Manufacturing Review vol 3, no 1, March 1990, pp 49-59.

Least Cost Tolerance Allocation for Mechanical Assemblies with Automated Process Selection

Kenneth W. Chase
Mechanical Engineering Department
Brigham Young University
Provo, Utah 84602

William H. Greenwood
Sandia National Laboratories
Albuquerque, NM 87185

Bruce G. Loosli
Boeing Aerospace Co.
Seattle, WA 98124

Loren F. Hauglund
International Technigroup Inc.
Milford, OH 45150

ABSTRACT

The allocation of tolerances among the components of a mechanical assembly can significantly affect the resulting manufacturing costs. If cost versus tolerance data are available for each dimension, the least cost tolerance allocation may be determined by optimization techniques. However, when alternate manufacturing processes are available for some of the components, a discrete optimization problem results. An exhaustive search of all possible combinations of processes will determine the global optimum, but the number of combinations increases geometrically, becoming very large for assemblies of only moderate complexity.

Several methods were tested for systematically searching for the minimum cost process set for an assembly. Both discrete and continuous optimization schemes were compared to an exhaustive search, based on CPU times and the number of combinations required to find the global optimum.

A new method is presented which is several orders of magnitude better than the exhaustive search.

1 INTRODUCTION

The specification of tolerances on the dimensions of manufactured parts has a significant impact on final production cost. Tight tolerances can result in excessive process costs, while loose tolerances may lead to increased waste and assembly problems. Improper tolerance specification may also result in inferior product performance and loss of market share. The modest investments required for a thorough tolerance analysis can yield substantial benefits.

This paper presents a procedure for tolerance specification based on quantitative estimates of the cost of tolerances, which permits the selection of component tolerances in mechanical assemblies for minimum cost of production. It also presents three new methods for automatically selecting the most economical manufacturing process for each part dimension from a set of alternative processes. Two of the new cost minimization procedures use Lagrange multipliers to determine the optimum tolerances combined with discrete search techniques to find the optimum processes from a set of alternatives (Exhaustive Search and Univariate Search). Another new procedure uses nonlinear programming techniques (SQP). These new methods are compared for efficiency with existing methods (Zero-One, Branch-and-Bound).

Tolerance Allocation

Critical tolerances in mechanical devices are generally the result of tolerance stack-up, or tolerance accumulation in assemblies of parts. The variation in the resultant clearances, interference fits, lubrication paths, end play, etc. depends on the variations in each of the component parts in the assembly. The assembly tolerance is generally specified based on performance requirements, while the component tolerances are closely related to the capabilities of the production processes. The most common tolerance specification problem encountered by engineering designers is tolerance allocation, which is the distribution of the specified assembly tolerance among the components of the assembly. This design procedure is illustrated in **Fig. 1**.

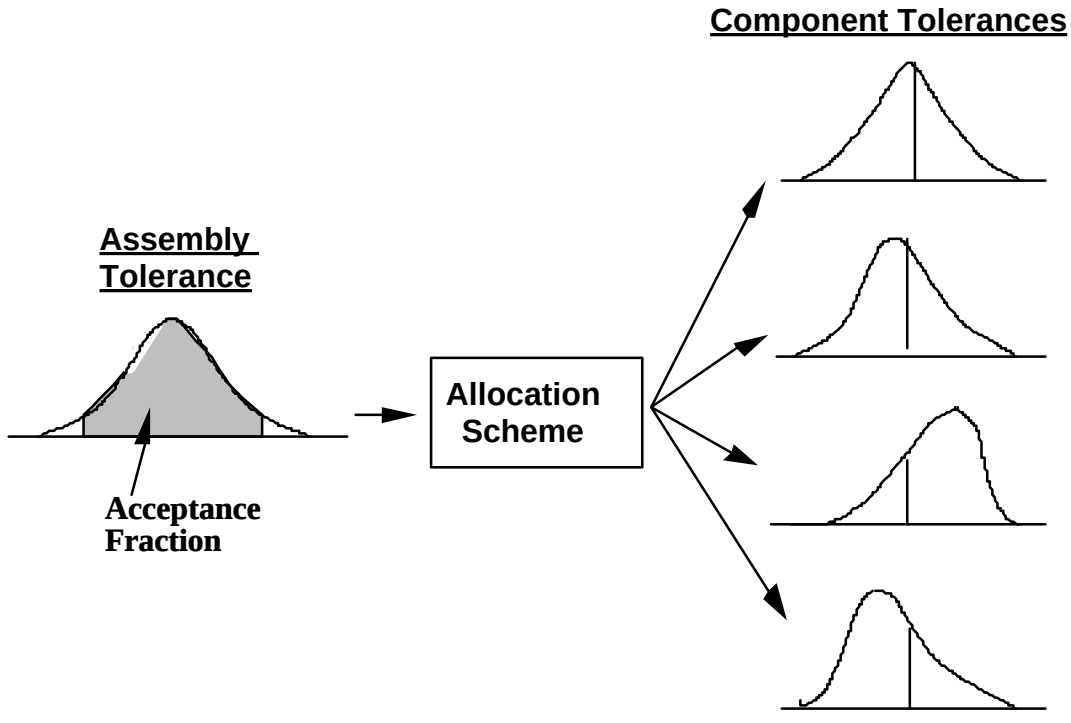


Figure 1. Tolerance specification by tolerance allocation.

The component tolerances could be distributed equally among all of the parts in an assembly. However, each component tolerance may have a different manufacturing cost associated with it due to part complexity or process differences. By defining a cost-vs-tolerance function for each component dimension, the component tolerances may be allocated to minimize cost of production. A typical cost-tolerance function is shown in **Figure 2**. It is basically a reciprocal function which estimates the decrease in cost for an increase in tolerance.

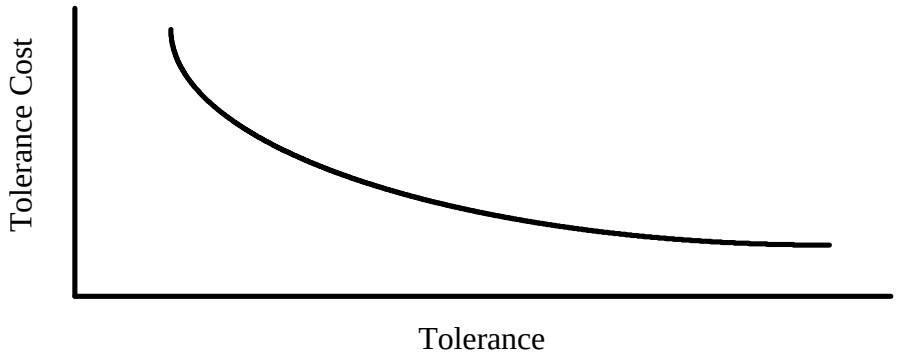


Figure 2. Typical cost-vs-tolerance function.

