

The optimization paradigm in engineering design: promises and challenges

Panos Y. Papalambros*

Optimal Design Laboratory, Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI, USA

Abstract

Formal design optimization involves application of mathematical optimization techniques to models derived from engineering science, often presented as computer simulations. Industry use of design optimization tools is now widespread and sophisticated. As the size and complexity of design problems addressed with these formal methods increase, so do the challenges for successful implementations. Based on the author's experiences, after briefly reviewing the mathematical challenges involved, the article describes several problem areas where optimization technology is likely to have a major impact in the development of improved artifacts in the coming years. These include conceptual and topological design, large complex systems, smart products, and enterprise-wide product design. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Optimization; Surrogate models; Topology design; Decomposition; Optimal control; Platform design

1. Introduction

'Optimization' is a term that is frequently and widely used in the description and conduct of design processes for product development. Broadly speaking optimization means improving or fine-tuning the design in terms of one or more performance aspects. However, there is a very specific technical meaning of 'optimization' as a rigorous mathematical statement. The basic assumption behind such a statement is that the design process is viewed as a decision-making process, whereby one selects the proper functional form among many alternatives. The embodiment of the given functional configuration is then developed by further selecting appropriate values for the design quantities that fully describe the configuration at hand. A formal optimal design model may apply to:

- the functional form (configuration or topology) of an artifact;
- the embodiment (proportioning and/or shape) of a given configuration;
- both of the above.

It is worth noting that the wealth of problems handled in this formal way has dramatically increased over what was

anticipated during the previous decade, particularly for the earlier stages of the design process [7,16].

The formal mathematical model of the optimization problem is a statement of the form:

$$\begin{aligned} & \text{minimize } f(x) \\ & \text{subject to:} \\ & h(x) = 0 \\ & g(x) \leq 0 \\ & x \in \chi \subseteq \mathcal{R}^n \end{aligned} \quad (1)$$

where the scalar objective function $f(x)$ provides the comparison criterion among different alternatives, the vector-valued functions $h = (h_1, h_2, \dots, h_m)^T$ and $g = (g_1, g_2, \dots, g_m)^T$ and are the *functional constraints* that determine whether a design is feasible, and x is the n -dimensional vector of the design variables, where n is finite. In many embodiment design problems the variables take continuous real values in the n -dimensional real space \mathcal{R}^n . However, even in proportioning problems variables may take only discrete values, such as standard sizes of cross-sections. In many configuration design problems the variables are defined with discrete values. For example, allowing only 0–1 values we can model the decision whether a component is used or not. In Eq. (1) the type of values used is described by the set χ , the *set constraint*. Models with continuous variables are generally easier to solve with techniques

* Corresponding author.

E-mail address: pyp@engin.umich.edu (P.Y. Papalambros).

based on differential calculus. Discrete problems are combinatorial in nature and a truly optimal solution is difficult to identify without some sort of complete enumeration of all possible combinations, a task that is often impractical. Frequently the natural development of the design model will indicate two or more objective functions that are competing, leading to the *multiobjective* or *multicriteria* problem:

$$\begin{aligned} &\text{minimize } c(x) \\ &\text{subject to:} \\ &h(x) = 0 \\ &g(x) \leq 0 \end{aligned} \quad (2)$$

where c is the vector of I real-valued criteria c_i , $i \in I$. The feasible values for $c(x)$ constitute the *attainable set* A . The multicriteria formulation is converted into a *scalar substitute problem* of the general form:

$$\begin{aligned} &\text{minimize } f = \sum_i f_1(\omega_i) f_2(c_i, m_i) \\ &\text{subject to:} \\ &h(x) = 0 \\ &g(x) \leq 0 \end{aligned} \quad (3)$$

where the scalars ω_i and vectors m_i are preference parameters. Design preferences are rarely known precisely a priori, so preference values are adjusted gradually and trade-offs become more evident with repeated solutions of the substitute problem with different preference parameter values. A common preference is to reduce at least one criterion without increasing any of the others. Under this assumption the set of solutions is reduced to a subset of the attainable set, the *Pareto set*. So in multicriteria minimization a point in the design space is a *Pareto (optimal)* point if there exist no feasible point that would reduce one criterion without increasing the value of one or more of the other criteria.

The functions f , h , and g can be explicit algebraic expressions but may also be the formal statement of a complex procedure involving internal calculations and realized only as a computer program—what is often called a *simulation model*. Finite element analysis and computational fluid dynamic analysis are examples of simulation models. More generally, analysis models are the computational means available for evaluating the functions f , h , and g , given some value of x . They can be simple algebraic expressions or complex simulation models. In contrast, an (*optimal*) design model is the decision model defined in Eq. (1). Solution of Eq. (1) typically involves generating a sequence of design alternatives, and each design alternative requires the analysis models to evaluate it. The cost of evaluating each design alternative is often a major consideration in using optimization. Finally, if the design vector x has a finite

number of components then Eq. (1) is called a *mathematical programming problem*. But if some variables are defined in an *infinite dimensional* space, for example, $x_i = x_i(t)$, $t \in \mathcal{R}$, then we have a *variational problem*. Controller design problems and some structural shape and topology problems are posed as variational ones.

Early work on decision-making formalisms for mechanical design appeared in the 1960's. Much of that work focused on designing mechanical components and structures and the challenges associated with solving the underlying mathematical problem. Model building was limited to relatively simple analysis models. For two decades the subject remained a somewhat esoteric one, primarily for academics. Advances in numerical analysis and computing capabilities, but more so in sophisticated modeling techniques, have revolutionized the field in the last decade. Efficient finite element modeling has made structural optimization a routine tool for practicing engineers, who can now solve problems thought impossible only a dozen years ago. Similar capabilities in multibody dynamics modeling, computational fluid dynamics, and parametric CAD tools are bringing mathematical optimization into the product development mainstream. Availability of accessible and reasonably robust commercial optimization software has also contributed greatly to the increased use of mathematical optimization in design.

Yet, in some ways, design optimization remains an unfulfilled promise:

- *Mathematical challenges.* Solution algorithms are not always robust enough for the uninitiated user, and the iterative processes required to reach an answer can be prohibitively expensive in computing time.
- *Topologies and configurations.* The toughest design problems involve creating a proper topology or configuration (in this article we use the terms synonymously); impressive advances have been made in optimal topology design of structures. Optimal configuration design of general mechanical systems that may contain many parts, including structural components, remains beyond the reach of present capabilities.
- *Systems design.* Optimal proportioning of single components is largely a solved problem. Yet for a given configuration of a system, such as an aircraft or automobile, it is still a challenge to find the optimal proportioning of components and subsystems that work together to achieve some overall system design goals.
- *Controlled artifacts.* The continuing trend towards more tightly integrated electromechanical systems at a macro- and microscale has raised the question on how can artifact embodiment and its associated controller be optimally designed in a concurrent fashion, as opposed to the present situation where artifact and controller design are largely treated as sequential design problems.
- *Enterprise-wide design.* Design optimization must

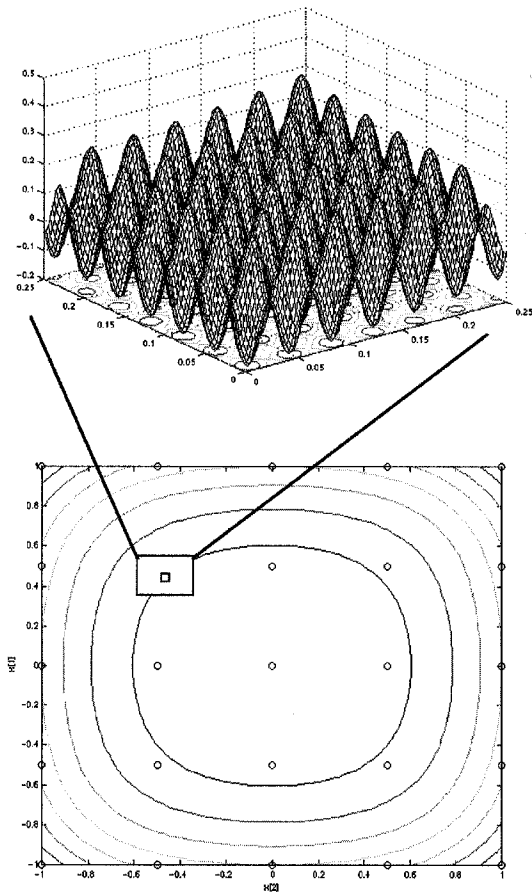


Fig. 1. Noise in functions generated by simulation-based models.

move beyond the boundaries of technological product performance and attempt to model and solve design problems that include broader issues, from bringing a product mix to the marketplace to product retirement.

