Minimizing Inconsistency in Pairwise Comparison Matrices Using Genetic Algorithm

McMaster University
Computing & Software

Atiyeh Sayadi, Baran Shajari
Department of Computing and Software, McMaster University

Introduction

- **Pairwise Comparison:**
  - Simplifies decision-making by evaluating objects in pairs.
  - Results are represented as a comparison matrix.
- **Challenge:**
  - Matrices are based on subjective expert judgments.
  - Inconsistencies can arise, reducing decision reliability.
- **Objective:**
  - Apply a genetic algorithm to minimize inconsistencies.
  - Optimize matrices efficiently while preserving expert input.

Pairwise Comparisons and Consistency

- **Pairwise Comparison:**
  - A method to express the relative importance of objects using a comparison matri $A = [a_{ij}]_{n\times n}$
  - Properties of the matrix:
    - $a_{ij} > 0$ for all i, j.
    - $a_{ii} = 1$ (diagonal entries are 1).
    - $a_{ij} \cdot a_{ji} = 1$ (reciprocal property).

$$A = [a_{ij}]_{n\times n} = \begin{bmatrix} 1 & a_{12} & \dots & a_{1n} \\ \frac{1}{a_{12}} & 1 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{a_{1n}} & \frac{1}{a_{2n}} & \dots & 1 \end{bmatrix}$$

- **Consistency:**
  A matrix is consistent if: $a_{ij} \cdot a_{jk} = a_{ik}, \quad \forall i, j, k$

$$cm_A = \max_{(i,j,k)} \left(\min \left(\left| 1 - \frac{a_{ij}}{a_{ik}.a_{kj}} \right|, \left| 1 - \frac{a_{ik}.a_{kj}}{a_{ij}} \right| \right) \right)$$

Genetic Algorithm Methodology

- **Initialization:**
  - Start with an input matrix A with a size of at least 3×3.
  - Generate 999 offspring by introducing random mutations to the matrix elements.
- **Mutation Process:**
  - Randomly select an element a_{ij} $(i \neq j)$ in the matrix, ensuring $a_{ij} > 1$.
  - Modify a_{ij} by adding a small random value $x \in [-0.5, 0.5]$: $a'_{ij} = a_{ij} + x$
  - Update the corresponding element a_{ij} to satisfy: $a'_{ij} \cdot a'_{ji} = 1$ or $a'_{ji} = \frac{1}{a'_{ij}}$
  - Ensure the updated values remain within the valid range: $\left[\frac{1}{7}, 7\right]$
  - This mutation process introduces variations, allowing the algorithm to explore a wide range of potential solutions.

- **Evaluation:** Each matrix in the population is evaluated using a fitness function, which measures inconsistency.
- **Selection:** The top 50% of matrices with the lowest inconsistency are selected.
- **Elitism:** 20% of the top matrices are randomly chosen and passed directly to the next generation to preserve highquality solutions.
- **Crossover:** The remaining 90% of the next generation are produced by performing crossover on selected parent matrices.
- **Direct Transfer:** Additionally, 10% of the offspring are passed directly to the next generation to maintain valuable characteristics.
- **Mutation:** The rest of the population is filled by mutating matrices after crossover, where only one randomly selected element is changed by adding a number from the interval [−0.5, 0.5].

Result

- **Figure 1:** Rapid fitness improvement is observed for 7×7 matrices in early generations. Progress slows as near-optimal consistency is approached.
- **Figure 2:** 4×4 matrices achieve consistency efficiently, stabilizing in 3–7 generations. This highlights the algorithm's effectiveness for smaller matrices.
- **Figure 3:** For 7×7 matrices, most reach consistency after 49–57 generations. Few achieve consistency in earlier generations, reflecting increased complexity.

