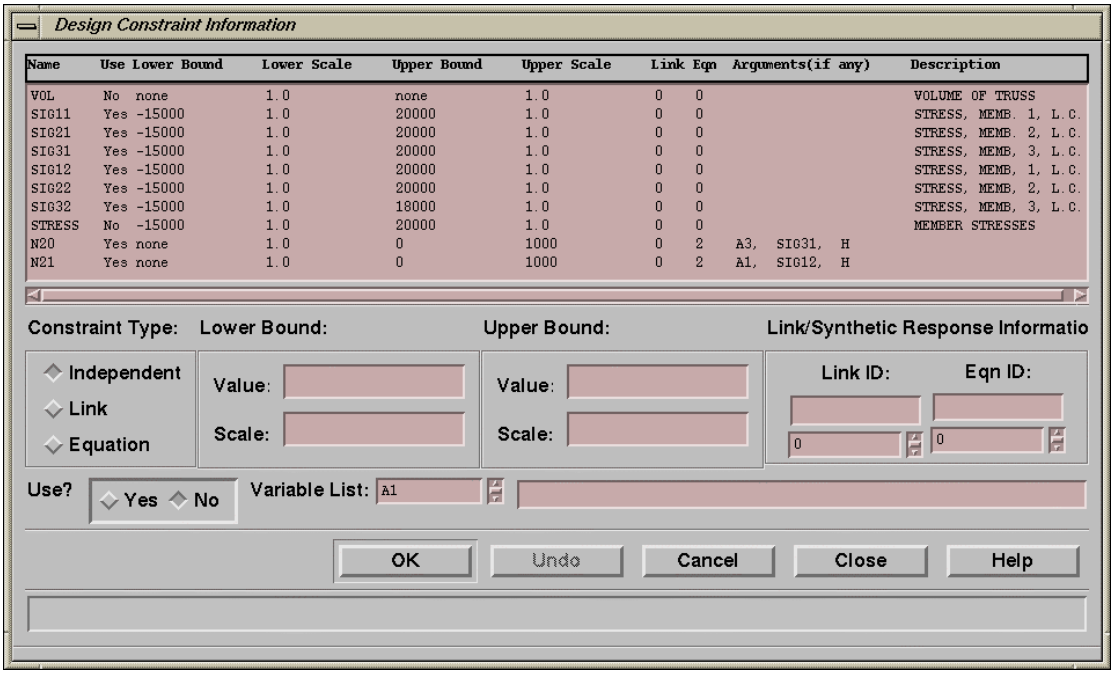

Figure 5: Create Constraint Information

will be able to create their design data directly from within a CAD/CAE tool, and run optimization using the functional module of their choice. The development of an interface to MSC/PATRAN has been started.

THE DESIGN PROCESS

The definition of the design project starts with definition of a catalog of design variables and responses. The user simply lists a number of possible input and output variables using the ***catalog*** window (see Fig.2).

Once the catalog of variables and responses are available, the user can open the ***variable*** window to see a list of possible design variables (see Fig.3).

All the catalog items that were specified as input are listed here. Using this window, the user can specify which of these variable values he/she wishes to change during optimization. Next, the user may open the ***objective*** window to see a list of possible objective functions (see Fig. 4). Here all catalog items, both input and output variables to the analysis program are treated as possible objective functions. The user may select one or more items as objective functions. Here multi-objective optimization is allowed. Objective function(s) could be linked linearly or nonlinearly with other independent design variables and responses. Next, the user may open the ***constraint*** window (see Fig. 5). Here, only those parameters specified as output in the catalog are shown. The user may specify one or more of these responses as constraints. Additionally, he/she may specify lower and upper bounds, scaling factors and other related information.

There are several other windows available to specify the control parameters (e.g., type of optimization algorithm to be used, print control parameter etc.), define discrete variable sets (if discrete optimization is desired), synthetic functions (i.e., equation), input candidate design and response values (if RS approximation is used) etc.