In the following sections we will address the above issues attempting to define them more closely and to identify the challenges involved.

2. Mathematical challenges

The mathematical optimization problem in its general form is difficult when the functions in Eq. (1) are nonlinear. In this section we review some of the key issues in the context of real engineering systems, and what can be done to address them. More information and references can be found in Ref. [16].

2.1. Local and global optimization

The usual mathematical assumption is that all model functions are continuous and differentiable with respect to all design variables. When this is mathematically proven for a given problem, numerical algorithms that use function and derivative information can locate a local minimum in a

relatively robust manner. However, the possibility of multiple minima still remains. Finding the global minimum is generally an intractable problem as the number of design variables increases.

One way to think about this difficulty is as follows. Optimality conditions for a local minimum of a continuous differentiable function constrained over a closed and bounded (compact) constraint set are using local slope and curvature information to determine that *locally* around a given point there exist no others with better values. These conditions are operationally useful because we can compute specific quantities involving gradient vectors and Hessian (second derivative) matrices that allow us to generate candidate solutions and test them for optimality using only local data. In contrast, there are no equivalent optimality conditions for global optimization. The only real test is to apply the definition of the global optimum, namely that there are no better points anywhere in the feasible domain that give a better value for the objective function. This test can be performed by enumerating all possibilities, either explicitly or implicitly. In an implicit enumeration scheme, such as general branch and bound techniques, we try to eliminate parts of the feasible domain by proving they cannot contain an objective value better than one we have computed already at a tested point. Systematically shrinking the set of possible better solutions will eventually lead to the global optimum.

Local gradient-based optimization algorithms have been extensively developed and are widely used. Design optimization applications present some difficulties because the mathematical assumptions required for proper performance of the algorithm are often hard to test. If model functions involve implicit simulations, such as finite element or multi-body dynamics analyses, these functions are not really known as mathematical entities; rather, they are input/output relations generated by a 'black box'. Therefore it is difficult or impossible to say whether they are continuous and differentiable functions, let alone convex, as proper convergence proofs often require, even in the neighborhood of the solution. Furthermore, errors associated with a variety of sources in numerical digital processes can result in 'noise', as in Fig. 1. A gradient algorithm will get trapped in local minima generated by the numerical noise and fail to reach the true design optimum.

Global methods, whether gradient-based or derivative-free (see, e.g. [19]), present primarily two challenges: they become unacceptably inefficient when the number of variables increases beyond a small threshold, say, five to 20; or they require a very large number of evaluations of the model functions to obtain a global optimum with reasonable certainty. If the function evaluations are expensive to compute one such optimization run may take days or weeks on typical tabletop computers. Genetic algorithms are typical derivative-free global algorithms requiring large amount of function evaluations. They are extremely appealing as they require no special mathematical knowledge

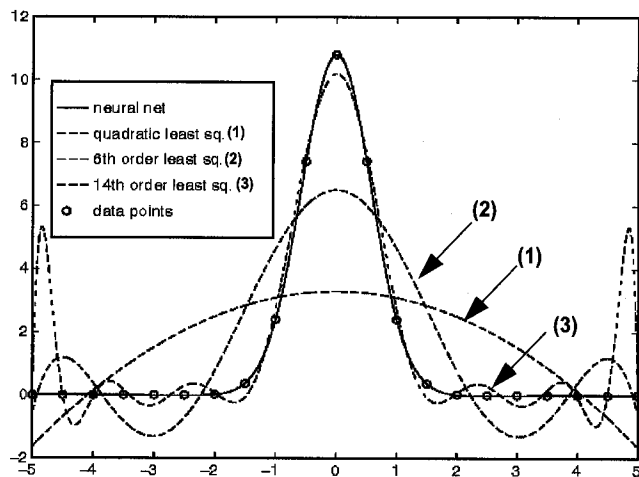


Fig. 2. A neural net model and three different-order polynomial models for the same data [16].

and can be easily programmed. Good implementations, however, require substantial skill and experience.

2.2. Surrogate models

A simulation model calculating an output for a given input can be viewed as a function. For many inputs and outputs we have a system such as:

$$y_1 = f_1(x_1, x_2, \dots, x_n) \quad (4)$$

$$y_2 = f_2(x_1, x_2, \dots, x_n)$$

...

$$y_m = f_m(x_1, x_2, \dots, x_n)$$

The functions f_1, f_2, \dots, f_m represent a complicated set of computations that has no explicit form. A good approach to deal with both noise and computational cost may be to use the simulation as a source of “computational experiments” that can supply us with the data points, just as if we had performed a physical experiment. Then curve-fitting or other data modeling techniques can be used based on these data points to derive new simpler functions that represent the original functions explicitly and with acceptable accuracy.

These new models are referred to as *surrogate models* or *metamodels*. Any subsequent use of the simulation inside the design optimization model is replaced by the surrogate models containing the explicit functions. Using surrogates may reduce drastically the computational load in a large design model that incorporates many analysis models. When a final design is reached, the original algorithms can be used to obtain more precise estimates.

Surrogate modeling methods include traditional polynomial curve-fitting techniques, as well as the more recently employed methods of neural networks and kriging. In Fig. 2, an example is shown for a set of given datapoints modeled by different order least-squares polynomials as well as a

neural net. In Fig. 3 a kriging metamodel for a multimodal function is shown. The metamodel captures the essential modes of the function but may not be very accurate, depending on how many points are used in the data sample. Regarding the use of surrogate models one must note the following.

1. Choosing the right underlying form of the model, e.g., the polynomial degree in a least-squares curve fitting, is subjective and requires judgment and/or trial and error.
2. Accuracy and fidelity increase as the number of sample points increase, but so does the computational cost; as the number of variables increases building surrogate models may itself become prohibitively expensive requiring tens of thousands of function evaluations.
3. A surrogate model is built based on a selected set of design variables; if this set changes a new surrogate model must be created, so the cost of generating a surrogate model must be considered in the context of the expected repeated use of the model generated.

The study of optimization algorithms that combine models of varying complexity is a promising research area, for example, the use of surrogates in trust region algorithms [11]. Traditional trust region algorithms create a sequence of subproblems that approximate the original one using Taylor series within a region of ‘trust’ where the approximation is considered valid. Instead of a Taylor series approximation a more general surrogate model can be used without jeopardizing convergence properties. A mix of Taylor series, exact functions and surrogate approximations can be combined in a subproblem, where the actual functions are kept if they are inexpensive while approximations are used for the expensive ones [13]. Although solving each subproblem may be more expensive the overall number of subproblems that must be generated seems to be substantially reduced.