Minimum-Cost Tolerance Allocation

A substantial amount of research has been carried out regarding optimal tolerance allocation using cost-vs-tolerance functions. Various functions have been proposed to describe the cost-tolerance relationship and various optimization methods have been applied. These are summarized in **Table 1**.

Table 1. Proposed Cost-vs-Tolerance Models

	Cost Model	Method	Author
Linear	$\mathbf{A - B\,T}$	Linear prog	Edel and Auer[4]
Reciprocal	$\mathbf{A + B/T}$	Lagrange mult	Chase and Greenwood[3]
Reciprocal Squared	$\mathbf{A + B/T^2}$	Lagrange mult	Spotts[18]
Reciprocal Power	$\mathbf{A + B/T^k}$	Lagrange mult	Sutherland and Roth[19]
	$\mathbf{B/T^{k_i}}$	Nonlin prog	Lee and Woo[6]
Exponential	$\mathbf{B\,e^{-mT}}$	Lagrange mult	Speckhart[17]
		Geom prog	Wilde and Prentice[20]
		Graphical	Peters[16]
Expon/Recip Power	$\mathbf{B\,e^{-mT}/T^k}$	Nonlin prog	Michael and Siddall[9,10]
Piecewise Linear	$\mathbf{A_i - B_i\,T_i}$	Linear prog	Bjork[2], Patel[15]
Empirical Data	**Discrete points**	Zero-one prog	Ostwald and Huang[12]
		Combinatorial	Monte and Datseris[11]
		Branch & Bound	Lee and Woo[7]

The constant coefficient **A** represents the fixed costs, such as tooling, setup, prior operations, etc. The **B** term represents the cost of producing a single component dimension to a specified tolerance **T**. All costs are calculated on a per part basis. A comprehensive comparison of most of the above models and optimization techniques is found in Reference 21.

Extending Tolerance Allocation by Lagrange Multipliers

Assuming a cost-tolerance function for each component of the form $\mathbf{C=A_i+B_i/T^{k_i}}$, where coefficients $\mathbf{A_i}$, $\mathbf{B_i}$ and the exponent $\mathbf{k_i}$ are unique for each component, results in a general model which can describe a wide range of processes. This allocation problem, with variable exponents has been solved by Lee and Woo[6] using nonlinear programming.

The method of Lagrange Multipliers is a closed-form solution for the least-cost component tolerances. This method can be extended to include the case of variable exponents. An iterative solution is required, due to the resulting mixed exponents, but it is a simple one-dimensional solution.

The derivation begins by combining the cost minimization function and the assembly constraint into an augmented system of equations to be minimized by setting the derivatives to zero:

$$\frac{d}{dT_i}(\text{Cost function}) + \lambda \frac{d}{dT_i}(\text{Constraint}) = 0 \quad (i = 1, \ldots n)$$

where λ is the Lagrange Multiplier.

Substituting for the cost function and assuming the assembly tolerance constraint is the statistical sum of component tolerances:

$$\frac{d}{dT_i}(\Sigma(A_j + B_j / T_j^{k_j})) + \lambda \frac{d}{dT_i}(?\,T_j^2 - T_{ASM}^2) = 0 \qquad (i = 1, \ldots n)$$

$$- k_i B_i / T_i^{(k_i+1)} + \lambda\, 2\, T_i = 0 \qquad (i = 1, \ldots n)$$

Solving for λ:

$$\lambda = \frac{k_i B_i / T_i^{(k_i+1)}}{2T_i} \qquad (i = 1, \ldots n)$$

Eliminating λ by expressing each tolerance in terms of T_1:

$$T_i = \left(\frac{k_i B_i}{k_1 B_1}\right)^{1/(k_i+2)} T_1^{(k_1+2)/(k_i+2)} \qquad (i = 1, \ldots n) \qquad (1)$$

Substituting into the assembly tolerance sum:

$$T_{ASM}^2 = T_1^2 + \sum \left(\frac{k_i B_i}{k_1 B_1}\right)^{2/(k_i+2)} T_1^{2(k_1+2)/(k_i+2)} \qquad (2)$$

Equation 2 may be solved iteratively for T_1 and the result substituted into **Equations 1** to determine the set of component tolerances, T_i, which yield minimum cost.

The above derivation assumes a statistical sum model (**RSS** or Root Sum Square) for tolerance accumulation constraint. It could also be derived assuming a linear sum model (**WC** or Worst Case), in which case **Equation 2** becomes:

$$T_{ASM} = T_1 + \sum \left(\frac{k_i B_i}{k_1 B_1} \right)^{1/(k_i+1)} T_1^{(k_1+1)/(k_i+1)} \quad (3)$$

The Lagrange Multiplier method, as derived above, is an efficient tolerance design tool because it is a closed form solution for the optimum tolerances. However it has certain limitations:

1. It can not treat discontinuous cost-tolerance functions, because it requires a continuous first derivative.

2. It can not treat cost-tolerance functions for which preferred tolerance limits are specified, that is, where the process is limited to a specified tolerance range.

3. It has difficulty when applied to assemblies with inter-dependent tolerance loops or chains, that is, assemblies which are described by more than one assembly function with shared dimensions, because this requires the simultaneous solution of a nonlinear system of equations. The nonlinear programming method of Lee and Woo[6] is able to handle multiple loops and nonlinear assembly functions.

A systematic procedure which largely overcomes the first two limitations will be described in a later section. It will also be shown how tolerance allocation can be combined with process selection to extend this method further.

2 COMBINATORIAL PROCESS SELECTION METHODS

When several alternative processes are available for each component dimension of an assembly, a discrete optimization problem results. For example, consider a three part assembly, with cost-vs-tolerance functions are as shown in **Figure 3**. There are two alternative processes each for producing Parts 1 and 2, and three alternative processes for Part 3.

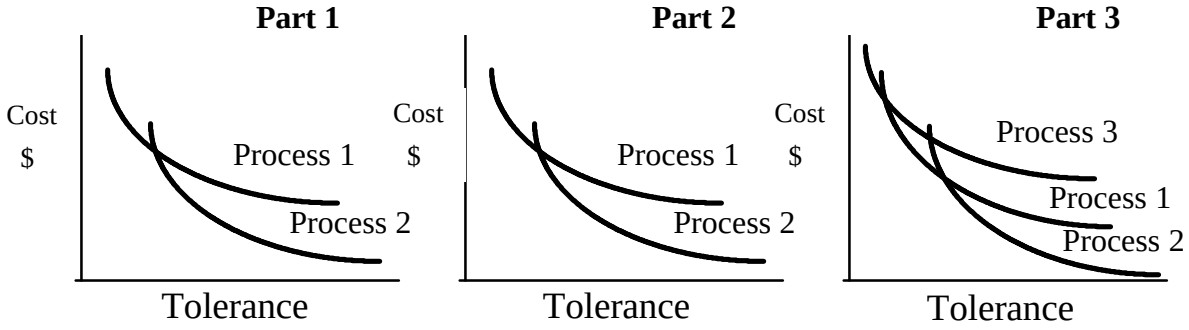


Figure 3. Example Problem - Process Cost Curves

Exhaustive Search

One method of finding the minimum cost tolerance allocation is to perform an exhaustive search over all possible combinations of processes. For the example problem, we would first choose Curve 1 for Part 1, Curve 1 for Part 2 and Curve 1 for Part 3 and then use Lagrange Multipliers to find the optimum tolerances for that set of processes. Next, we would choose Curve 2 for Part 3, while keeping the first curve for the other two parts, and calculate the optimum tolerances for this case. This procedure would be repeated until all possible combinations had been tried. By comparing the resulting assembly cost, the minimum cost specifications may be selected. **Figure**

4 illustrates the procedure as a tree in which one process is selected from each column as you move from left to right.

