CS-6776 Evolutionary Computation

March 13, 2014

Standard Binary Coding

• Given the upper bound \( U \) and lower bound \( L \), a binary string of length \( n \) is converted to a real number as follows:

\[
\Gamma(a_1a_2\cdots a_n) = L + \frac{U - L}{2^n - 1} \times k
\]

• where \( k \) is the integer value the binary string represents:

\[
k = \sum_{i=0}^{n-1} a_i \times 2^i
\]

Examples

• Rosenbrock function:
• \( L=-30; U=30; n=10; \)
• \( 1111111111 = -30 + \frac{30 - (-30)}{2^{10} - 1} \times 1023 = 30 \)
• \( 0001000011 = -30 + \frac{30 - (-30)}{2^{10} - 1} \times 67 = -26.070381 \)

Gray Code Binary Coding

• Characteristics: an increase of one step in the variable value corresponds to a change of a single bit in the code.

• First convert the gray code to binary code:

\[
\beta_j = \sum_{i=j}^{n} \gamma_i \quad \text{Summation is done by mod 2}
\]

• Convert the binary code to the real value use the format on the previous slide.
Examples

- Rosenbrock function:
- \( L = -30; \quad U = 30; \quad n = 10; \)
- Gray code 1111111111 = binary 1010101010 = 682
  \[ -30 + \frac{30 - (-30)}{2^{10} - 1} \times 682 = 10 \]
- Gray code 0001000111 = binary 0001111010 = 24
  \[ -30 + \frac{30 - (-30)}{2^{10} - 1} \times 24 = 28.592375 \]

Floating Point Binary Coding

- IEEE 754 standard
  \[ a \times 10^b \]
  - \( a \) is mantissa
  - \( b \) is exponent
- The floating-point number is represented in 32 bits. The first bit represents the sign (0 = plus, 1 = minus); the following 8 bits store the biased exponent (largest bias = 127) and the remaining 23 bits represent the fractional part (mantissa).

Example - Continued

- Rosenbrock function:
  - 1 sign bit; 6 mantissa bits; 3 exponent bits
    (only allows to move decimal point to a maximum of 5 places)
  - 1111111111 = -31.5
    - sign bit is 1, so it is negative
    - Exponent bits = 7, so move decimal point 5 (the maximum) bits right: 11111.1 =
      \[ 2^4 + 2^3 + 2^2 + 2^1 + 2^0 + 2^{-1} = 31.5 \]
- 0001000011 = 1
  - sign bit is 0, so it is positive
  - Exponent bits = 3, so move decimal point 3 (the maximum) bits right: 001.000 = 1

Examples - Continued