A sequential metamodeling strategy can be also used [18]. A metamodel is fit to a data sample and the optimum design is obtained. A new sampling area is selected around this optimum and a new metamodel is fit, essentially ‘zooming in’ on the point. The process is repeated until convergence. The method generally fails to identify multiple optima efficiently and becomes cumbersome when the number of variables increases. An interesting optimization algorithm that uses metamodels is EGO for ‘efficient global optimization’ [19]. A kriging metamodel is fit to a data sample and a typically derivative-free optimization algorithm is used to find the optimum of an ‘expected improvement function’ (Fig. 4). This function balances the probability that one is close to the optimum with the probability that one has missed a significant portion of the design space (hence a high chance that one has missed something good). Sampling points are added at places where the expected improvement function has (local) maxima and the process is repeated. The method is efficient for small problems but may converge prematurely. A metamodel

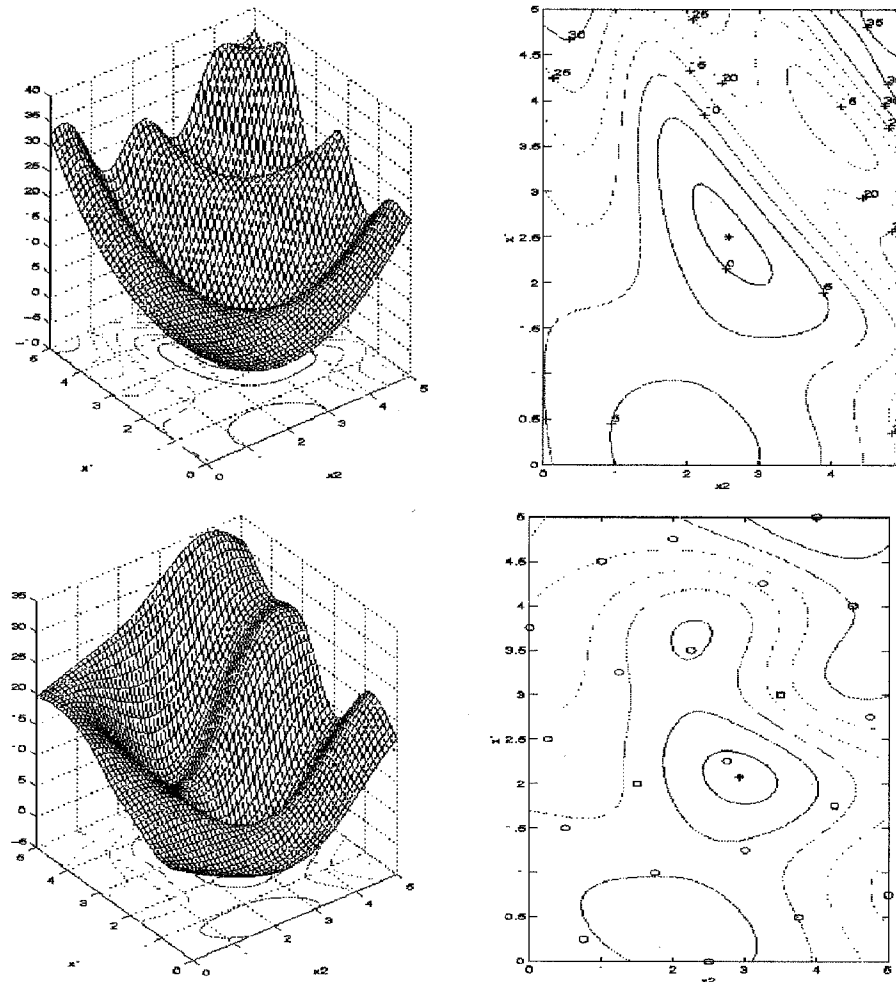


Fig. 3. Kriging model for a multimodal two-dimensional function; original function at the top and kriging metamodel at the bottom [18].

management framework (MMF) has been proposed recently [1,23], where an initial metamodel is used in conjunction with a pattern search method. The method is still not well developed, particularly for constrained problems.

3. Creativity: topologies and configurations

Conceptual design has always been associated with human creativity. Until relatively recently the generation of the initial morphology of a product has been beyond the realm of formal design methods. Artificial intelligence approaches during the 1980's showed promise but the domain of application remained one of highly simplified designs that would provide little excitement to a practicing designer.

3.1. Structural topologies

This situation changed dramatically in the later part of the 1980's in the field of topology design of structures. Until that time the problem had been primarily posed in the same

way as size or shape problems, namely, attempting to describe the design domain through a set of geometric parameters and finding their optimal values, possibly in an infinite domain (optimizing functionals using variational methods). The major breakthrough idea was a modeling one. Instead of looking for a parametric representation of geometry to describe topological alternatives, the problem was posed as a material distribution one: given an available design space and material, find how to distribute the material within the space so that boundary conditions of loads and supports are satisfied. Early work used sophisticated mathematical methods, such as homogenization [2], but other simplified methods, including genetic algorithms [3], were quickly invented to address at least the simpler problems.

As a result, topology optimization of structures today has become a mature field with commercial software available and used in industry. To illustrate the approach, consider the design of a bicycle frame [4]. The design problem may be posed as in Fig. 5. The results of topology optimization using a homogenization method are shown in Fig. 6(a).

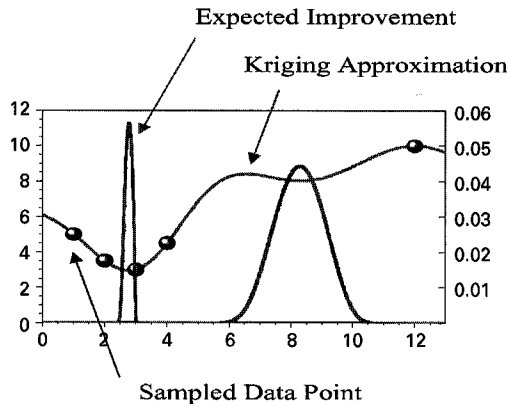


Fig. 4. An expected improvement function in EGO based on kriging.

The image is pixel based and must be further manipulated using image processing techniques such as filtering and smoothing, Fig. 6(b). The resulting frame can be further optimized by parameterizing it in the usual way and applying standard size-optimization techniques. The final frame is shown in Fig. 7.

These early results have been expanded today to highly complex problems involving three-dimensional geometries, material design, compliant mechanism design, and flexible structures. Applications span the automotive, aerospace, electronics and MEMS fields. See, for example Refs. [5,15,24].

3.2. Configuration design

Topology design for systems other than structures is often referred to as *configuration design*. Configuring non-structural systems, for example, mechanical systems involving flow and motion, remains a challenging problem. To illustrate how such problems may be formulated consider configuration design as a process of generating artifacts by assembling pre-defined components. This is a combinatorial problem often referred to as 'catalog selection'. Combinatorial explosion occurs quickly for relatively small catalogs.

One way to reduce the size of configuration problems is to abstract required components to higher levels of abstraction

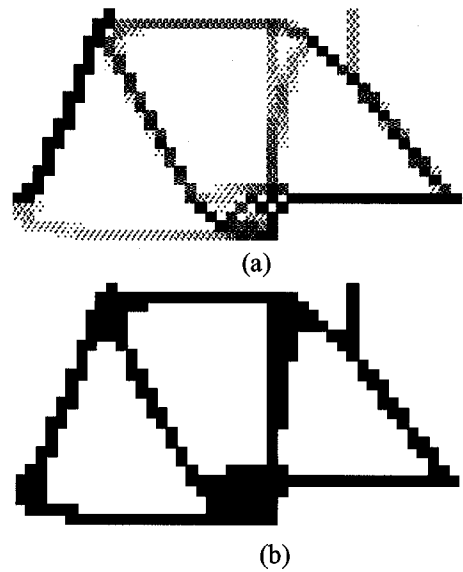


Fig. 6. (a) Topology optimization output; (b) filtered and smoothed image.