The number of combinations increases geometrically as the number of parts and the number of processes increases. For the example problem the number is 2 x 2 x 3 = 12. In general, if you have an assembly of **N** parts, each part having $\mathbf{n_i}$ alternative processes, the number of combinations may be calculated from: $\mathbf{n_1 \times n_2 \times n_3 \times \ldots \times n_N}$. Thus, a larger assembly of 10 parts, having just two process cost curves per part, would have 2^{10} or 1024 combinations to check, with each one requiring a Lagrange Multiplier solution for the optimum tolerance allocation.

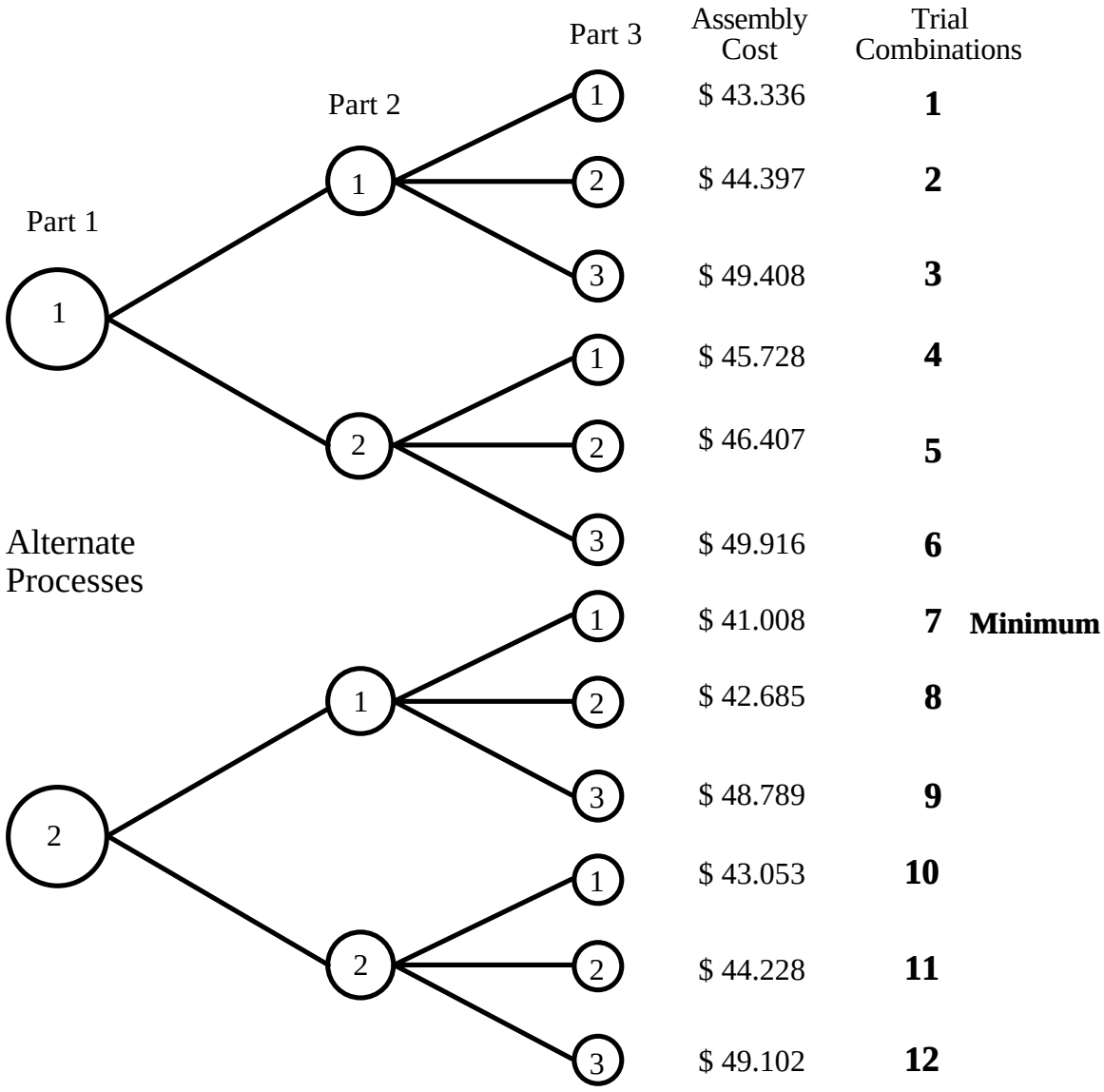


Figure 4. Exhaustive search tree.

Zero-One Discrete Search Method

If the tolerance-cost curves are known only at discrete points, a completely combinatorial search procedure may be used. The method is described by Huang and Ostwald [12]. It is based on an algorithm by Balas [1] in which a set of binary coefficients, having a value of zero or one, are

used to select which combinations of tolerance values and corresponding costs are to be evaluated.

All of the discrete cost values are summed in the assembly cost function, but the binary coefficients, $\mathbf{b_{ij}}$, are used to "turn off" some of the terms. Only one cost value for each part is permitted in a trial evaluation. A complex systematic search procedure evaluates each combination. If the sum of the corresponding tolerances is greater than the assembly tolerance limits, that combination is eliminated. During evaluation, any partial sums which violate a constraint are terminated and eliminated also.

The problem may be expressed algebraically as follows:

Minimize the objective function,

$$\text{Cost} = \sum_{i=1}^{N} \sum_{j=1}^{m_i} b_{ij} C_{ij} \qquad (4)$$

subject to the assembly constraint,

$$\text{tol}_{ASM} = \sum_{i=1}^{N} \sum_{j=1}^{m_i} b_{ij} t_{ij} \qquad (5)$$

and constraining the binary coefficients to assure only one process per component

$$\sum_{j=1}^{m_i} b_{ij} = 1 \, (i = 1,...,N) \qquad (6)$$

where

C_{ij} = the cost of producing the i^{th} component to a specified tolerance using the j^{th} process.

$$b_{ij} = \left\{ \begin{array}{l} 1 \ \text{(on)} \\ 0 \ \text{(off)} \end{array} \right\} \qquad (i = 1,...,N),(j = 1,...,m_i) \qquad (7)$$

N = number of parts

m_i = number of discrete points for Part_i

The assembly constraint shown above is for a linear sum or Worst Case model. A statistical, or RSS model could also have been chosen.