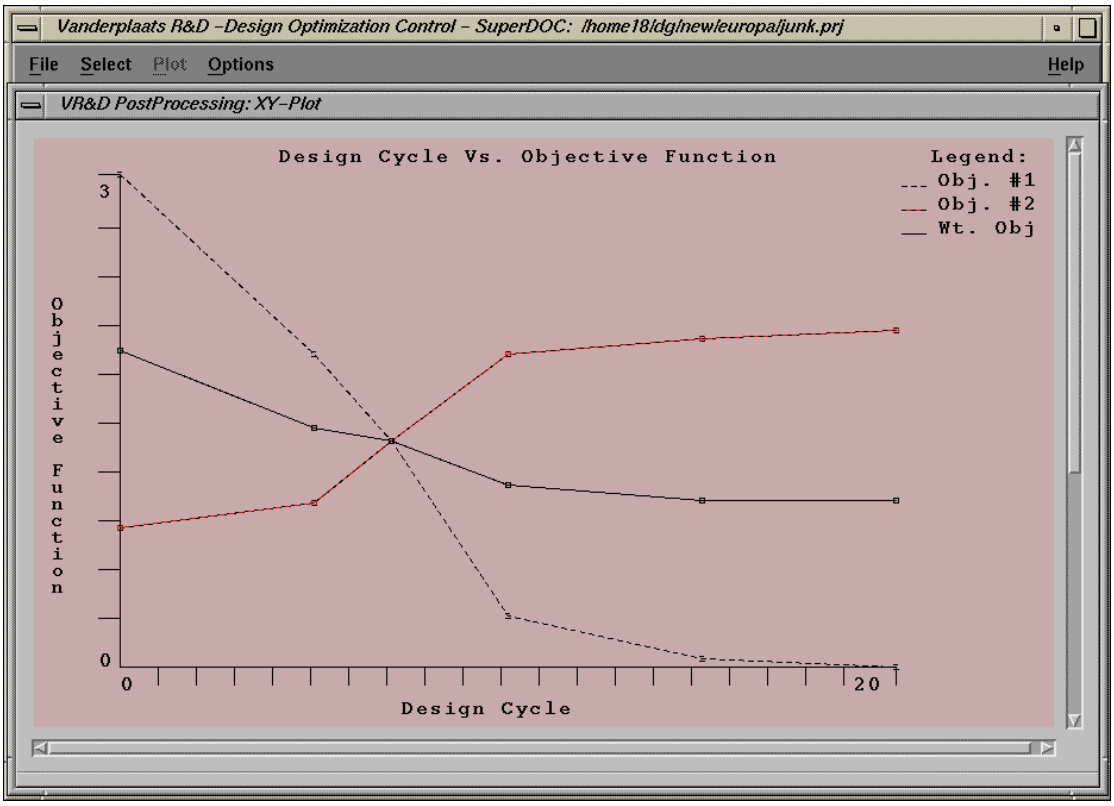

Figure 6: Objective Functions Vs. Design Cycle

Once the creation of design project data is complete, it is stored in the database. The user may then create the design data for a particular functional module and store it into the database. The user then runs the functional module of his/her choice. This could be done both from within windows environment, and from outside the windows at the console prompt. At the end, the results are transferred to the same database for later use.

POST PROCESSING

The purpose of this module is to assist the user in reviewing the optimization or other results, and in performing "what-if" studies. The user may wish to trace the optimization progress to gain insight into his/her design. This could be done in several ways. The user may simply plot the objective function(s) versus iteration time. If there are multiple objective functions, the plotting the progress of each of them in the form of xy-plot will provide insight into how they compete. The Fig. 6 shows such a plot as an example. Additionally, the user may plot the gradient of the objective function(s) and active and near active constraints, as well as the resulting search direction. The key idea here is that the user can view important information about the design progress. Thus, the user can choose to continue, terminate or perhaps modify the design specifications to direct the optimization process.

AN EXAMPLE USING ABAQUS

Here we present the design of composite stiffeners of a pallet using ABAQUS as the third-party analysis program. The objective is to minimize the mass subject to frequency, local buckling, and stress constraints. The model representing a

pallet consists of 5961 shell and general section beam elements and has 9462 degrees of freedom. The pallet was modeled using composite shell elements and the T-section stiffeners were represented using offset general section beams. The following parameters were allowed to change in the design process: palette skin, thickness, stiffener web and flange thickness, width of the stiffener flange, and the height of the stiffener web. These parameters were treated as independent design variables. The stress and local buckling constraints had to be calculated for each element, and the lowest natural frequency needed to be greater than 6Hz. There were total of 119,386 constraints for this design problem. The lowest natural frequency of the initial design was 3Hz, which indicates that the initial design was highly infeasible.

The size of the model and the large number of constraints involved made ABAQUS output file very large. Also the local buckling of composite panels were not directly calculated by ABAQUS. Her we calculated the local buckling constraints by recovering the sectional forces at several points for each beam and composite shell elements, and performed the buckling calculations outside of ABAQUS. ABAQUS binary result file was read directly and then the objective function and constraints were calculated. This type of data transfer (as opposed to reading ASCII output file) was very efficient and reliable. The optimization required 42 ABAQUS analyses to locate the optimum, and each analysis required about 5 minutes of the clock time on an SGI Indigo workstation. The optimum design showed substantial improvements in the response characteristics. The frequency constraint was satisfied for the optimal design, and the mass of the structure was increased by about 6.5%. The final design was feasible and satisfied all the constraints.

ACKNOWLEDGMENT

The research and development described herein is supported by the U. S. Air Force under Small Business Innovative Research (SBIR), Phase II Project Number F336125-97-C-3204.

We would also like to thank Prof. P. Hajela, Rensselaer Polytechnic Institute, for providing us with the Genetic Algorithm Code.

REFERENCES

1. "DOT Users Manual," Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1995.

2. "DOC Users Manual," Vanderplaats Research & Development, Inc., Colorado Springs, CO, 1995.

3. Ghosh, Dipankar, and Vanderplaats, Gary, 'Development of a Flexible Design Optimization Capability', presented at the 'Optimization In Industry' Conference Palm Coast, Florida, March 23-27, 1997, and also published by ASME International, American Society of Mechanical Engineers (ASME), (1997).

4. Ghosh, Dipankar, and Vanderplaats, Gary, “Development of a Design Optimization Interface to ABAQUS”, presented at the ABAQUS Western Users Regional Conference, Hilton and Towers Airport Hotel, Los Angles, California, Sept’ 25, 1997.

5. Ghosh, Dipankar, and Vanderplaats, Gary, “Development of a Design Optimization Interface to LS-DYNA”, to be presented at the 5th International LS-DYNA Users

Conference, Southfield, Michigan, September 21-22, 1998

6. Stevens, Al, C++ Database Development, MIS: Press, 1994, New York, New York.

7. EVOLVE: A Genetic Search Optimization Code, Department of Mechanical Engineering, Aerospace Engineering and Mechanics. Rensselaer Polytechnic Institute (1993).