Fix_2_Genes program overview. Royal Truman

Python program Fix_2_Genes **simulates the probability of fixation and the number of generations required for alleles in two unlinked genes (A and B) in a diploid organism**. The program can be used to test assumptions relevant to evolutionary and young earth creationist models, specifically concerning **initial allele proportions** (pA_i and $pBi_)$ and **selection coefficients** (sA and sB).

Overview and assumptions

- **Wright-Fisher Model**: It assumes **random mating**, where allele frequencies in the next generation are sampled from a binomial distribution based on the current generation's effective population size and allele frequencies.
- **Population Growth**: The population size (N) changes over generations according to the **discrete Beverton-Holt model**, using an intrinsic growth rate (r) and a carrying capacity (K). This allows constant, increasing, or decreasing population sizes.
- **Two Unlinked Genes**: The simulation tracks two separate genes, A and B, located on different chromosomes, meaning they segregate independently. Each gene has two alleles (e.g., A/a and B/b).
- **Selection and Dominance**: The fitness of genotypes for each gene is determined by a **selection coefficient** (s) and a **dominance coefficient** (h). This allows for modeling advantageous, deleterious, or neutral alleles, and different modes of inheritance (recessive, additive, dominant, or over/under-dominance).
- **Fixation/Loss Tracking**: The core goal is to determine the probability of an allele (A, a, B, or b) reaching **fixation** (frequency of 1.0) or **loss** (frequency of 0.0), and the number of generations this process takes. These are determined for A, a, B, and b individually, and also for the final homozygous state (AABB, aaBB, AAbb, or aabb).
- **Multiple Attempts and Repetitions**: For each set of input parameters specified in input_data.txt, the program runs a specified number of attempts (independent simulation runs). Furthermore, each simulation scenario (defined by a line in input_data.txt) is also repeated multiple times using parallelization.
- **Input/Output Handling**:
  - Parameters for each simulation scenario are read from an input_data.txt file, which is validated for correct format and values.
  - Detailed results for each individual simulation repetition are saved to results_data.txt.
  - Averaged results across all repetitions for each parameter set are saved to results_data_avg.txt, providing overall statistics like average fixation probabilities and average generations to fixation, along with their standard deviations.
- **Multiprocessing**: The program utilizes Python's multiprocessing module to run multiple simulation jobs in parallel, significantly speeding up the execution of large numbers of simulations by leveraging all available CPU cores.