Representation and evaluation

In the context of Keen, the representation of candidate solutions is fundamental. These solutions are modeled as collections of "genetic material," mirroring the concept of genes and chromosomes in biological systems. This genetic material, depending on its organization and granularity, can represent different dimensions of solutions:

Genetic material	Mathematical construct
Gene	Scalar
Chromosome	Vector
Genotype	Matrix

Table 1: Genetic material and their mathematical equivalent

Given the binary nature of the problem domain in Keen, solutions can be effectively represented as arrays of boolean values, encapsulated by the `BooleanGene` and `BooleanChromosome` classes.

Consider the following Kotlin code snippet for defining a genotype:

```
// Define the size of each chromosome, here set to 50 genes.
private const val CHROMOSOME_SIZE = 50
// Initial probability for each gene to be `true`, set at 15%.
private const val TRUE_RATE = 0.15

// Constructing a genotype with specified characteristics.
val gt = genotypeOf {
  chromosomeOf {
    booleans {
      size = CHROMOSOME_SIZE  // Number of genes in a chromosome.
      trueRate = TRUE_RATE    // Probability of a gene being `true`.
    }
  }
}
```

This code defines a genotype with chromosomes consisting of 50 genes each, where each gene has a 15% chance of being `true`. This setup is particularly useful in problems where the solution space can be binary encoded.

Evaluating the fitness of these genotypes is crucial for guiding the genetic algorithm towards optimal solutions. The fitness function, in this case, is designed to count the number of `TrueGenes` within a genotype:

```
// Function to evaluate the fitness of a genotype.
private fun count(genotype: Genotype<Boolean, BooleanGene>) =
  genotype.flatten()  // Convert the genotype to a list of genes.
          .count { it }  // Count the number of `TrueGenes`.
          .toDouble()  // Convert the count to a Double for compatibility.
```

In this function, the genotype is first flattened to transform its structured genetic material into a linear list of genes, making it easier to apply operations like counting. The number of `TrueGenes` reflects the fitness of the genotype, with higher counts indicating potentially more optimal solutions.

By meticulously crafting the representation of solutions and defining a meaningful fitness function, Keen leverages the principles of genetic algorithms to efficiently navigate the solution space of complex optimization problems.