4.1 Introduction

Design optimization can be defined as the process of finding the optimal parameters, which yield maximum or minimum value of an objective function, which must also satisfy a certain set of specified requirements called constraints. In general, an engineering design optimization problem can be expressed as in Deb [Deb95], Rao [Rao96] and Reklaitis et al. [Rek82].

Minimize $f(x)$ with constraints (Restrictions)

$$G_i(x) \geq 0 \quad \text{where } i = 1, 2, 3, \ldots, m$$
$$H_j(x) = 0 \quad \text{where } j = 1, 2, 3, \ldots, n$$

One of the major difficulties that classic linear programming and non-linear programming methods face for solving optimization problems is that they stuck in the local minima when applied to the constrained design and manufacturing optimization problems. Difficulties arise because either the computation requirements quickly become intractable as the size of the problem increases, or the constraints are violated.

The last two decades have witnessed tremendous growth in the application of stochastic search techniques over classic linear programming and non-linear programming methods. The primary reason for this is that these are well suited to the concurrent manipulation of models of varying resolution and structure. This is due to their ability to search non-linear space without gradient information or a prior knowledge relating to model characteristic. The most important stochastic search techniques that have been popular are Evolutionary Strategies (ES), Genetic Algorithms (GA), Simulated Annealing (SA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Immune Algorithm and Tabu Search (TS).
Many efforts have been made by researchers to overcome limitations of earlier algorithms such as slow and premature convergence by establishing a good balance between exploitation (local search) and exploration (global search). One such effort resulted in the hybridization of Evolutionary Algorithm (EA) with other heuristics such as simulated annealing, local search, tabu search, hill climbing, dynamic programming, greedy random adaptive search procedure and quantum computing concepts. This hybridization resulted in the improvement of performance in terms of convergence speed and quality of the solutions obtained by EA. [Gan05, Nall11]

Hans Raj et al., [Hans05] have proposed a hybrid Evolutionary Computational Technique (ECT) by combining GA and SA. It is a hybrid scheme which incorporates a real-coded GA to provide multi-point search along with simulated annealing method to overcome local convergence and the problem of multiple minima. This technique provides more rapid and robust convergence on many function optimization problems. Two levels of competition are introduced between the strings in the population to ensure that only the better strings continue in the population. The concept of “Acceptance Number” is introduced to ensure that more computational effort is devoted to search in “better” regions of the search space. Constraints are handled by the use of the concept of penalty functions and by better coding.

In the presented work a new Quantum Seeded Hybrid Evolutionary Computational Technique (QSHECT) is developed which incorporates ideas from the principles of quantum computation and integrates them in the current framework of hybrid ECT as proposed by Hans Raj et al. [Hans05]. QSHECT is a flexible and real coded computational technique used here for solving, mechanical engineering design optimization problems. This technique incorporates Simulated Annealing (SA) in the selection process of Genetic Algorithm (GA). It has been designed with genetic operator called the blend crossover (BLX) to provide a better search capability. Other features include concentration of search in “better areas” of search space and incorporation of some constraints into the coding. Extensive testing has been carried out on standard test problems taken from the literature. The efficiency, effectiveness and ease of application of the proposed technique are demonstrated by solving a gear design problem, a truss design problem and a spring design problem. The results obtained compare favorably with those reported in the literature and suggest the use
of proposed technique in mechanical engineering design and process optimization and are a step forward in meeting the challenges posed in the design by present day intelligent manufacturing systems.

4.2 Quantum Seeded Hybrid Evolutionary Computational Technique (QSHECT)

QSHECT is a hybrid approach, which incorporates SA in the selection process of GA. Thus, QSHECT provides the advantages of both GA and SA. Here the basic steps and the problem specific implementation details are discussed. The algorithm concentrated its search only in the permissible regions of the search space using penalty approach. The solutions obtained using this technique are superior to those obtained by other methods [Aro89], [Bel82], [Coe99], [Hern94], [Kun98], [Li85], [Ray01].

In QSHECT the idea is to seed the initial population with a quantum approach in the framework of ECT. The algorithm concentrated its search only in the permissible regions of the search space using penalty approach. QSHECT search technique starts out with a guess of \( N \) grandparents, chosen at random in the search space. Initially each grandparent generates a number of quantum parents \( Q \). Number of quantum parents is chosen to be 10. Larger population sizes might yield further improvement in the results obtained but would entail higher computational effort. All the design variables are encoded as floating point numbers. The method selects grandparents as random numbers in the range \((0, 1)\) for each element of chromosome. Considerably distant points in the solution space are generated as grandparents to avoid any domination of particular schemata during the initialization process. Initially all grandparents generate an equal number of quantum parents. The advantage of using a quantum seeded generation is that a number of quantum parents are generated with each grandparent without using GA. The idea is taken from the point of view that in a parental string any of the values generated will either be bigger or smaller than another randomly generated number, which is known as probability of finding this string value in a particular state.

