

GA is also able to solve the following nonlinear goal programming,

$$\left\{ \begin{array}{l} \text{lexmin } \{d_1^- \vee 0, d_2^- \vee 0, d_3^- \vee 0\} \\ \text{subject to:} \\ 3 - \sqrt{x_1} = d_1^- \\ 4 - \sqrt{x_1 + 2x_2} = d_2^- \\ 5 - \sqrt{x_1 + 2x_2 + 3x_3} = d_3^- \\ x_1^2 + x_2^2 + x_3^2 \leq 100 \\ x_1, x_2, x_3 \geq 0. \end{array} \right.$$

We may encode a solution  $\mathbf{x} = (x_1, x_2, x_3)$  into a chromosome  $V = (v_1, v_2, v_3)$ , and decode the chromosome into the solution in the following way,

$$x_1 = v_1, \quad x_2 = v_2, \quad x_3 = v_3.$$

Since the feasible coding space is contained in the following hypercube

$$\mathcal{V} = \{(v_1, v_2, v_3) \mid 0 \leq v_1 \leq 10, 0 \leq v_2 \leq 10, 0 \leq v_3 \leq 10\},$$

we may take  $v_1 = \mathcal{U}(0, 10)$ ,  $v_2 = \mathcal{U}(0, 10)$ , and  $v_3 = \mathcal{U}(0, 10)$ , and accept it as a chromosome if it is feasible. It is clear that we can make 30 feasible chromosomes in finite times. A run of GA with 2000 generations shows that the optimal solution is

$$(x_1^*, x_2^*, x_3^*) = (9.000, 3.500, 2.597)$$

which satisfies the first two goals, but the last objective is 0.122.