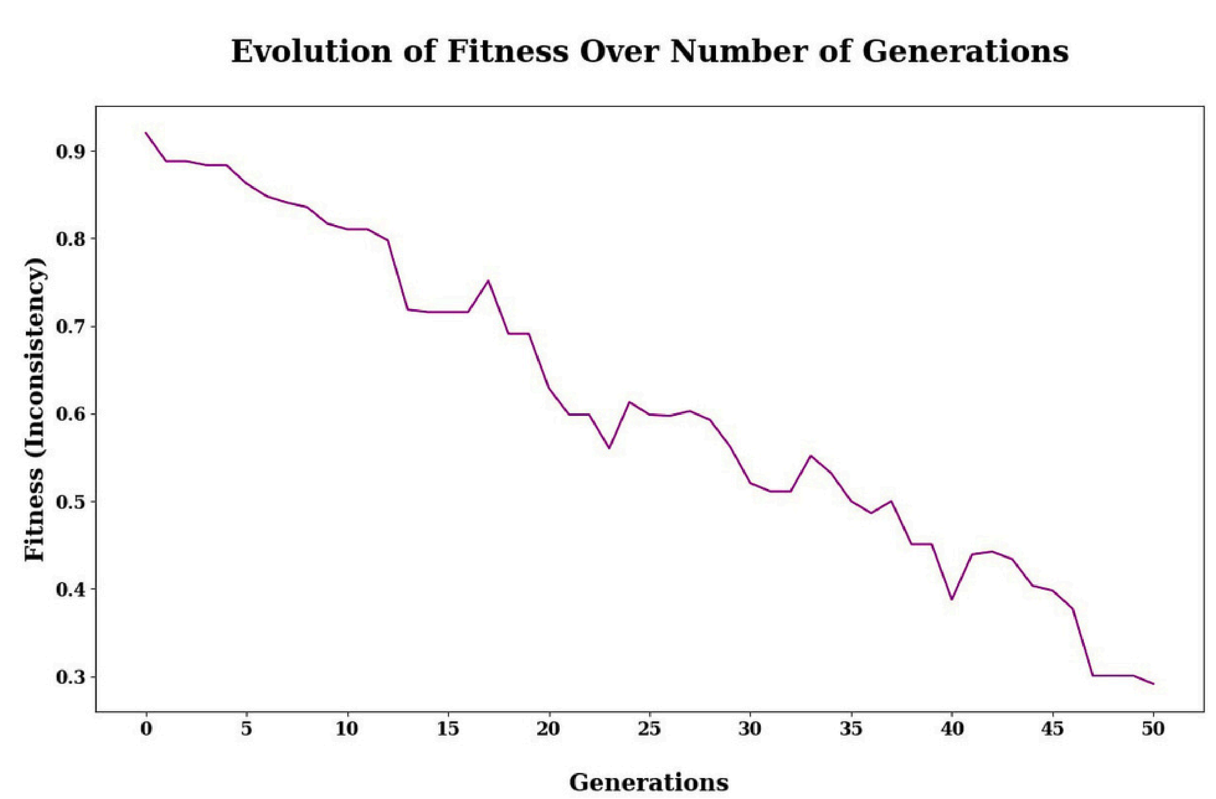

Figure 1

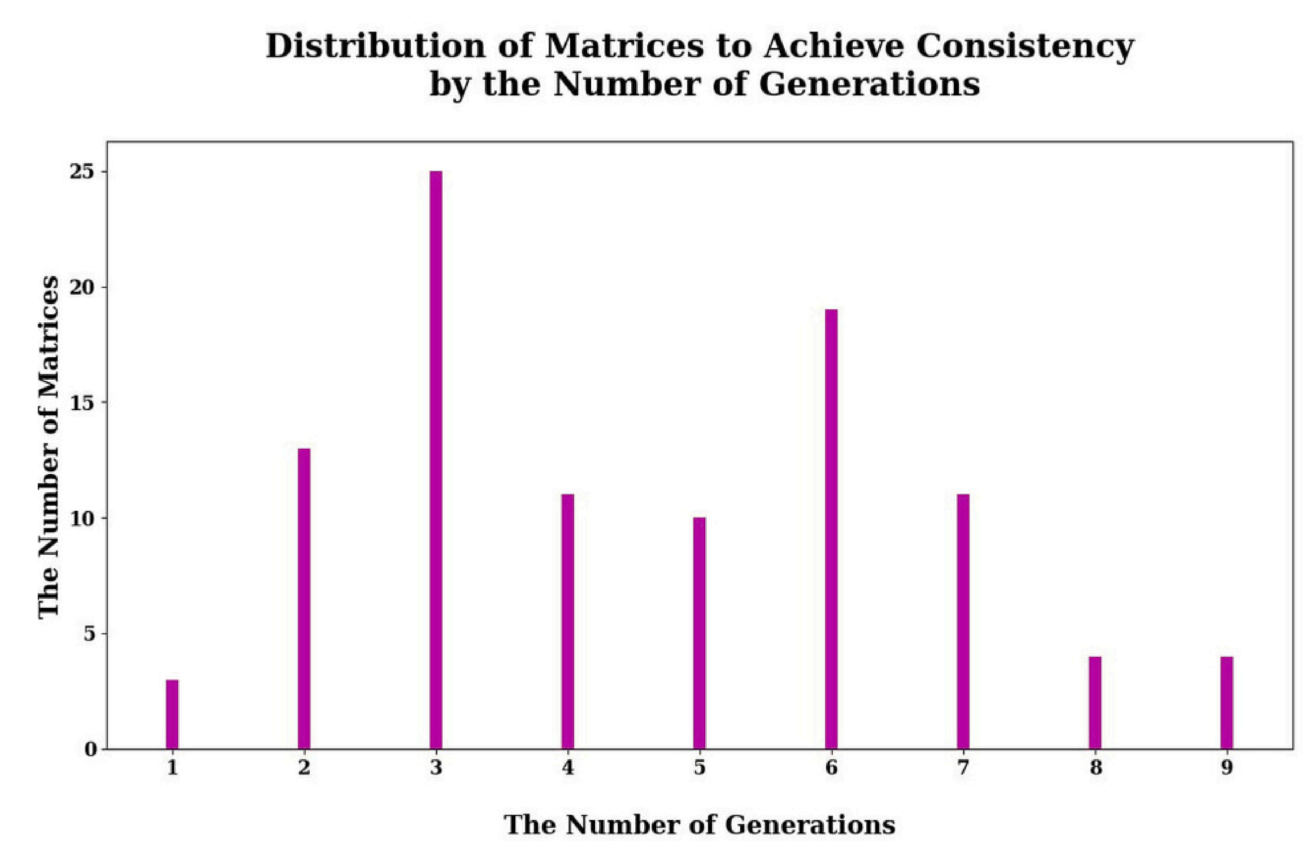