If the string value is less than the random probability it is retained as such else it is changed as:
This idea is in accordance with the principles of quantum computation as described in section 1.3, which states that the probability of a qubit to be in any state satisfies the condition \(|\alpha|^2 + |\beta|^2 = 1\). Thus in a single pass numerous quantum parents can be generated with a single grandparent. Each quantum parent is checked over its functional value and constraints violation. From these quantum parents further parents are selected. A quantum parent is made a parent only when it clears a criterion of the sum of penalties for all the constraints violated. Thus the total number of parents selected varies for each iteration and also varies with a particular run of the program. Now these parents are sent into ECT which is combined GA/SA and is used for further child generation. Initially all parents generate an equal number of children given by \(m(i) = M\). The total number of children in a generation is fixed and is given by:

\[
TC = \sum_{i=1}^{N} m(i)
\]

For each parent \(i\), mates are selected from the other parents at random and cross-over is applied to generate \(m(i)\) children. For each family a blend cross-over operator (BLX-\(\alpha\)) based on the theory of interval schemata is employed in the study. BLX-\(\alpha\) operates by randomly picking a point in the range \((p_1 - \alpha (p_2 - p_1), p_2 + \alpha (p_2 - p_1))\) where \(p_1\) and \(p_2\) are two parent points and \(p_1 < p_2\). In a number of test problems, BLX-0.5 performed better than the BLX operators with any other \(\alpha\) values and has, therefore, been used. The ECT differs from simulated evolution in that there are two levels of competition. The first is the local competition. The children in the same family (i.e. generated from the same parent) compete with each other and only the one with the better objective value survives. The best child then competes with its parent to select the parent for the next generation. If the best child is better than its parent it is accepted as a parent in the next generation. If the best child is worse than its parent then Boltzmann criterion is applied before the child be accepted. As in SA, the selection of temperatures is such that initially the probability of acceptance of a bad move, i.e. when the best child is worse than the parent is high (approximately 1) but as the temperatures are successively lowered through a cooling schedule, this probability is decreased until, at the end, the probability of accepting a bad move is negligible (approximately 0). Logarithmic cooling schedule is adopted in this work.
Such a strategy enables the technique to seek the global optimum without getting stuck in any local optimum. The initial and final temperatures are calculated as follows. A bad move is accepted according to the Boltzmann Criterion. Initially the probability of accepting a bad move is approximately one, i.e.

$$\exp\left(-\frac{\Delta X_{\text{average}}}{T_1}\right) = 0.99$$

and finally

$$\exp\left(-\frac{\Delta X_{\text{average}}}{T_{\text{MAXIT}}}\right) = 0.0001$$

therefore

$$T_1 = -\frac{\Delta X_{\text{average}}}{\log(0.99)}$$

$$T_{\text{MAXIT}} = -\frac{\Delta X_{\text{average}}}{\log(0.0001)}$$

Where $T_1$ is the initial temperature, $T_{\text{MAXIT}}$ is the final temperature, $\Delta X_{\text{average}}$ is the average difference between the objectives $X$ for any two neighborhood points in the search space. This average is calculated over a number of chromosomes. This mechanism ensures that the algorithm does not get stuck in local optima. When the temperature is high, the algorithm exhibits good exploration property by accepting even solutions that are inferior to the current solution. This ensures that the search space is adequately explored. When the temperature is reduced the probability of acceptance of the worse solutions is lowered and, in this case, the regions that are identified to be the better regions in terms of having better solutions are exploited to ensure that any “good” solution in that region is not missed. Thus, the algorithm has finer points that enable it to perform better.

The number of children that are generated by a parent is proportional to a parameter called the acceptance number. This is the second level of competition in which each string competes with its siblings to generate more children in the next generation. The number provides a measure of the goodness of solutions in the vicinity of the current parent. This strategy enables the algorithm to focus search on the better regions of the search space. It has been shown that eventually only one family survives. The procedure continues until a certain number of iterations have been reached or until an acceptable solution has been found. It is the effect of the second level of competition that gives a measure of the regional information of how good the region is. If a region is found to contain a larger number of good candidate solutions, more attention is
devoted to search in that region by enhancing the number of children that are generated in that region.

