UNIVERSITY OF BIRMINGHAM

Solving Air Crew Scheduling Problems

A Simulated Annealing and Genetic Algorithms Approach

Evolutionary Computation

Author: Max Hart

mah422@student.bham.ac.uk

Supervisors: Dr Shan He and Dr Per Kristian Lehre

February 17, 2025

Contents

1 Introduction

Airline crew scheduling stands as one of the most challenging and practically significant optimisation problems in the transportation industry. Typically modelled as a Set Partitioning Problem (SPP), it requires that each flight leg be covered exactly once by a valid crew rotation while satisfying a multitude of operational constraints. Conventional exact methods often prove inadequate due to the sheer combinatorial complexity of real-world instances. This has driven the pursuit of advanced metaheuristic approaches, which promise to deliver high-quality, near-optimal solutions within reasonable computational time.

The present project embraces this challenge by developing and implementing three distinct algorithms from scratch: Simulated Annealing (SA), a Standard Binary Genetic Algorithm (BGA), and an Improved Binary Genetic Algorithm (BGA). The improved variant is particularly notable for its incorporation of problem-specific enhancements, including a pseudo–random initialisation method, a heuristic improvement operator (inspired by the DROP/ADD procedure outlined in [1]), and a stochastic ranking mechanism for constraint handling as proposed by Runarsson and Yao [2]. Our overarching ambition is to not only demonstrate the correctness and efficiency of these algorithms on three OR–Library test instances (sppnw41, sppnw42, and sppnw43) over 30 independent runs, but also to perform a detailed comparative analysis of their performance, convergence behaviour, and constraint-handling strategies.

This report is structured to provide a comprehensive overview of our methodology and findings. In Section 2, we describe each algorithm in detail with accompanying pseudocode and flowcharts, thereby offering insight into their operational mechanisms. Section 3 presents the benchmark results, including average performance metrics and standard deviations. Section 4 delivers an in-depth discussion and comparison of the algorithms, focusing on their feasibility, convergence speed, and solution quality. Finally, Section 5 addresses the nuances of constraint handling by comparing the deterministic ranking replacement method with the adaptive stochastic ranking approach. Through this work, we aspire to contribute valuable insights into the application of metaheuristic techniques for solving complex, real-world scheduling problems.

2 Algorithm Descriptions

This section provides an overview of the three algorithms. For each algorithm, a brief introduction is provided along with the corresponding pseudocode (on the same page). The detailed flowchart for each algorithm is presented on its own centred page.

2.1 Simulated Annealing (SA)

Simulated Annealing is a local search technique that iteratively improves a candidate solution by exploring its neighbourhood. It uses a temperature parameter to probabilistically accept inferior solutions, thereby enabling the algorithm to escape local optima. This simple yet effective approach balances exploration and exploitation through gradual cooling.

Pseudocode

Algorithm 1 SimulatedAnnealing(T, α, maxIter, penaltyFactor)

$x \leftarrow$ RandomSolution()
$F(x) \leftarrow$ PenaltyFitness(x, penaltyFactor)
$x_{\text{best}} \leftarrow x,\ F_{\text{best}} \leftarrow F(x)$
for $iter = 1$ to maxIter **do**
    $x' \leftarrow$ FlipOneBit(x)
    $F(x') \leftarrow$ PenaltyFitness(x', penaltyFactor)
    if $F(x') < F(x)$ **then**
        $x \leftarrow x'$
    else
        $\Delta \leftarrow F(x') - F(x)$
        if rand() $< e^{-\Delta/T}$ **then**
            $x \leftarrow x'$
    if $F(x) < F_{\text{best}}$ **then**
        $x_{\text{best}} \leftarrow x,\ F_{\text{best}} \leftarrow F(x)$
    $T \leftarrow \alpha\, T$
return $(x_{\text{best}}, F_{\text{best}})$