In the example problem, suppose the cost-tolerance curve for each process were described by two discrete points, corresponding to the cost for two typical tolerances, rather than a continuous cost-tolerance function. The seven process curves would then be reduced to 14 discrete points. Since each point may be either turned on or off, there would be 2^{14} = 16,384 possible combinations. The larger, 10 part, two processes-per-part assembly would yield 2^{20} = 1,048,576 combinations. Of course, many of these would be eliminated before complete evaluation due to constraint violation. Huang and Ostwald [12] claim that for moderate to large problems, up to 2^{30} in size, only 2 to 10 percent of the total combinations need be evaluated.

Another method of combining tolerance allocation with process selection using discretized cost-tolerance curves was presented by Lee and Woo[7]. Using a "Branch and Bound" selection method, they were able to prune the search tree substantially. In one case, they reduced a problem which had over 1.5 million combinations by Exhaustive Search to just 2554, which is 1.6 percent.

There is no separate solution for the optimum tolerance allocation for discrete methods. Each evaluation is only a summation and a comparison with the current miminum cost. The more discrete points chosen to represent the process cost curves, the more combinations there will be to evaluate and the closer the optimum tolerance values and cost will be to those found by continuous methods.

Univariate Search Method

The Zero-One method considers many combinations which are invalid since it can select more than one process curve per component for trial evaluation. By restricting the search to valid combinations, the search space is reduced to the exhaustive search tree, which may still be quite large.

If we treat this discrete set of possible assembly costs as a sampled surface, we can apply techniques developed for continuous search surfaces. The method proposed here is an application of the Univariate Search method in which one searches over all the processes for each component part one at a time to find the minimum cost process for each part.

This strategy greatly reduces the search tree, however, it is not guaranteed to find the global minimum. It is based on the assumption that the relative cost for a one-dimensional search does not depend on processes selected for the other parts. This is probably true if the tolerance allocation calculations do not make large changes in the tolerances. But, if one of the component tolerances changes enough to move to a section of the cost-tolerance plot where the curves have crossed over, so their order has changed, the global minimum may get by-passed. A modified strategy will be presented which appears to increase the probability of finding the global minimum.

Figure 5 illustrates the application of the Univariate Search method to the example problem. The method begins by arbitrarily selecting the first process for all three parts and evaluating the cost by Lagrange Multipliers. The first univariate search is performed by evaluating each of the three alternate processes of Part 3, while holding the processes constant for Parts 1 and 2. Note that the tolerances for Parts 1 and 2 are not held constant. They are adjusted by the Lagrange Multiplier allocation scheme for each process tried for Part 3.

After determining that Process 1 for Part 3 yields the minimum cost, we maintain Process 1 for Part 3 for the remainder of the search. The second univariate search is over the process curves for Part 2. We find the minimum cost also occurs for Process 1 for Part 2, so it is held fixed thereafter. The final search is over the processes for Part 1. The minimum occurs for Process 2. The best combination then is the process set 2,1,1 and the corresponding allocated tolerances.

The possible combinations by the Exhaustive Search method was 12. The Univariate Search method reduced this to five. In general, for a well behaved search tree, the Univariate method will require $\mathbf{1 + n_1 + n_2 + ... + n_N - N}$ evaluations, where **N** is the number of parts and $\mathbf{n_i}$ are the number of alternate processes for Part_i, as defined previously. Ideally, the 10 part assembly with two processes per part would require just 11 combinations, compared to a 2.0 percent estimate of 20,971 for the Zero-One method.

However, there are other factors that make the Univariate method slightly less efficient than this. These factors include discontiuities in the search surface due to tolerance allocation noise and process tolerance limits. Each will be discussed later. It was found that due to some of these effects, the Univariate search sometimes found only a local minimum. But, by repeating the Univariate search, beginning at the local minimum, the system was generally able to find the global minimum. The search was stopped when a repeated cycle produced no reduction in the minimum cost. In practice, two or three complete cycles of the Univariate search were required to be assured of finding the global minimum, which is still a substantial reduction.