For highly constrained problems, infeasible solutions may occupy a relatively big portion of the population. The penalty technique is perhaps the most common technique used to handle infeasible solutions in the constrained optimization problems. In essence, this technique transforms the constrained problem into an unconstrained problem by penalizing infeasible solutions, in which a penalty term is added to the objective function for any violation of the constraints. The major concern is how to determine the penalty term so as to strike a balance between the information preservation (keeping some infeasible solutions) and the selective pressure (rejecting some infeasible solutions), and avoid both under penalty and over penalty. There are no general guidelines on designing penalty function. Constructing an efficient penalty function is quite problem dependent. The same has been incorporated in the QSHECT algorithm in evaluating the objective function i.e.

\[ E_{\text{val}}(x) = f(x) + \beta \times D(x), \]

where, \( \beta \) is the problem dependent constant, and \( D(x) \) is the difference measure for constraint violation. The result of such an approach is that the infeasible strings have much worse objective function values and are eliminated from the population whereas the strings with better objective function values survive and contribute more to the evolution of better solutions.

The number of grandparents taken depends upon what is the criterion for the selection of parents from the quantum parents. The relaxation gives us choice to start with small number of grandparents. It is seen that more constrained the criterion for the selection of parents is, the better is the convergence. The initial generation of the quantum parent ensures that parents with a better fitness value are sent into GA to produce further children. This gives GA a better convergence towards the optimum solution. The results show that the convergence is very fast. The various features explained above have been combined together to develop an optimization algorithm and is represented succinctly in the form of pseudo-code below.
Pseudo-code of QSHECT

1. Generate random initial grandparent strings.
2. Generate a random probability.
3. If any of the string values is less than the random probability it is retained as such.
4. Otherwise it is changed as $\sqrt{1 - \left(\text{String Value}\right)^2}$.
5. Initialize $T_i$ & $T_{MAXIT}$, $N$ parent string and $M$.
6. For each parent $i$, generate $m(i)$ children using crossover
7. Find the best child for each parent (1st level of competition).
8. Select the best child as the parent for the next generation. For each family accept the best child as the parent for the next generation if

   \[ Y_i < Y_2 \text{ OR } \exp\left(\frac{Y_2 - Y_i}{T}\right) \geq \rho \]

   where:
   - $Y_i$ is the objective value of the best child, $Y_2$ is the objective value of its parent,
   - $T$ is the temperature co-efficient, $\rho$ is a random number uniformly distributed between 0 and 1.
9. Repeat step 10 to Step 14 for each family
10. Count = 0
11. Repeat step 12 for each child: Go to step13
12. Increase count by 1, if $\left( (Y_i < Y_2) \text{ OR } \exp\left(\frac{Y_{LOWEST} - Y_i}{T}\right) \geq \rho \right)$

   where:
   - $Y_i$ is the objective value of the child, $Y_2$ is the objective value of its parent
   - $Y_{LOWEST}$ is the lowest objective value ever found
   - $T$ is the current temperature
   - $\rho$ is a random number uniformly distributed between 0 and 1.
13. Acceptance number of the family is equal to count (A)
14. Sum up the acceptance number of all the families (S)
15. For each family $i$, calculate the number of children to be generated in the next generation according to the following formula:

   \[ m(i) = \left( TC \times A \right) / S \]

   where $TC$ is the total number of children generated by all the families
16. Decrease the temperature
17. Repeat Step 6 to Step 16 until a certain number of iterations has been reached.
4.3 QSHECT applied to standard benchmark problems

To evaluate the efficiency and usefulness of QSHECT, four test benchmark problems are solved and the results obtained are compared with the results of Thak and Sun [Tha00] and Evolutionary Computational Technique (ECT) as reported by Hans Raj et al. [Hans05]. For all the test bench problems initial population size is different and offspring population size is 100 for each generation. Maximum number of iterations is taken to be hundred. Penalty term is taken to be:

$$\text{Penalty} = \beta \times \sum_{i=1}^{n} C_i$$

Where $C_i$ is the violation in $i^{th}$ constraint and the value of $\beta$ is problem specific.