Figure 2

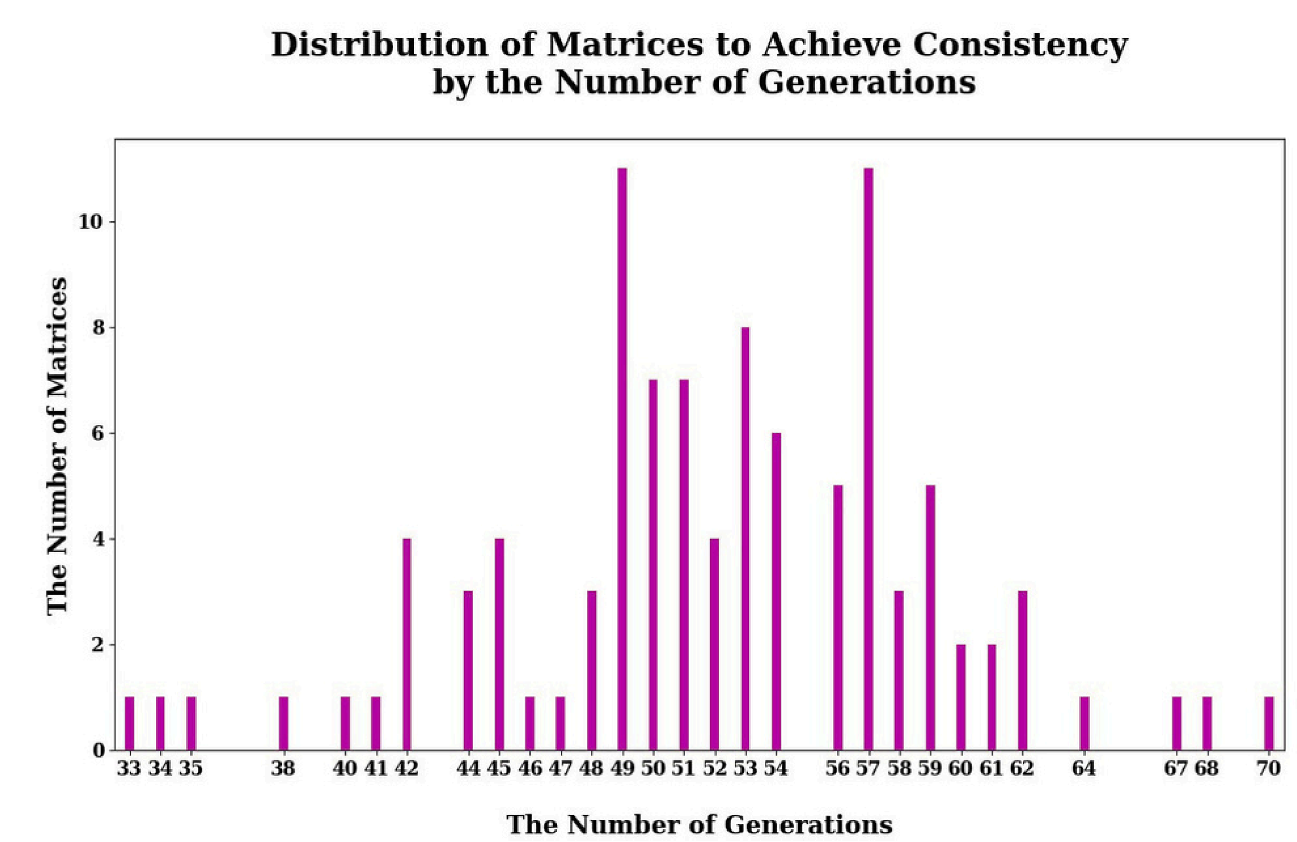

Figure 3