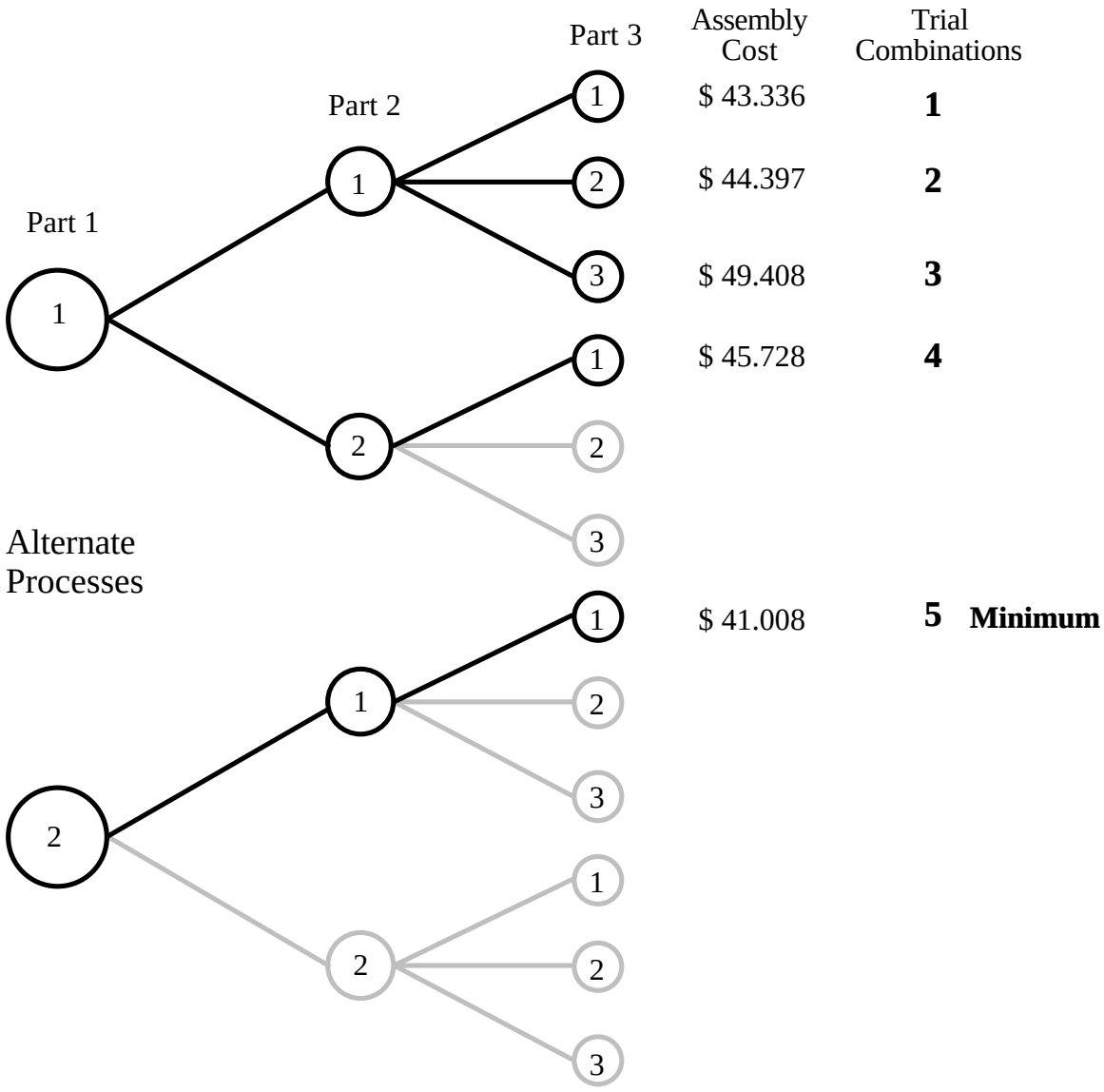


Figure 5. Sample Tree for Univariate Search

3 NONLINEAR PROGRAMMING METHODS

In nonlinear programming, the computer systematically searches for the minimum of an objective function by testing the local gradient and constraints. Nonlinear programming may be applied to the cost-minimization problem, but care must be taken in setting up the problem. For example, an exhaustive search could be made, using the optimization algorithm only to allocate the tolerances for each combination of the process curves, but this would be inefficient. It would be preferred to include the process curves in a search procedure which eliminates most of the combinations required by an exhaustive search.

Continuous Zero-One Selection Coefficients

The discrete optimization problem created by having alternative processes may be transformed to a continuous problem by letting the binary coefficients vary continuously from zero to one. They may then be included as search parameters under control of the optimization algorithm, along with the component tolerances. The intent is that the optimization procedure will drive each factor to a zero or a one rather than 20 percent of one process cost and 80 percent of another.

This seems reasonable since, for any trial value of the tolerance, one cost curve will have a lower value of cost than the other, thus the algorithm will prefer the lower value and adjust the b_{ij} accordingly.

The problem may be expressed algebraically as follows:

Minimize the objective function,

$$\text{Cost} = \sum_{i=1}^{N} \sum_{j=1}^{m_i} b_{ij} C_{ij} \tag{8}$$

subject to the assembly constraint,

$$\text{tol}_{ASM} = \sum_{i=1}^{N} \sum_{j=1}^{m_i} b_{ij} t_{ij} \tag{9}$$

and constraining the binary coefficients to assure only one process per component

$$\sum_{j=1}^{m_i} b_{ij} = 1 \; (i = 1,...,N) \tag{10}$$

$$0 \leq b_{ij} \leq 1 \qquad (i = 1,...,N),(j = 1,...,m_i) \tag{11}$$

where

C_{ij} = the cost of producing the ith component to a specified tolerance using the jth process.
b_{ij} = the decision variable which selects the jth process for the ith dimension.
N = number of parts.
m_i = number of process cost for $part_i$.

A WC assembly constraint was assumed above. RSS could also have been used.

Although iterative search methods are less efficient for small problems, they have the advantage that optimum tolerance allocation and process selection are performed simultaneously, so the search surface is smoother. However, they cannot guarantee finding a global minimum if local minima exist.

A general design optimization software package was used for this part of the study. This system was developed at Brigham Young University and has many features for controlling and monitoring the progress of the search [13]. Several search algorithms are available in the program. The Sequential Quadratic Programming (SQP) algorithm was chosen for this study because it is robust and one of the more efficient algorithms. A subroutine was provided containing the objective function and constraints.

4 SURFACE NOISE

The sampled surface upon which the process optimization takes place does not describe a smooth, continuous surface. There are two principal effects which introduce noise: 1) tolerance allocation and 2) process limits.

Tolerance Allocation Effects

For each trial combination of process curves a tolerance allocation must be performed to find the minimum cost tolerances. A change from one curve to another causes a step-change in cost which may result in a pronounced shift in distribution of the tolerances. If there are N parts, the search surface has N dimensions, but there are N additional tolerance variables contributing to cost variations. Thus, the surface is noisy and likely to have local minima which can halt a search prematurely. The nonlinear programming method (SQP) was hampered by local minima caused by allocation noise, while the Univariate Search method seemed robust enough to be unaffected. The Exhaustive Search was unaffected because it always tried every combination. The Zero-One method did not allocate tolerances for each trial combination, so it was not affected.

Effects of Process Limits

Preferred process tolerance limits may be specified for a given process, causing the cost-tolerance curve to be valid over a finite range. Adjusting the tolerances to accommodate these limits is another source of discontinuities, or noise, in the resulting cost surface.

Process tolerance limits come in three grades of severity. The most severe is a two-sided limit, as shown in **Figure 6**, in which each curve has an upper and a lower tolerance limit with no overlap. If the optimum tolerance without limits should fall in one of the gaps, the search process must pull it back inside the allowable range, resulting in a change in cost. Two-sided limits were the most noise-producing, considerably greater than allocation noise. The level of severity is decreased by a one sided limit, in which each curve has only a lower limit and no upper limit is specified. The lowest level of severity is an overall two-sided limit, in which all the curves for a given part have the same upper and lower limits.

All of the methods except Zero-One had trouble finding the global optimum when process limits were specified. For the Zero-One method, points outside the process tolerance limits were simply not included in the initial data set of discrete points. The SQP method had particular difficulty as it sometimes split the cost between two processes for a single part, making the method unacceptable for such problems.