The first benchmark problem (G1) is explained in figure given below:

```
Problem G1: Minimize

$$f(x) = 5x_1 + 5x_2 + 5x_3 + 5x_4 - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

subject to:
$$2x_1 + 2x_2 + x_{10} + x_{11} \leq 10$$
$$2x_1 + 2x_3 + x_{10} + x_{12} \leq 10$$
$$2x_2 + 2x_3 + x_{11} + x_{12} \leq 10$$
$$-8x_1 + x_{10} \leq 0$$
$$-8x_2 + x_{11} \leq 0$$
$$-8x_3 + x_{12} \leq 0$$
$$-2x_4 - x_5 + x_{10} \leq 0$$
$$-2x_6 - x_7 + x_{11} \leq 0$$
$$-2x_8 - x_9 + x_{12} \leq 0$$

and bounds  $0 \leq x_i \leq 1$  $i = 1, \ldots, 9$;
$0 \leq x_i \leq 100$, $i = 10, 11, 12$
$0 \leq x_i \leq 1$
```

First benchmark problem (G1)

The function $f(x)$ has a global optimum of $f(x) = -15$ which is located at $x = (1,1,1,1,1,1,1,1,3,3,3,1)$. The results of QSHECT are $f(x) = -15$ at $x = (1,1,1,1,1,1,1,1,3,3,3,1)$. The final results have been obtained with initial population size of 12 and offspring size of 100 in each generation for thirty-five generations as depicted in the convergence graph of objective function in Fig. 4.1.
The second benchmark problem (G7) is explained in figure given below:

**Problem G7: Minimize**

\[
\begin{align*}
 f(x) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + \\
 &4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + \\
 &2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45
\end{align*}
\]

Subject to:
\[
\begin{align*}
 105 - 4x_1 - 5x_3 + 3x_7 - 9x_5 &\geq 0 \\
 -10x_1 + 8x_2 + 17x_7 - 2x_8 &\geq 0 \\
 8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 &\geq 0 \\
 -3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 &\geq 0 \\
 -5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 &\geq 0 \\
 -x_1^2 + 2(x_2 - 2)^2 + 2x_1x_2 - 14x_3 + 6x_6 &\geq 0 \\
 -0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_3^2 + x_6 + 30 &\geq 0 \\
 3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} &\geq 0 \\
\end{align*}
\]

and bounds: \(-10 \leq x_i \leq 10, \ i = 1, \ldots, 10\)
The function $f(x)$ has a global optimum of $f(x) = 24.3062091$ which is located at $x = (2.171996, 2.63683, 8.773926, 5.095984, 1.321644, 9.828726, 8.280092, 8.375927)$. The results of QSHECT are $f(x) = 24.3070$ at $x = (2.1669, 2.3772, 8.7456, 5.0942, 0.9956, 1.4032, 1.3128, 9.8191, 8.2603, 8.3941)$. The final results have been obtained with initial population size of 20 and offspring size of 100 in each generation for fifty generations as depicted in the convergence graph of objective function in Fig. 4.2.

![Fig. 4.2: Convergence graph of the objective function of problem G7](image)

The third benchmark problem (G9) is explained in figure given below:

**Problem G9: Minimize**

$$f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^2 + 7x_6^4 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7$$

Constraints:

$$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0$$

$$282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0$$

$$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0$$

$$-4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0$$

and bounds: $-10 \leq x_i \leq 10$ for $i = 1, \ldots, 7$
The function \( f(x) \) has a global optimum of \( f(x^*) = 680.6300573 \), which is located at \( x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227) \). The results of QSHECT are \( f(x) = 680.630716 \) at \( x = (2.33987, 1.951487, -0.464913, 4.363717, -0.631491, 1.041926, 1.605400) \). The final results have been obtained with initial population size of 20 and offspring size of 100 in each generation for fifty generations as depicted in the convergence graph of objective function in Fig. 4.3.

![Convergence graph of the objective function of problem G9](image_url)

Fig. 4.3: Convergence graph of the objective function of problem G9

The fourth benchmark problem (G10) is explained in figure given below:

**Problem G10: Minimize**

\[
\begin{align*}
f(x) &= x_1 + x_2 + x_3 \\
1 - 0.0025 (x_4 + x_6) &\geq 0 \\
1 - 0.0025 (x_5 + x_7 - x_4) &\geq 0 \\
1 - 0.01(x_8 - x_3) &\geq 0 \\
x_1 x_6 - 833.33252 x_4 - 100 x_4 + 83333.333 &\geq 0 \\
x_2 x_7 - 1250 x_5 - x_2 x_4 + 1250 x_4 &\geq 0 \\
x_3 x_8 - 1250000 - x_3 x_5 + 2500 x_5 &\geq 0 \\
\text{and bounds:} & \\
100 &\leq x_i \leq 10000, \\
1000 &\leq x_i \leq 10000, \text{ for } i = 2, 3 \\
10 &\leq x_i \leq 1000, \text{ for } i = 4, \ldots, 8
\end{align*}
\]
The function \( f(x) \) has a global optimum of \( f(x^*) = 7049.330923 \), which is located at 
\[ x^* = (579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979) \].