[22]. At higher abstraction levels, less important detail is temporarily ignored, and each component represents a family of lower-level components. Configuration is then performed at the highest level, explicitly enumerating all configurations at that level. Any complete configuration at the highest level is recursively instantiated to lower levels. At the same time, any incomplete configuration at the highest level is eliminated, thereby eliminating all possible lower-level instantiations of that configuration. In this manner, all configurations of components at the lowest level of abstraction are implicitly enumerated.

Configuration at any level is performed using a catalog of components or modules. Each module has an energy flow 'port' associated with it. The problem statement specifies certain input and output functions. The design task is to select other modules from the catalog that will properly connect all input and output ports.

As an example, consider the design of a hybrid vehicle powertrain having two or more sources of power, of which at least one can store and reuse energy. Components in the

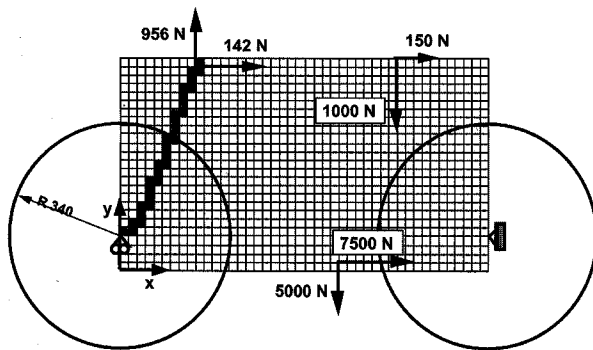


Fig. 5. Initial design model for the bicycle-frame problem.

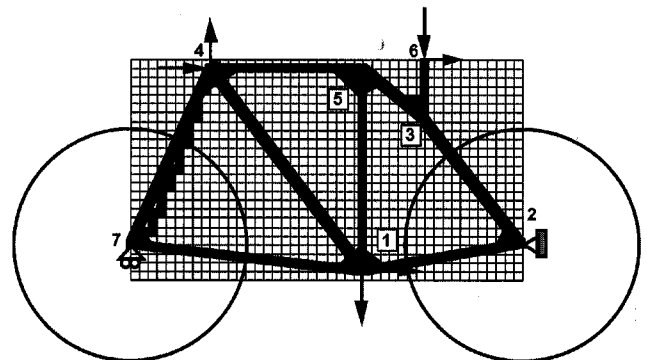


Fig. 7. Final design model for the bicycle frame problem.

Component Name	Port 1 Specifications	Port 2 Specifications	Port 3 Specifications
battery	(e, power, out)		
capacitor	(e, effort, out)		
reservoir	(h, power, in)		
accumulator	(h, power, out)		
engine	(mr, power, out)		
tors. spring	(mr, effort, out)		
road	(mt, power, in)		
pipe	(h, power, in)	(h, power, out)	
brake	(mr, effort, in)	(mr, effort, in)	
shaft	(mr, power, in)	(mr, power, out)	
elec. motor	(e, power, in)	(mr, power, out)	
generator	(mr, power, in)	(e, power, out)	
wheel	(mr, power, in)	(mt, power, out)	
rack & pin.	(mt, power, in)	(mr, power, out)	
planet. gears	(mr, power, in)	(mr, power, in)	(mr, power, out)
hyd. motor	(h, power, out)	(h, power, in)	(mr, power, out)
hyd. pump	(h, power, out)	(h, power, out)	(mr, power, in)

Fig. 8. Hybrid vehicle component list at level 3.

hybrid vehicle domain are represented at three levels of abstraction. At level 1, specifications represent the physical domain of a component port. Physical domains include electrical (e), mechanical rotational (mr), mechanical translational (mt), and hydraulic (h). At level 2, specifications represent the presence of power or effort at a port. At level 3, specifications represent the direction (in or out) of the power or effort at a port. Components in this domain include hydraulic devices such as pumps, motors, accumulators and reservoirs; electrical devices such as motors, generators and batteries; and mechanical devices such as engines, planetary gear sets and wheels. The least abstract level of the component database for this problem domain is shown in Fig. 8. All higher level components are abstracted from these level-3 components.

The configuration problem is to design a powertrain that supplies translational power to a road and includes an engine as a source of rotational power and a hydraulic accumulator as a source of hydraulic power. To model it, three components, engine (E), hydraulic accumulator (HA), and road (R), are required, Fig. 9. One of many feasible solutions generated automatically is shown in Fig. 10. A hydraulic motor (HM) transforms the hydraulic power of the accumulator into

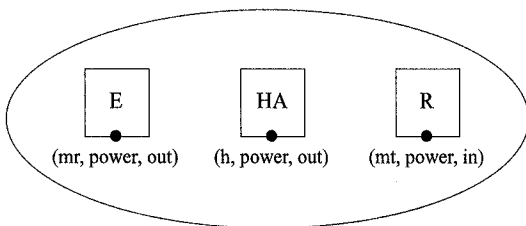


Fig. 9. Design ‘requirements’ for vehicle configuration problem.

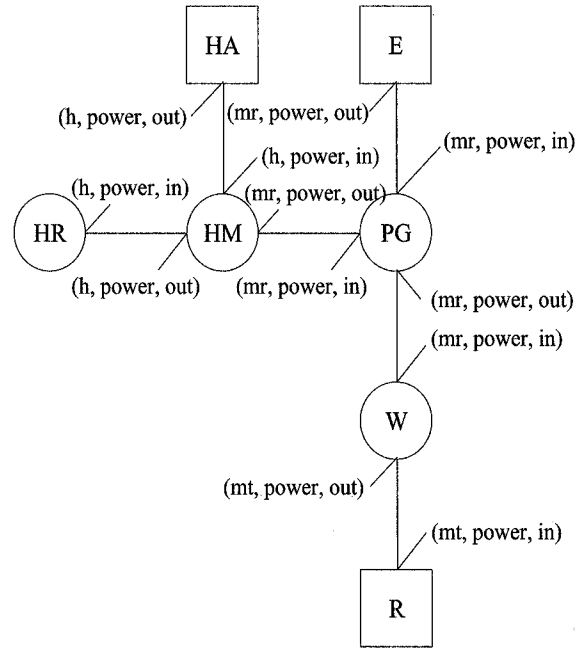


Fig. 10. A feasible topology for a hybrid powertrain.

mechanical rotational power, a planetary gear set (PG) combines power from the engine (E) and the hydraulic motor, and wheels (W) transform the rotational output of the planetary gears into translation on the road (R). For more details see Ref. [22]. The problem still suffers from combinatorial explosion. An optimization strategy can be used to sift through feasible designs and increase the method’s efficiency.

Configuration design remains a highly challenging problem for mechanical systems. Most current efforts use some sort of modular design. One potential approach may involve the generation of appropriate design ‘grammars’ similar to those that have been successfully used in architectural and structural design [20]. Finding the proper grammatical elements and syntactical rules may crack this problem, just as the material distribution formulation solved the structural topology problem in the late 1980’s.