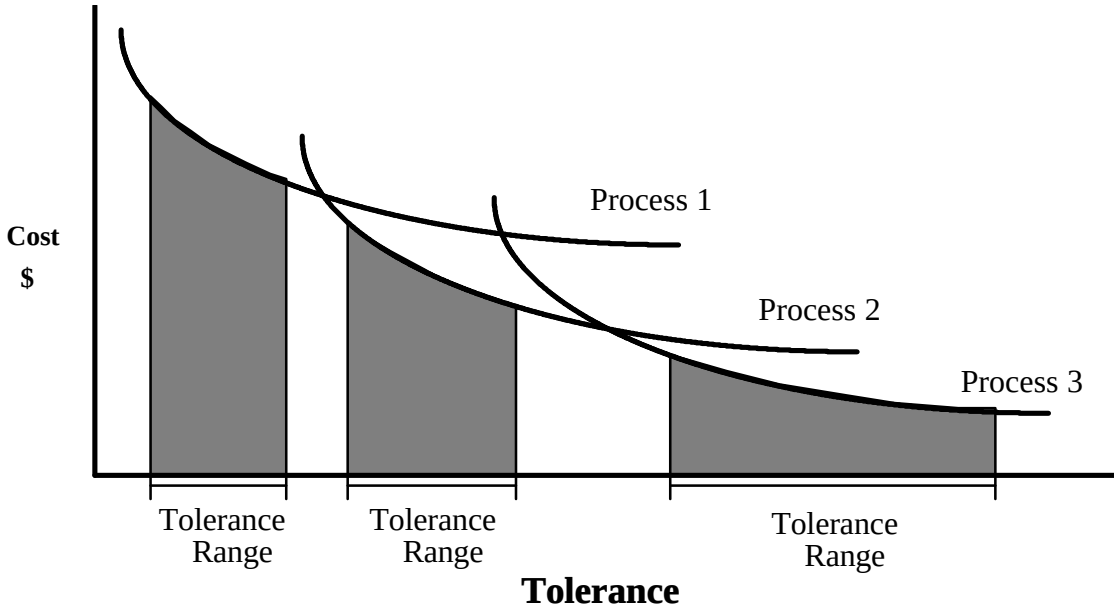


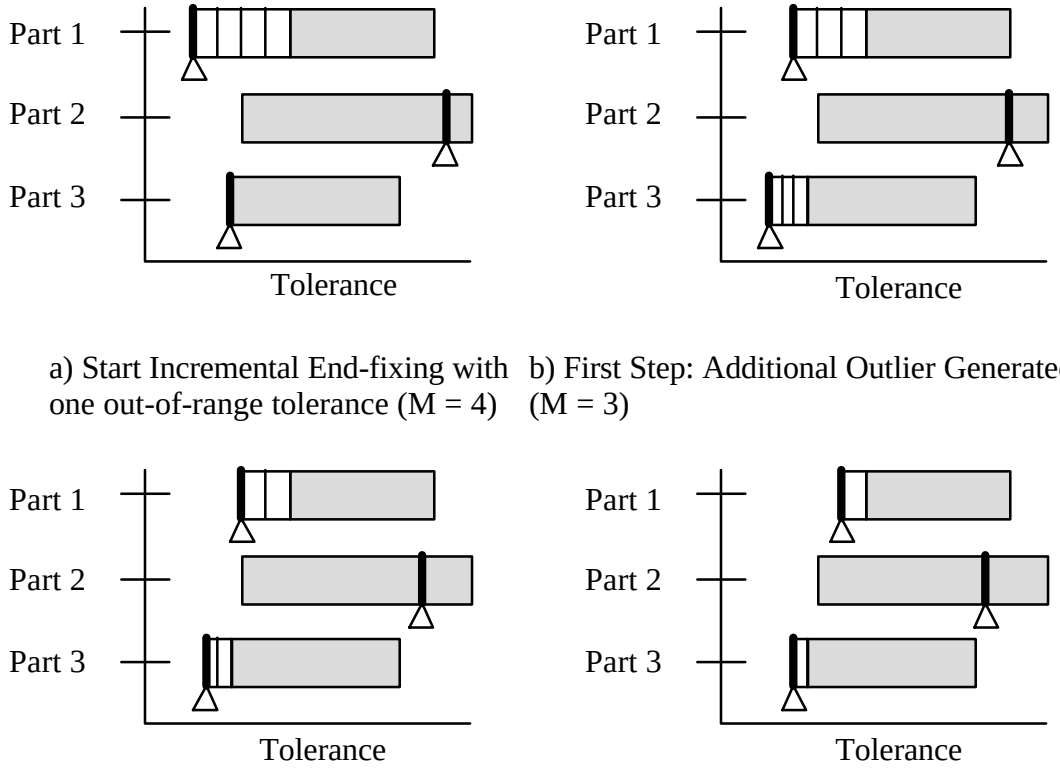
Figure 6. Bounded Process Cost Curves

Minimizing Noise with Incremental Limits

A system for applying process tolerance limits incrementally was developed which greatly reduced the effects of noise on combinatorial search procedures. The tolerance allocation is first performed without invoking the specified limits. The resulting tolerances are checked to see if any are outside the limits. Those that are found to be in violation are gradually pulled back inside the allowable range by adjusting the offending tolerance and holding it fixed, while repeating the optimization. The out-of-limit tolerances are brought a little closer to the limit for each iteration until the limit is reached. In practice it was found that only four incremental iterations were needed to bring all the tolerances within limits. **Figure 7** illustrates the method.

In the first panel of **Figure 7**, following tolerance allocation, Part 1 is found to be out of tolerance. The amount that the tolerance for Part 1 exceeds its nearest limit is divided into four equal increments, M=4. Panel (b) shows the tolerance for Part 1 is adjusted one increment toward the limit and held fixed while the tolerance allocation is repeated. The resulting tolerance for Part 3 pops out of range due to the allocation procedure. Its excess is divided into three equal increments, M=3, and fixed while the tolerance allocation is repeated a second time. No additional tolerances move out of range, so the allocation is simply repeated two more times, as shown in Panels (c) and (d), while decreasing the excess tolerances to zero.

The application of this procedure effectively extends the method of Lagrange Multipliers into constrained optimization space. The only danger in this procedure is if all of the tolerances should become fixed before they have been brought back into range. In this case, it is simply concluded that no optimum exists and the system moves on to another process combination. End-fixing was not required for SQP or Zero-One.



a) Start Incremental End-fixing with one out-of-range tolerance (M = 4)

b) First Step: Additional Outlier Generated (M = 3)

c) Second Step:Continue Constraint Application (M = 2)

d) Third Step: One Step From Full Constraint (M = 1)

Figure 7. Incremental End-fixing

5 RESULTS

A set of test problems of various sizes was used to test the effectiveness of the different methods. The smallest problem had 4 components and 10 process curves, the largest had 13 components and 38 process curves. The methods were compared on the basis of: 1) the number of combinations required to find the minimum cost assembly, 2) the number of CPU seconds of computer time required to find the minimum, and 3) whether or not the global minimum was reached. All the problems were analyzed using a Worst Case assembly constraint.

The number of combinations required by the Exhaustive Search was used as the standard for comparison. This measure was not quite accurate for the Zero-One method since many trial combinations were only partially evaluated. The number of trial combinations for the SQP method was the number of function evaluations required. The computer time measure may not be used to compare the methods to each other due to a change of computers half-way through the project. However, the growth in CPU time with problem size is clearly demonstrated.