The results of QSHECT are \( f(x) = 7049.330 \) at \( x = (562.843, 1342.103, 5135.251, 183.241, 296.856, 212.728, 254.323, 381.440) \). The final results have been obtained with initial population size of 20 and offspring size of 100 in each generation for forty-six generations as depicted in the convergence graph of objective function in Fig. 4.4.

![Convergence graph of the objective function of problem G10](image)

**Fig. 4.4: Convergence graph of the objective function of problem G10**

The results obtained by QSHECT are compared with the results of Thak and Sun [Tha00] and Evolutionary Computational Technique (ECT) as reported by Hans Raj et al. [Hans05] in Table 4.1. Table 4.2 shows statistical information obtained after 50 runs of QSHECT algorithm for each problem.

<table>
<thead>
<tr>
<th>Problem</th>
<th>G1</th>
<th>G7</th>
<th>G9</th>
<th>G10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact Solutions</td>
<td>-15</td>
<td>24,306</td>
<td>680,630</td>
<td>7049,330</td>
</tr>
<tr>
<td>Thak and Sun [Tha00]</td>
<td>-15</td>
<td>24,394</td>
<td>680,651</td>
<td>7074,600</td>
</tr>
<tr>
<td>Hans Raj et al. [Hans05]</td>
<td>-15</td>
<td>24,307</td>
<td>680,650</td>
<td>7049,331</td>
</tr>
<tr>
<td>QSHECT</td>
<td>-15</td>
<td>24,307</td>
<td>680,630</td>
<td>7049,330</td>
</tr>
</tbody>
</table>

**Table 4.1: Solution with QSHECT of test bench problems reported in [Tha00] and [Hans05]**
Table 4.2: Statistical information obtained after 50 runs of QSHECT algorithm for each problem

<table>
<thead>
<tr>
<th>Problem</th>
<th>G1</th>
<th>G7</th>
<th>G9</th>
<th>G10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>-15</td>
<td>24.3070</td>
<td>680.630</td>
<td>7049.330</td>
</tr>
<tr>
<td>Median</td>
<td>-15</td>
<td>23.95</td>
<td>680.645</td>
<td>7123.551</td>
</tr>
<tr>
<td>Mean</td>
<td>-14.6471</td>
<td>24.02397</td>
<td>680.675</td>
<td>7136.462</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.779992</td>
<td>0.177684</td>
<td>0.130095</td>
<td>170.6321</td>
</tr>
<tr>
<td>Worst</td>
<td>-11.9969</td>
<td>23.8726</td>
<td>681.356</td>
<td>7469.332</td>
</tr>
</tbody>
</table>

The results obtained (Table 4.1) illustrates that QSHECT compares favorably with those reported by [Tha00] and ECT reported by Hans Raj et al. [Hans05].

4.4 QSHECT applied to engineering design problems

The proposed algorithm is a powerful approach for determining the optimum of a variety of functions. To study the performance of QSHECT approach in engineering design, three problems of design i.e. design of a gear train, a truss, and a spring are solved. The best results obtained by QSHECT are compared with those reported in the literature by Ray and Saini [Ray01]. For the proposed approach, the operational parameters control the balance between exploitation (using the existing material to best effect) and exploration (searching for better solutions).

4.4.1 Design of a three bar truss

The three bar truss structure is shown in figure 4.5.

![Fig. 4.5: A three bar truss structure](image-url)
The volume of the truss structure is to be minimized subject to stress constraints. The mathematical formulation is presented below:

\[
\text{Minimize } f(x) = [2(2)\frac{1}{2} x_1 + x_2]L
\]

Subject to:

\[
\sigma_1 = \left[ \frac{\frac{1}{2} x_1 + x_2}{(2)^2 x_1^2 + 2x_1x_2} \right] P \leq 2
\]

\[
\sigma_2 = \left[ \frac{1}{x_1 + (2)^\frac{1}{2} x_2} \right] P \leq 2
\]

\[
\sigma_3 = \left[ \frac{x_2}{(2)^\frac{1}{2} x_1^2 + 2x_1x_2} \right] P \leq 2
\]

Where, 0 <= x_1 <= 1 and 0 <= x_2 <= 1. The other constraints are l=100cm, P=2kN/cm^2 and \(\sigma=2kN/cm^2\).