4. Complexity: from components to systems

Producers of engineered artifacts are generally considered highly competent in system component design. In designing large complex artifacts, such as air or ground vehicles, the challenge today is the proper design of each component and subsystem so that the overall system is optimized. The equivalent problem in mathematical optimization would be to create a large optimal design model that contains all components and subsystems and to solve the entire problem ‘all at once’. Frequently this approach may not be possible, as we will see below.

4.1. Partitioning and coordination

When the models are nonlinear and the problem size

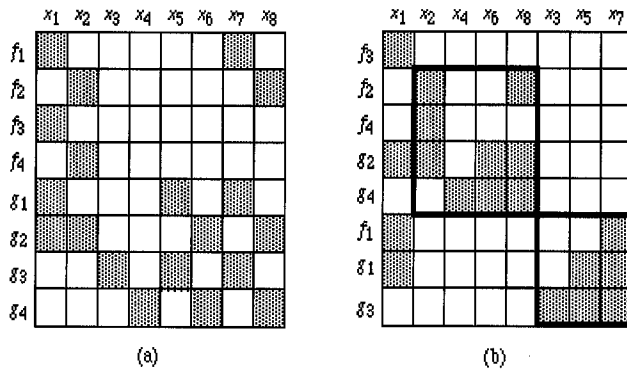


Fig. 11. Original form of an FDT and derived form after reordering rows and columns to identify subproblems.

becomes large one cannot expect reliable results from numerical optimization algorithms. Even when numerical results are successfully obtained one may not be able to interpret the engineering trade-offs and to use intuition to confirm the computed numbers. The obvious alternative is to employ some form of problem decomposition. Breaking up the optimization model into smaller submodels must be done with rigorous decomposition and coordination strategies.

In principle, a good decomposition should allow:

- improved coordination and communication;
- conceptual simplification of the design;
- modularity and parallel computation;
- simpler and more efficient computational procedures, and
- use of optimization techniques tailored to the specific subproblem.

Decomposition strategies are commonly classified as *object* (by physical components), *aspect* (by knowledge domains), *sequential* (by directed flow of elements or information), and *model-based*. Object and aspect decomposition assume a ‘natural’ decomposition of the problem. However, drawing ‘boundaries’ around physical components and subassemblies is subjective, while division by specialties (knowledge domains) may be dictated by management considerations that fail to account for disciplinary coupling. Sequential decomposition presumes unidirectionality of design information flow that may contradict the cooperative behavior desirable in concurrent engineering. Model-based decomposition uses the mathematical functional representation of objectives and constraints in the model to identify unconnected or weakly-connected structures implicit in the mathematical design model, and is limited to the design decisions included in the model.

Function dependence on variables may be represented by a Boolean matrix termed the *functional dependence table* (FDT). Rows are labeled with relation/function names and columns are labeled with variable names. The entry in the i th row and j th column is ‘true’ if the i th function depends on the j th variable; otherwise, it is ‘false’. A typical FDT is shown in Fig. 11(a), the shaded boxes indicating a ‘true’

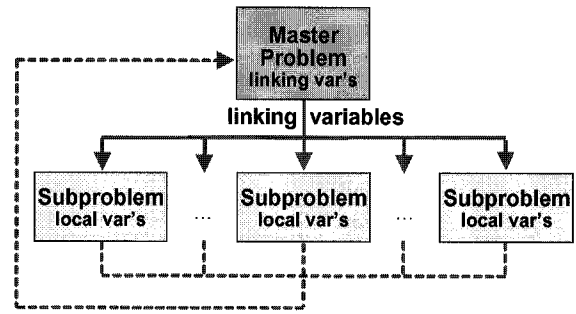


Fig. 12. Coordination in solving hierarchical decomposed system.

Boolean value. Fig. 11(b) shows the FDT for the same problem after x_1 has been selected as the linking variable, and rows and columns have been reordered to reveal two partitions of the problem: subproblem 1 with functions $\{f_2, f_4, g_2, g_4\}$ and local variables $\{x_2, x_4, x_6, x_8\}$, and subproblem 2 with functions $\{f_1, g_1, g_3\}$ and local variables $\{x_3, x_5, x_7\}$. This approach makes use of recent advances common to such diverse areas as graph theory, VLSI design, computational mechanics, parallel computing, and organizational management. The resulting partitioned problem can be solved using a coordination strategy, as depicted schematically in Fig. 12.

The decomposition above can be made rigorous by modeling the decomposition problem as a network optimization problem [8]. Mathematical relations are modeled as processing units of a communication network and design and state variables are communication links between these units. The optimal decomposition problem is then formulated as one of finding the communication links whose failure would reduce network reliability the most. Extending this form of the problem to include other graph metrics, such as keeping subsystem sizes balanced and number of linking variables small, has led to partitioning methods that use hypergraph or integer programming formulations [6,9].

Coordination strategies can be posed in an ad hoc manner. However, the key requirement is that they converge to the same solution set as that of the undecomposed problem. This convergence requirement poses serious theoretical challenges. Most rigorous coordination strategies developed by operations researchers make very strong assumptions, such as linearity or convexity of all models. Strategies developed in the design engineering community usually cannot demonstrate rigorous convergence properties. A compromise approach has been introduced where existing nonlinear programming algorithms are modified to take advantage of problem structure without losing basic convergence properties [12].

4.2. Design target cascading

In the *target cascading* concept, product design is viewed as a four-step process: (i) specify overall product targets; (ii) propagate product targets to system, subsystem and component ‘sub-targets’; (iii) design system, subsystems and components to achieve their respective sub-targets and

$$\begin{array}{l}
 P_{ij}^* : \quad \text{Minimize } x_{ij}, y_{i+1}, \tilde{R}_{i+1}, \varepsilon \quad \|\mathbf{R}_{ij} - \mathbf{R}_{ij}^*\| + \|y_{ij} - y_{ij}^*\| + \varepsilon \\
 \text{where } \quad \mathbf{R}_{ij} = r_{ij}(\tilde{R}_{i+1}, x_{ij}) \quad \text{subject to} \\
 \|\mathbf{R}_{i+1} - \mathbf{R}_{i+1}^\dagger\| \leq \varepsilon \quad \|y_{i+1} - y_{i+1}^\dagger\| \leq \varepsilon \\
 g_{ij}(\tilde{R}_{i+1}, x_{ij}) \leq 0 \quad h_{ij}(\tilde{R}_{i+1}, x_{ij}) = 0
 \end{array}$$

Fig. 13. Formulation of target matching and propagation problem level i and element j .

(iv) verify that the resulting product meets overall product targets. The goal of the target cascading process is that systems, subsystems and components operate together in a compatible and consistent manner. A mathematically rigorous decision-making methodology that takes advantage of hierarchical system partitioning and coordination can lead to reduction in product design-cycle time, avoidance of design iterations late in the development process, and increased likelihood that physical prototypes will be closer to production quality. Target setting also allows outsourcing the design of a particular subsystem or component after providing the supplier with the corresponding set of targets.

The formal process assumes existence of analytical models for systems, subsystems, and components. Computationally inexpensive models are essential for target cascading, given the large number of components and the number and complexity of interactions. Hence, high-fidelity, expensive models need to be replaced by surrogate models.

In the formulation of the optimal design problem an index i corresponds to the level (e.g., vertical decomposition), whereas index j correspond to the element within the level (e.g., horizontal decomposition). Note that design variables change during a specific optimization problem, while parameters remain constant for that problem. Responses are design functions computed as ‘outputs’ of a particular problem solution. The following terminology is used to create a formal representation.