Unconstrained Processes

Problems A through I were used to test the effects of problem size on the different methods. For each problem, the number of components in the assembly and the total number of process curves are presented. Problem E has three tolerance loops with shared dimensions. The rest are single loop problems. **Table 1** presents the efficiency in terms of the number of trial evaluations required to find the minimum cost processes and tolerances. The Exhaustive Search and Zero-One methods appear to grow geometrically, the SQP method exhibits less extreme growth, while the Univariate appears to grow linearly with problem size. The same trends are also evident in the CPU results of **Table 2**.

The cost comparisons in **Table 3** show that all of the methods except Zero-One were reasonably successful in finding the minimum cost. Since the Zero-One method uses discrete points, it could only find the closest tolerance to the optimum. With more points, it could get closer. The accuracy of the SQP method depends upon the minimum step size permitted for the tolerance iteration.

The SQP method, however, failed to find the global minimum for five of the problems. For each of these problems it chose a different process for one or two of the components than the Exhaustive Search. Three of the SQP solutions also reached process limits, which drove the process cost up a little. Limits had been set wide to permit an unconstrained search, but one or two component tolerances were constrained in each of the three problems.

The asterisks in the tables indicate that no solution was obtained. Problem E could not be resolved by the Exhaustive or Univariate methods since Lagrange Multipliers cannot handle multiple loop problems. The last two problems were not solved by Zero-One because the computer time was excessive.

Table 1. Trial Combinations Required by Process Selection Methods

Problem	#Comp	#Proc	Exhaust	Univar	Zero-One	SQP	
A	4	10	36	14	205	335	
B	6	13	96	16	681	512	#
C	7	15	192	18	1,344	371	#
D	8	19	864	24	3,275	>>858	
E	8	20	***	***	10,839	931	
F	12	24	4096	26	20,049	>>1,371	#
G	7	30	25,600	48	133,556	>>1,524	
H	12	36	531,441	50	*******	>>2,335	
I	13	38	1,062,882	52	*******	>>2,540	

>> Indicates failure to find global minimum. # Indicates search reached a constraint.

Table 2. CPU Seconds for Process Selection Methods

Problem	#Comp	#Proc	Exhaust	Univar	Zero-One	SQP
A	4	10	0.16	0.06	0.95	23.05
B	6	13	0.59	0.12	5.80	25.85 #
C	7	15	1.24	0.14	15.97	18.05 #
D	8	19	6.50	0.21	53.68	>>31.69
E	8	20	*****	***	190.90	185.92
F	12	24	67.27	0.34	649.47	>>N/A #
G	7	30	175.00	0.38	3464.48	>>225.40
H	12	36	5388.00	0.64	*******	>>400.35
I	13	38	11616.00	0.73	*******	>>460.40
	Computer		VAX	VAX	HP	HP

>> Indicates failure to find global minimum. # Indicates search reached a constraint.

Table 3. Minimum Cost Found by Process Selection Methods

Problem	#Comp	#Proc	Exhaust	Univar	Zero-One	SQP
A	4	10	$21.87	$21.87	$25.00	$21.93
B	6	13	29.05	29.05	36.00	30.09 #
C	7	15	17.98	17.98	31.00	18.05 #
D	8	19	28.84	28.84	40.00	>>31.70
E	8	20	*****	***	5.11	4.27
F	12	24	67.27	67.27	99.00	>>73.26 #
G	7	30	29.50	29.50	51.00	>>30.29
H	12	36	53.62	53.62	*******	>>54.53
I	13	38	55.17	55.17	*******	>>56.05

>> Indicates failure to find global minimum. # Indicates search reached a constraint.

Constrained Processes

This set of problems was used to test the noise-producing effects of processes with tolerance limits and mixed exponents in the tolerance-cost functions. Problems J, K, and KK were analyzed for three cases of process limits: 1) None, 2) Lower limit only, and 3) Non-overlapping upper and lower limits. Limits were chosen to assure end-fixing would be required. Only the Exhaustive Search and Univariate were tested. Zero-One and SQP were eliminated due to inefficiency and inability to find the global minimum.

Problems A through I in the previous section had the same tolerance exponent for all the components in the assembly. It was held constant at a value of -1.0. For problems K and KK in this series of tests, the exponents were -1.0, -2.0, and -3.0 for the first, second and third process for each part, respectively. Problem J was identical to K, except it was given a constant exponent value of -2.0 for all processes for comparison purposes.

Examination of **Table 4** reveals the increased number of trials required when process tolerance limits are present. The more restrictive the limits, the more difficulty there was in finding the global minimum. The increased number of trials is due to the gradual application of the tolerance limits when any optimized tolerances are out of range.

Problem KK was just Problem K doubled to create a bigger problem. A second set of parts and processes was added, identical to the original set. **Table 4** shows that the Univariate method works better on a large problem than a small one, since it failed to find the global minimum for two cases on the smaller problem, while it succeeded when the same problem was doubled in size with identical data.

Table 5 indicates that the Univariate method clearly conserves computer time compared to the Exhaustive Search method. **Table 6** shows that even for the cases in which the Univariate method failed to find the global optimum, the cost was not far off.

Table 4. Trial Combinations Required by Process Selection Methods

Problem	#Comp	#Proc	Limits	Expon	Exhaust	Univar
J	3	7	None	Const	12	10
J	3	7	Lower	Const	66	32
J	3	7	Up/Low	Const	76	57
K	3	7	None	Mixed	12	>>10
K	3	7	Lower	Mixed	49	32
K	3	7	Up/Low	Mixed	74	>>62
KK	6	14	None	Mixed	144	27
KK	6	14	Lower	Mixed	811	75
KK	6	14	Up/Low	Mixed	886	115

>> Indicates failure to find global minimum.

Table 5. CPU Seconds for Process Selection Methods

Problem	#Comp	#Proc	Limits	Expon	Exhaust	Univar
J	3	7	None	Const	0.06	0.03
J	3	7	Lower	Const	0.18	0.10
J	3	7	Up/Low	Const	0.15	0.16
K	3	7	None	Mixed	0.08	>>0.07
K	3	7	Lower	Mixed	0.15	0.11
K	3	7	Up/Low	Mixed	0.14	>>0.14
KK	6	14	None	Mixed	1.50	0.29
KK	6	14	Lower	Mixed	4.66	0.51
KK	6	14	Up/Low	Mixed	2.90	0.42

>> Indicates failure to find global minimum.

Table 6. Minimum Cost Found by Process Selection Methods

Problem	#Comp	#Proc	Limits	Expon	Exhaust	Univar
J	3	7	None	Const	$41.01	$41.01
J	3	7	Lower	Const	41.01	41.01
J	3	7	Up/Low	Const	45.73	45.73
K	3	7	None	Mixed	29.84	>>29.99
K	3	7	Lower	Mixed	30.05	30.05
K	3	7	Up/Low	Mixed	35.09	>>35.34
KK	6	14	None	Mixed	59.67	59.67
KK	6	14	Lower	Mixed	60.10	60.10
KK	6	14	Up/Low	Mixed	68.74	68.74

>> Indicates failure to find global minimum.