The table 4.2 shows the results obtained for the three bar truss design using QSHECT algorithm. The final results have been obtained with initial population size of 12 and offspring size of 100 in each generation for twenty generations as depicted in Fig. 4.6. It is clearly observed from table 4.3 that the results obtained from QSHECT algorithm are more accurate and consistent than the values reported by Hernandez [Hern94] and Ray [Ray01] and Hans Raj [Hans05]. Table 4.4 shows statistical information obtained after 50 runs of QSHECT algorithm for three bar truss design problem. In solving this problem, following penalty function is used:

\[
\text{Penalty} = 2000 \times \sum_{i=1}^{3} C_i^2
\]

Where \(C_i\) is the \(i^{th}\) constraints deviation from limits.
### Variables

<table>
<thead>
<tr>
<th>Variables</th>
<th>Best</th>
<th>Worst</th>
<th>Mean</th>
<th>Median</th>
<th>Std. Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>X₁</td>
<td>0.788</td>
<td>0.789</td>
<td>0.788558</td>
<td>0.7887</td>
<td>0.000365</td>
</tr>
<tr>
<td>X₂</td>
<td>0.408</td>
<td>0.447</td>
<td>0.409022</td>
<td>0.4082</td>
<td>0.005483</td>
</tr>
<tr>
<td>Objective</td>
<td>263.8958</td>
<td>263.8962</td>
<td>263.8958</td>
<td>263.8958</td>
<td>0.000182</td>
</tr>
</tbody>
</table>

Table 4.4: Statistical information obtained after 50 runs of QSHECT algorithm for three bar truss design problem

**Fig. 4.6: Convergence graph of the objective function of three bar truss design**

4.4.2 Gear Train Design

Gears are machine elements that transmit motion by means of successively engaging teeth. The gear teeth act like small levers. Gear trains consist of driving gears that are attached to the input shaft and driven gears that are attached to the output shaft for the purpose of transmitting motion from one axis to another (Fig. 4.7). These gears work together by meshing their teeth and turning each other in a system to generate power and speed. Gear train reduces speed and increases torque. To create large gear ratio, gears are connected together to form gear trains. The most common of the gear train is the simple gear train with a gear pair connecting parallel shafts having axis, relative to the frame, for all gears comprising the train. Fig. 4.8 shows a simple ordinary train in which two or more gears may rotate about a single axis. The design of a gear train is considered with the face width \( b \), module teeth \( m \), number of teeth on pinion \( z \), length of shaft 1 \( d_1 \), and diameter of shaft 2 \( d_2 \) as design variables \( x_1, x_2, ........., x_7 \), respectively. The constraints include limitations on the bending stress of...
gear teeth, surface stress, transverse deflections of shafts 1 and 2 due to transmitted force, and stresses in shafts 1 and 2 due to transmitted force.

**Fig. 4.7: A gear pair in mesh**

The weight of the gear train is to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shafts. The variables $x_1$, $x_2$, $x_3$, $x_4$, $x_5$, $x_6$, $x_7$ are the face width, module of teeth, number of teeth in the pinion, length of the first shaft between bearings, length of the second shaft between bearings and the diameter of first and second shafts. The details of this single objective problem with 11 behavioral constraints are provided in S. S. Rao [Rao96].

**Fig. 4.8: Gear Pair**
The mathematical expression of the objective function of gear train is given below:

Minimize \( f(x) = 0.7854 x_1 x_2^2 (3.3333 x_3^2 + 14.9334 x_4 - 43.0934) - 1.508 x_1 (x_6^2 + x_7^2) \\
+ 7.4777 (x_3^3 + x_4^3) + 0.7854 (x_4 x_6^2 + x_3 x_7^2) \)