- x_{ij} : design variables unique to element j at level i (i.e., local variables).
- y_{ij} : design (linking) variables for element j at level i common to another element at level i
- x_{ij} : design variables for element j at level i , namely, $x_{ij} = (\tilde{x}_{ij}, y_{ij})$
- y_i : design linking variables for all elements at level i
- x_i : design variables for all elements of level i , i.e., $x_i, x_{i2}, x_{i3}, \dots, x_{in}$
- R_{ij} : response vector for element j at level i
- R_i : response vector for all elements at level i , i.e., $R_i = (R_{i1}, R_{i2}, R_{i3}, \dots)$. The response vector for element j at level i depends on its design variables x_{ij} and on level $-(i + 1), -(i + 2)$, responses:

$$R_{ij} = r_{ij}(R_{i+1}, \dots, x_{ij}) \quad (5)$$

Response dependencies are given a tree-like structure by augmenting response vectors with lower level responses of

descendent elements that are directly used at upper levels. In general:

$$\tilde{R}_{ij} \equiv (R_{ij} \cup R_{in}) \quad (6)$$

where element n at level 1 is a descendent of element j at level i whose response directly influences the response of an element at a level above level i , so:

$$\tilde{R}_i \equiv (\tilde{R}_{i1}, \tilde{R}_{i2}, \tilde{R}_{i3}, \dots), \tilde{R}_{ij} \equiv r_{ij}(\tilde{R}_{i+1}, x_{ij}). \quad (7)$$

The family of subproblems is shown in Fig. 13. In the solution of subproblem P_{ij}^* at level i , R_{ij}^* and y_{ij}^* are determined at the upper level $(i - 1)$ by solving subproblems $P_{(i-1)j}^*$. Similarly, R_{i+1}^\dagger and y_{i+1}^\dagger are determined at the lower level $(i + 1)$ by solving subproblems $P_{(i+1)j}^*$. Superscript asterisks relate to quantities generated at the upper level and superscript daggers to the lower one. The notation is cumbersome and an example would elucidate matters but is omitted here for lack of space. For an example and more details see Ref. [10].

The process of information exchange for an automotive vehicle is shown in Fig. 14. Vertical coupling (by responses) and horizontal coupling (by linking variables) enables partitioning the problem into temporarily independent subproblems by fixing responses and linking parameters at goal values. Overall coordination is achieved by minimizing the errors of the actual responses and linking variables from their goals. Targets for responses and linking parameters are updated, and the process is repeated until convergence. System, subsystem, and component targets are retrieved from the solution of subproblems P_{ij}^* . Implementing such computer-based product development processes in industry is an on-going challenge.

5. Smart products: designing controlled artifacts

‘Smart’ artifacts are designed to function in a changing environment and to adapt to these environmental changes. In creating such artifacts we must create the embodiment of the design and also add the ‘smarts’, i.e., design a control system and associated sensors. In most product design processes we tend to first ‘create the design’ and then ‘design the control’. Actually there are three individual tasks: design the artifact (create its topology and embodiment), select a control strategy (identify a theoretical control function), and design the physical controller that will perform this function (create its topology and embodiment).

A sequential design approach is often natural, as each task may be itself difficult and coupling them together could make the design task intractable. Problems with nonlinearities in the control tasks have particular difficulties. Decision criteria for each design task are often different and competing. Nevertheless, the initial design of the artifact affects its control characteristics, so if adaptive performance is important and/or expensive, control-related design criteria must

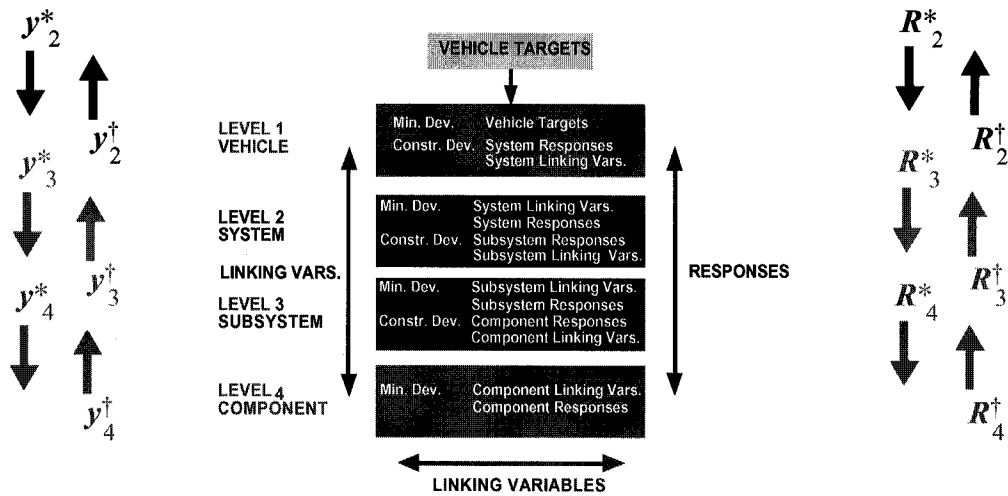


Fig. 14. Target cascading process in the design of an automotive product.

be incorporated early in the design process. As products become increasingly smart, examining the tight coupling between embodiment design and control design becomes a critical step in the design process.

In a formal statement, the *optimal design problem* (ODP) is defined as:

minimize $f(d; c)$

$$d \in D \subseteq \mathcal{R}^d$$

Table 1
Nomenclature for optimal design and control

<i>Design problem formulation</i>	
<i>a</i>	Simple design parameters
<i>b</i>	Design parameters, based upon the control
<i>c</i>	Parameters of the design problem = { <i>a</i> , <i>b</i> }
<i>d</i>	Design variables
<i>f</i>	Objective for the design problem
<i>h</i>	Equality constraints for the design problem
<i>g</i>	Inequality constraints for the design problem
<i>q</i>	Number of design variables
<i>v</i>	Control parameters, based upon the design
<i>Control problem formulation</i>	
<i>b</i>	Design parameters, based upon the control
<i>J</i>	Performance index, objective for the control problem
<i>j</i>	Time dependent performance index
<i>k</i>	Equality constraints for the control problem
<i>l</i>	Inequality constraints for the control problem
<i>m</i>	Number of state variables
<i>n</i>	Number of control variables
<i>p</i>	Control gains in controller configuration
<i>r</i>	Dynamical equations for the control problem
<i>s</i>	Response relations for the control problem
<i>t</i>	Time continuum
<i>u</i>	Simple control parameters
<i>v</i>	Control parameters, based upon the design
<i>w</i>	Parameters of the control problem = { <i>u</i> , <i>v</i> }
<i>x(t)</i>	State variables as a function of time
<i>y(t)</i>	Measurable response of the controlled system
<i>z(t)</i>	Control variables as a function of time

subject to:

$$h(d; c) = 0 \tag{8}$$

$$g(d; c) \leq 0$$

$$c = \{a, b\}$$

$$v = v(d^*, c) \text{ results}$$

and the *optical control problem* (OCP) is defined as minimize

$$J = \int_0^{t_f} j(x(t), z(t), t; w) dt$$

$$t \in [0, t_f], x(t) \in \mathcal{R}^n, z(t) \in \mathcal{R}^m$$

subject to:

$$\dot{x} = r(x(t), z(t), t; w) \tag{9}$$

$$y = s(x(t), z(t), t; w)$$

$$k(x(t), z(t), t; w) = 0$$

$$l(x(t), z(t), t; w) \leq 0$$

$$w = \{u, v\}$$

$$b = b(x(t), z^*(t)(t; w) \text{ results}$$

using the nomenclature given in Table 1. In the problem functions, the quantities to the left of the semicolon are variables for the particular problem, while those to the right of the semicolon are parameters.