6 CONCLUSIONS

1) Of the four methods evaluated for performing a combined minimum cost tolerance allocation and process selection, the **Exhaustive Search** method is the most reliable procedure for finding the global minimum. It is only practical for problems with fewer than 25 or 30 variables. However, most problems would likely fall within this size range.

2) The **Zero-One** method is too inefficient to be of practical value. **Branch and Bound** is far more efficient, however, to get results close to the global optimum requires several discrete points for each cost-tolerance curve, spaced as closely as the required resolution. The problem then becomes dramatically larger, which reduces the overall efficiency. On the other hand, Branch and Bound has no problem with process limits or multiple loop assembly functions.

3) The **SQP** method is also cabable of treating multiple loop assembly functions. However, it cannot guarantee to find the global minimum. The inability to deal with non-overlapping process limits would also have to be overcome.

4) The **Univariate Search** method is the most efficient of the processes tested by a wide margin. While it cannot guarantee finding the global minimum, it always found it for the unconstrained problems and moderate-to-large constrained problems. But for very small constrained problems it does not perform well. Apparently, it needs more room to move around. It also requires special procedures for handling process limits. The fact that it searches over the whole space in each search direction makes it less sensitive to local minima than SQP.

5) Neither the Exhaustive Search nor the Univariate Search methods in their present form can treat multiple loop assemblies. Current research is underway to remove this limitation.

6) The **End-fixing** method of gradually applying process tolerance constraints appears to be an effective means of extending unconstrained process selection procedures into the more difficult arena of constrained optimization.

7 SUMMARY

In summary, it has been demonstrated that the methods for tolerance allocation for minimum production cost can be extended to include process selection from a set of alternate processes. The recommended procedure is to perform an Exhaustive Search for problems with fewer than 25 process variables and a Univariate Search for larger problems, and to implement the End-fixing algorithm when process limits are specified. It would also be wise to report the four or five lowest cost combinations of processes so a decision could be made on the basis of the availability of machines or other considerations.

The recommended methods are simple enough and efficient enough to be practically implemented on a PC for the majority of problems. They are currently being integrated into a software package that is designed for engineers. It is hoped that they will be accepted by the engineering design community and used to specify design tolerances on a rational basis.

The biggest drawback for this new concept is the total lack of data on which to base the cost-vs-tolerance functions that are so necessary to the analysis. Currently, a co-operative effort is underway with local industries to gather data for this purpose under NSF sponsorship.

This study was sponsored by the Association for the Development of Computer-Aided Tolerance Software (ADCATS), which is a joint effort by Brigham Young University and 12 industrial and government sponsors. The complete reports may be found in References 5 and 8.

REFERENCES

1. Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," Operations Research, vol. 13, 1965, pp. 517-546.

2. Bjorke, O., Computer-Aided Tolerancing, Tapir Publishers, 1978.

3. Chase, K. W., and Greenwood, W. H., "Design Issues in Mechanical Tolerance Analysis," Manufacturing Review, ASME, vol. 1, no. 1, Mar. 1988, pp. 50-59.

4. Edel, D. H., and T. B. Auer, "Determine the Least Cost Combination for Tolerance Accumulations in a Drive Shaft Seal Assembly," General Motors Engineering Journal, Fourth Quarter, 1964, pp.37-38, First Quarter, 1965, pp. 36-38.

5. Hauglund, L. F., "Combining Manufacturing Process Selection and Optimum Tolerance Allocation in Design Automation," MS Thesis, Brigham Young University, Provo. UT, 1987.

6. Lee, W. and T. C. Woo, "Tolerancing: It's Distribution, Analysis, and Synthesis," Tech. Report No. 86-30, Dept. of Indust. and Operations Engineering, U. of Michigan, Ann Arbor, MI, Dec. 1986.

7. Lee, W. and T. C. Woo, "Optimum Selection of Discrete Tolerances," Tech. Report No. 87-34, Dept. of Indust. and Operations Engineering, U. of Michigan, Ann Arbor, MI, Dec. 1987.

8. Loosli, B. G., "Manufacturing Tolerance Cost Minimization Using Discrete Optimization for Alternate Process Selection," MS Thesis, Brigham Young University, Provo. UT, 1987.

9. Michael, W., and Siddall, J. N., "The Optimization Problem With Optimal Tolernce Assignment and Full Acceptance," J. of Mechanical Design, ASME, vol. 103, Oct. 1981, pp. 842-848.

10. Michael, W., and Siddall, J. N., "The Optimal Tolerance Assignment with Less Than Full Acceptance," J. of Mechanical Design, ASME, vol. 104, Oct. 1982, pp. 855-860.

11. Monte, M.E., and Datseris, P., "Optimum Tolerance Selection for Minimum Manufacturing Cost and Other Criteria," ASME Paper No. 82-DET-35, 1982, pp. 1-9.

12. Ostwald, P. F., and Huang, J., "A Method for Optimal Tolerance Selection," J. of Engineering for Industry, ASME, vol.99, Aug. 1977, pp. 558-565.

13. Parkinson, A. R., R. J. Balling, and J. C. Free, "OPTDES.BYU: A Software System for Optimal Engineering Design," Computers in Engineering, vol. 1, ASME, 1984, pp. 429-434.

14. Parkinson, D. B., "Assessment and Optimization of Dimensional Tolerances," Computer-Aided Design, vol. 17, No. 4, May 1985, pp. 191-199.

15. Patel, A. M., "Computer-AidedAssignment of Manufacturing Tolerances," Proc. of the 17th Design Automation Conf., Minneapolis, Minn., June 1980.

16. Peters, J., "Tolerancing the Components of an Assembly for Minimum Cost," J. of Engineering for Industry, ASME, Aug. 1970, pp. 677-682.

17. Speckhart, F. H., "Calculation of Tolerance Based on a Minimum Cost Approach," J. of Engineering for Industry, ASME, vol. 94, May 1972, pp. 447-453.

18. Spotts, M. F., "Allocation of Tolerances to Minimize Cost of Assembly," J. of Engineering for Industry, ASME, vol. 95, Aug. 1973, pp. 762-764.

19. Sutherland, G. H., and Roth, B., "Mechanism Design: Accounting for Manufacturing Tolerances and Costs in Function Generating Problems," J. of Engineering for Industry, ASME, vol. 97, Feb. 1975, pp. 283-286.

20. Wilde, D., and Prentice, E., "Minimum Exponential Cost Allocation of Sure-Fit Tolerances," J. of Engineering for Industry, ASME, vol. 97, Nov. 1975, pp. 1395-1398.

21. Wu, Z., W. H. ElMaraghy and H. A. ElMaraghy, "Evaluation of Cost-Tolerance Algorithms for Design Tolerance Analysis and Synthesis," Manufacturing Review, ASME, vol. 1, no. 3, Oct. 1988, pp. 168-179.