Subject to

\[
27 x_1^{-1} x_2^{-2} x_3^{-1} \leq 1 \\
397.5 x_1^{-1} x_2^{-2} x_3^{-2} \leq 1 \\
1.93 x_3^{-1} x_4^{-1} x_5^{-1} x_6^{-4} \leq 1 \\
1.93 x_2^{-1} x_3^{-1} x_5^{-1} x_7^{-4} \leq 1 \\
x_2 x_3 \leq 40.0 \\
5 \leq \frac{x_1}{x_2} \leq 12.0 \\
(1.5 x_6 + 1.9) x_4^{-1} \leq 1 \\
(1.1 x_7 + 1.9) x_5^{-1} \leq 1 \\
\left[ \frac{(745 x_4 x_2^{-1} x_3^{-1})^2 + 16.9 \times 10^6}{110.0 x_6^3} \right]^{1/2} \leq 1 \\
\left[ \frac{(745 x_4 x_2^{-1} x_3^{-1})^2 + 157.5 \times 10^6}{85.0 x_7^3} \right]^{1/2} \leq 1 \\
\]

where,

\( 2.6 \leq x_1 \leq 3.6 \) \\
\( 0.7 \leq x_2 \leq 0.8 \) \\
\( 17 \leq x_3 \leq 28 \) \\
\( 7.3 \leq x_4 \leq 8.3 \) \\
\( 7.8 \leq x_5 \leq 8.3 \) \\
\( 2.9 \leq x_6 \leq 3.9 \) \\
\( 5.0 \leq x_7 \leq 5.5 \) \\

The table 4.5 shows the results obtained for the gear train design using QSHECT algorithm. The final results have been obtained with initial population size of 10 and offspring size of 100 in each generation for sixty generations as depicted in Fig. 4.9. It can be clearly observed from table 4.5 that the results obtained from QSHECT algorithm are more accurate and consistent than the values reported by Rao [Rao96], Li et al. [Li85], SHGAP [Li85], Kuang et al. [Kua98], Ray et al. [Ray01] and Hans Raj et al. [Hans05]. Table 4.6 shows statistical information obtained after 50 runs of QSHECT algorithm for gear train design problem.
In solving this problem following penalty function is used:

\[ \text{Penalty} = \sum_{i=1}^{10} C_i^2 \]

Where \( C_i \) is the \( i^{th} \) constraints deviation from limits

<table>
<thead>
<tr>
<th>Design Variables</th>
<th>Rao</th>
<th>Li et al.</th>
<th>Kunag et al.</th>
<th>Ray et al.</th>
<th>Hans Raj et al.</th>
<th>QSHECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>3.50</td>
<td>3.6</td>
<td>3.6</td>
<td>3.514185</td>
<td>3.5000</td>
<td>3.5000</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0.70</td>
<td>0.7</td>
<td>0.70</td>
<td>0.700005</td>
<td>0.7000</td>
<td>0.7000</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17.000</td>
<td>17.000</td>
</tr>
<tr>
<td>( x_4 )</td>
<td>7.30</td>
<td>7.2999999</td>
<td>7.3</td>
<td>7.497343</td>
<td>7.3000</td>
<td>7.3000</td>
</tr>
<tr>
<td>( x_5 )</td>
<td>7.30</td>
<td>7.715317</td>
<td>7.8</td>
<td>7.834649</td>
<td>7.8207</td>
<td>7.8000</td>
</tr>
<tr>
<td>( x_6 )</td>
<td>3.35</td>
<td>3.350540</td>
<td>3.4</td>
<td>2.901786</td>
<td>2.9001</td>
<td>2.9002</td>
</tr>
<tr>
<td>( x_7 )</td>
<td>5.29</td>
<td>5.28665</td>
<td>5.0</td>
<td>5.002231</td>
<td>5.0000</td>
<td>5.0002</td>
</tr>
<tr>
<td><strong>Obj. Fun.</strong></td>
<td>2985.22</td>
<td>2994.40</td>
<td>2876.22</td>
<td>2732.90</td>
<td>2724.05</td>
<td>2723.34</td>
</tr>
</tbody>
</table>

Table 4.5: Comparison of results for gear train design problem

<table>
<thead>
<tr>
<th>Variables</th>
<th>Best</th>
<th>Worst</th>
<th>Mean</th>
<th>Median</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>3.5000</td>
<td>3.6000</td>
<td>3.5342</td>
<td>3.5</td>
<td>0.04766</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0.7000</td>
<td>0.8000</td>
<td>0.732</td>
<td>0.7</td>
<td>0.047121</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>17.000</td>
<td>19.0803</td>
<td>17.04172</td>
<td>17.000</td>
<td>0.294183</td>
</tr>
<tr>
<td>( x_4 )</td>
<td>7.3000</td>
<td>8.0784</td>
<td>7.331356</td>
<td>7.3000</td>
<td>0.133187</td>
</tr>
<tr>
<td>( x_5 )</td>
<td>7.8000</td>
<td>8.2765</td>
<td>7.82768</td>
<td>7.8000</td>
<td>0.097557</td>
</tr>
<tr>
<td>( x_6 )</td>
<td>2.9002</td>
<td>3.5483</td>
<td>3.35406</td>
<td>3.3502</td>
<td>0.028128</td>
</tr>
<tr>
<td>( x_7 )</td>
<td>5.0002</td>
<td>5.5000</td>
<td>5.29108</td>
<td>5.2867</td>
<td>0.03018</td>
</tr>
<tr>
<td><strong>Obj. Fun.</strong></td>
<td>2723.34</td>
<td>3234.75</td>
<td>3054.634</td>
<td>2996.48</td>
<td>75.10002</td>
</tr>
</tbody>
</table>