Directly applying the function $z^*(t)$ does not yield the desired response due to model deficiencies and disturbances in the real system. As a remedy, a feedback control strategy is used where $z(t)$ is a function of the measurable response $y(t)$. One must recognize then that determining $z^*(t)$ as a mathematical entity does not solve the control problem

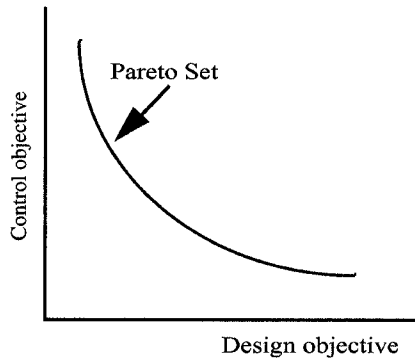


Fig. 15. Pareto optimal solutions for the combined optimal control and optimal design problem.

completely, since the actual controller that produces $z^*(t)$ must still be designed. Designing a controller is actually a *configuration design* problem, namely, different configurations or strategies—each with its own design variables—must be generated and compared. This process is generally very difficult to perform under a single mathematical decision-making model. The control designer relies on experience and experimentation to select the proper controller configuration for the problem at hand. With a priori determination of the controller configuration, an optimization problem may be formulated to choose the gains p that relate z and y directly. This new problem is the *optimal gain problem* (OGP). Pre-selecting the controller configuration transforms the optimization problem to one that is time independent,

minimize

$$p \in \mathcal{P}^M J = \int_0^t j(x(t), z(t), t; w) dt$$

subject to:

$$\dot{x} = r(x(t), z(t), t; w)$$

$$y = s(x(t), z(t), t; w)$$

$$z(t) = z(p, y(t))$$

$$k(x(t), z(t), t; w) = 0$$

$$l(x(t), z(t), t; w) \leq 0$$

$$w = \{u, v\}$$

$$b = b(x(t), z^*(t), t; w) \text{ results.}$$

The results from the OGP do not necessarily lead to an optimal controller as defined by the OCP. However, one expects the OGP to lead to a controller design that can be easily implemented physically.

The procedure to control an artifact ‘optimally’ may then be divided into the following tasks:

1. Solve the OCP for an optimal control function $z^*(t)$.

2. Select a configuration of the controller that can attempt to achieve $z^*(t)$.
3. For the chosen controller configuration, solve the OGP for p^* . Keep in mind that $z = z(p^*, y(t))$ may or may not be equal to $z^*(t)$.
4. If z is not acceptably close to z^* , return to step 2 and select another configuration.
5. If z is close enough to z^* , build the controller with z and apply to the physical system.

The combined optimal design and optimal gains problem can now be solved as a multiobjective optimization problem and the Pareto set of solutions can be generated, as in Fig. 15. It is interesting to note that the traditional sequential methods of solving the combined problem do not result in Pareto solution. The problem is examined in more detail and applied to the design and control of DC motors in Ref. [17].

The combined design and control problem becomes more challenging as the system complexity increases. The multiobjective formulation prompts one to think of the problem as a decomposition. Then solving the ‘all at once’ problem in decomposed form requires a coordination strategy that will guarantee generation of all Pareto solutions for the original problem. Moreover, both the design and the control problem may be themselves multiobjective in nature, and dealing with the resulting complexity poses obvious challenges.

6. Enterprise-wide designs

Traditional design processes typically address design of a single product. However, manufacturing firms increasingly make several variations of a product, each aimed at a different market niche. In a product line, products are likely to be related in some way. If it is possible to manufacture two different products using some of the same parts, a company often sees benefits through nonproliferation of parts, and reductions of inventory, design lead-in time, design efforts for new market niches, and number and type of factory machinery and processes. A *product platform* is a set of common components, modules or parts from which a stream of derivative products can be efficiently created and launched. Designing platforms as opposed to individual products is an example of an enterprise-wide design process.

Formal optimization can be used in designing several individual products for market niches [14]. If two or more products are to share the same part, the *performance* of each product within its own niche will change in comparison to its performance if it were designed to not share any parts. Optimization formulations can quantify this change in performance, one of the criteria used to justify or deny the use of the common part. Furthermore, if the common part is to be used in two or more products, the design objectives of the two products will often be in competition—leading to the use of multicriteria optimization.

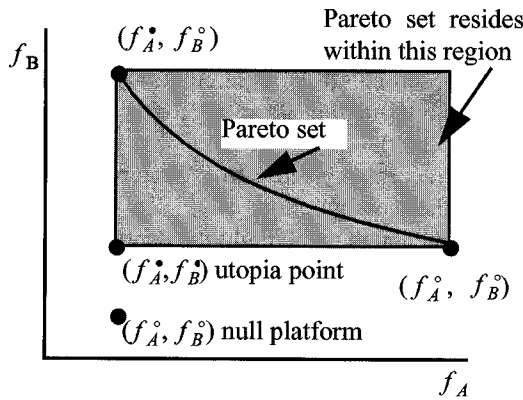


Fig. 16. Pareto set for a platform with two products A and B sharing a component.

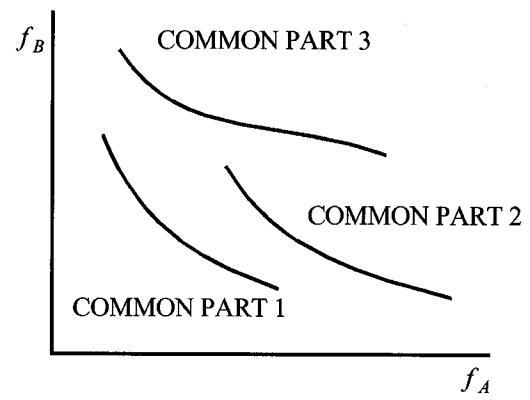


Fig. 17. Pareto sets for several potential platforms. Each Pareto set represents a different set of shared parts.

When making comparisons between products one must compare the best possible designs when sharing parts to the best possible designs when the products are not sharing parts. In the present discussion a product platform is a specific set of shared parts among the different products. A single multicriteria optimization problem can capture the decision required in evaluating the entire platform. Assuming that a criterion i depends only on variables x_i , we pose the problem:

$$\begin{aligned}
 &\text{minimize } f_i(x_i) \quad i = 1, \dots, p \\
 &\text{subject to:} \\
 &g_i(x_i) \leq 0 \quad i = 1, \dots, p \\
 &h_i(x_i) = 0 \quad i = 1, \dots, p \\
 &x_{i,k_1} = x_{j,k_2} \quad (k_1, k_2) \in P_i \\
 &\quad \quad \quad i, j = 1, \dots, p \\
 &\quad \quad \quad i < j
 \end{aligned}
 \tag{11}$$

P_{ij} is a set of index pairs used to represent a set of equality constraints. If products i and j share some of the same parts, then P_{ij} contains the index pairs of the design variables describing the common parts, effectively forcing the part to be the same for both products i and j . Thus a platform configuration is defined by a distinct set of index pairs. To compare two different product platforms, the solutions to Eq. (11) are compared with different sets of index pairs, say $\{P_{ij}\}$ and $\{Q_{ij}\}$. For a specific set of P_{ij} the solutions of Eq. (8) are a Pareto set. As a convention, different superscripts will represent optimal values from different platform configurations. For two products A and B in a platform, a superscript *circle* represents optimal quantities for the null platform (f_A° and f_B°). A superscript *bullet* (f_A^\bullet and f_B^\bullet) represents optimal quantities for the platform with the common parts. A more involved notation must be used where more combinations of shared parts are used.