Table 4.6: Statistical information obtained after 50 runs of QSHECT algorithm for gear train design problem

![Fig. 4.9: Convergence graph of the objective function of gear train design](image-url)
4.4.3 Spring Design

Springs are fundamental mechanical components which form the basis of many mechanical systems. A spring is a flexible elastic object which exerts a resisting force when its shape is changed. It is used to store mechanical energy. A spring can be defined to be an elastic member. Most springs are assumed linear and obey the Hook’s law:

\[ P = K \delta \]

Where \( P \) is the applied load, \( \delta \) is the displacement and the \( K \) is the spring constant, which is a function of the material properties of the spring.

![Coil Spring](Fig. 4.10: Coil Spring)

Here a design of a minimum mass spring is considered to carry a given axial load without material failure and while satisfying two performance requirements i.e. the spring must deflect by at least \( \Delta \delta \) and the frequency of surge waves must not be less than \( \omega_o \). The minimization of the mass of a tension/compression spring is considered subject to constraints on minimum deflection, shear stress, surge frequency, and limits on the outside diameter and design variables. The design variables \( x_1, x_2 \) and \( x_3 \) are the mean coil diameter, the wire diameter and the number of active coils respectively (Fig. 4.10). This problem is described in Arora [Aro89] and Belugundu [Bel82]. Optimum design formulation for the spring problem is as follows:
Minimize \( f(x) = (x_1 + 2)x_2^2 \)

Subject to the:

- deflection constraint \( 1 - \left( \frac{(x_1^3 + x_2)}{71785x_2^4} \right) \leq 0 \)
- shear stress constraint \( \frac{4x_1^2 - x_2}{12566(x_2^3 - x_2^4)} + \frac{1}{5108x_2^3} - 1 \leq 0 \)
- surge wave frequency constraint \( \frac{1 - 140.54x_2}{x_1^2x_3} \leq 0 \)
- and the outer diameter constraint \( \frac{x_1 + x_2}{1.5} - 1 \leq 0 \)

The lower bounds and upper bounds on the design variables selected are as follows:

- \( 0.25 \leq x_1 \leq 1.3 \),
- \( 0.05 \leq x_2 \leq 2.0 \) and
- \( 2 \leq x_3 \leq 15 \)

The table 4.7 show the results obtained for the spring design using QSHECT algorithm. The final results have been obtained with initial population size of 15 and offspring size of 100 in each generation for thirty five generations as depicted in fig.4.11. It can be clearly observed from table 4.7 that the results obtained from QSHECT algorithm are more accurate and consistent than the values reported by Arora [Aro89], Belegundu [Bel82], Coello [Coe99], Ray et al. [Ray01] and Hans Raj [Hans05]. Table 4.8 shows statistical information obtained after 50 runs of QSHECT algorithm for spring design problems. In solving this problem following penalty function is used:

\[
Penalty = \sum_{i=1}^{4} C_i^4
\]

where \( C_i \) is the \( i^{th} \) constraints deviation from limits.
In this chapter a Quantum Seeded Hybrid Evolutionary Computational Technique, QSHECT, is developed for solving constrained design optimisation problems and is explained in detail. The technique has been carefully designed with various features that enable it to seek the global optimum rapidly without getting stuck in the local optima. The utility of this technique to solve engineering design problems is amply demonstrated by comparing its performance with previously reported algorithms on various examples reported in literature. It is clear from the examples presented that
QSHECT finds better solutions than the previously known best optimal solutions. The algorithm allows a natural coding of design variables by considering discrete/continuous variables. QSHECT finds a number of solutions as an end results. Multiple optimal solutions can be simultaneously captured with QSHECT. This gives designer more flexibility in optimization problems. The results show a great promise in solving even system level design problems in future with this new technique. This versatile technique is further refined and applied in tandem with Neuro-Fuzzy and Neural Network models for optimising hot extrusion and some manufacturing processes.

***********End of chapter 4***********