The individual minima f_i° of Eq. (11) are defined as the

extreme values of the Pareto set. These are the solutions to Eq. (11) with only one of the scalar functions (f_A or f_B) used as an objective. Note that the entire Pareto set for the configuration lies in a rectangle with corners at (f_A^\bullet, f_B^\bullet) and (f_A°, f_B°), see Fig. 16. Not surprisingly, the utopia point (f_A^\bullet, f_B^\bullet), is not as good as the point representing separate designs (f_A°, f_B°). The additional equality constraints of the product platform imply that the feasible space is smaller. This leads to a general statement about product platforms: If there are two sets of index pairs $\{R_{ij}\}$ and $\{S_{ij}\}$ such that $R_{ij} \subseteq S_{ij}$ for each i and j , then the feasible space for platform S cannot be larger than the feasible space for platform R . Therefore, the performance for platform R will be at least that of platform S as measured by the objective function values.

In fact, the solution to the separate optimal design problems will often be *better* than the utopia point, and the designer of a product platform should *expect* to give up some acceptable amount of performance. This point is important because by defining the individual minima, the limits of the Pareto set and therefore the changes in performance are known. Simply quantifying the change in performance is useful to justify further investigation. If there is too much degradation in performance, (i.e., $f_A^\bullet \ll f_A^\circ$ or $f_B^\bullet \ll f_B^\circ$) the platform can be disregarded and there is no need to find other points within the Pareto set.

A general design strategy would proceed as follows. First identify parts that could be shared between two or more products. For each possible pair of products, assign a set of index pairs, i.e., $\{O_{ij}\}, \{P_{ij}\}, \{Q_{ij}\}, \dots, \{Y_{ij}\}, \{Z_{ij}\}$. Formulate the multicriteria problem Eq. (11), and determine the individual optima (i.e., extreme points) for each possible configuration. Use the individual optima for each configuration to decide if it is worthwhile to investigate the Pareto set of that configuration. For each combination of common components that has an acceptable degradation in performance, calculate the Pareto set. Finally, choose the design that offers the best value for all appropriate products while still allowing for the benefits of having a flexible product platform. Fig. 17 shows a plot of Pareto sets associated with

different potential platforms. The optimal platform design should lie in one of these Pareto sets, but exactly which Pareto set is the best is a question of performance as well as other business issues.

7. Concluding remarks

The preceding sections provided a glimpse into several highly diverse design problems where a decision-making model was used to describe the design situation. Although the challenges in the mathematics of optimization remain daunting, the greater challenge to engineering creativity is to exploit the richness of the decision-making paradigm and model design problems in new and creative ways. The two areas where this challenge is more pointed are the conceptual design of mechanical artifacts and the design of complex, multicomponent artifacts. In the latter case, the demand for modeling “whole-life” and business requirements and incorporating them in the more traditional performance-based design will only become stronger. Formal decision models can serve us well in helping us make more rational and more informed design decisions.

Acknowledgements

This review article was composed based on joint work with a number of collaborators at the University of Michigan, including M. Chirehdast, R. Fellini, N. Kikuchi, H.M. Kim, R.P. Krishnamachari, S. Nelson, M. Parkinson, M. Sasena, G. Snively, J. Reyer, and especially Nestor Michelena. A manuscript review by J. Michalek is also appreciated. The work described has been supported by a variety of sponsors, notably, the Automotive Research Center under the auspices of the US Army TACOM, Ford Motor Co, General Motors Corp., the U.S. National Science Foundation, and the US Office of Naval Research. The author gratefully acknowledges the support of the sponsors and the contributions of his colleagues and students.

References

- [1] Alexandrov N, Dennis Jr JE, Lewis RM, Torczon V. A trust region framework for managing the use of approximate models in optimization. *Structural Optimization* 1998;15(1):16–23.
- [2] Bendsoe M, Kikuchi N. Generating optimal topologies in structural design using a homogenization method. *Computer Methods in Applied Mechanics and Engineering* 1988;71:197–224.
- [3] Chapman C, Saitou K, Jakiela M. Genetic algorithms as an approach to configuration and topology design. *ASME Journal of Mechanical Design* 1994;116:1005–12.
- [4] Chirehdast M, Gea HC, Kikuchi N, Papalambros YP. Structural configuration examples of an integrated optimal design process. *ASME Journal of Mechanical Design* 1994;116(4):997–1004.
- [5] Frecker MI. Optimal design of compliant mechanisms. Doctoral dissertation, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 1997.
- [6] Krishnamachari R, Papalambros PY. Optimal hierarchical decomposition synthesis using integer programming. *ASME Journal of Mechanical Design* 1997;119(4):440–7.
- [7] Medland AJ. *The computer-based design process*. New York: Springer-Verlag, 1986.
- [8] Michelena N, Papalambros PY. A network reliability approach to optimal decomposition of design problems. *ASME Journal of Mechanical Design* 1995;117(3):433–40.
- [9] Michelena N, Papalambros PY. A hypergraph framework to optimal model-based decomposition of design problems. *Computational Optimization and Applications* 1997;8(2):173–96.
- [10] Michelena N, Kim HM, Papalambros PY. A system partitioning and optimization approach to target cascading. In: Lindemann U, editor. *Proceedings of the 12th Int. Conf. on Engineering Design*, 2. 1999. p. 1109–12.
- [11] Nelson SA. Optimal design of hierarchical systems using sequential decomposition programming. Doctoral dissertation, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 1997.
- [12] Nelson SA, Papalambros PY. A modified trust region algorithm for hierarchical NLP. *Structural Optimization* 1998;16:19–28.
- [13] Nelson SA, Papalambros PY. Exploiting discrepancies in function computation time within optimization models, *ASME Design Automation Conference*, Atlanta 1998. Also to appear as ‘The use of trust region algorithms to exploiting discrepancies in function computation time within optimization models’, *ASME Journal of Mechanical Design*.
- [14] Nelson SA, Parkinson MB, Papalambros PY. Multicriteria optimization in product platform design, *ASME Design Automation Conference*, Las Vegas, 1999.
- [15] Nishiwaki S. Optimum Structural topology design considering flexibility. Doctoral dissertation, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 1998.
- [16] Papalambros PY, Wilde DJ. *Principles of optimal design: modeling and computation*. 1st and 2nd ed., Cambridge University Press, New York, 1988, 2000.
- [17] Reyer JA, Papalambros PY. Optimal design and control of an electric DC motor, *ASME Design Automation Conference*, Las Vegas, 1999.
- [18] Sasena MJ. Optimization of computer simulations via smoothing splines and kriging metamodels. MSc thesis, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 1998.
- [19] Schonlau M, Welch WJ, Jones DR. Global versus local search in constrained optimization of computer models, *Institute for Improvement in Quality and Productivity*, University of Waterloo, Waterloo, Ontario, Canada, paper RR-97-11, 1997.
- [20] Shea K, Cagan J. Innovative dome design: applying geodesic patterns with shape annealing. *Artif Intel for Engin Des, Anal and Manuf* 1997;11:379–94.
- [22] Snively GL, Papalambros PY. Abstraction as a configuration design methodology, *Advances in design automation*. New York: ASME, 1993.
- [23] Torczon V, Trosset MW. Using approximation to accelerate engineering design optimization, *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis (AIAA Paper 98-4800), 1998.
- [24] Yoo J. Structural optimization in magnetic fields using the homogenization design method. Doctoral dissertation, Department of Mechanical Engineering, University of Michigan, Ann Arbor, 1999.