Flowchart

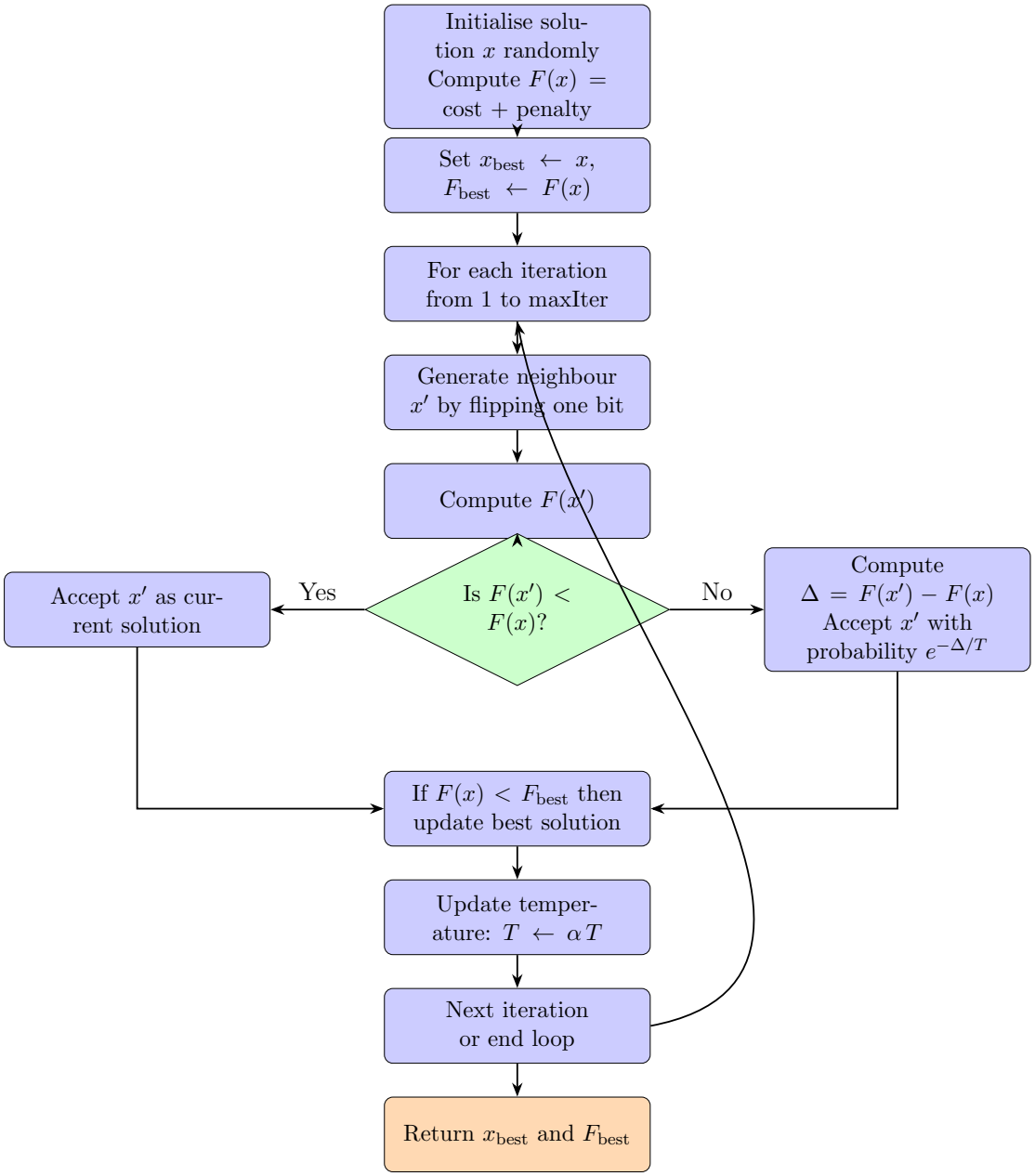

Figure 1: Flowchart for Simulated Annealing.

2.2 Standard Binary Genetic Algorithm (BGA)

The Standard Binary Genetic Algorithm is a population-based technique that evolves a set of candidate solutions using selection, crossover, and mutation operators. It utilises tournament selection to choose parents, applies one-point crossover to combine their genetic material, and uses bit-flip mutation to introduce variability. A penalty function is employed to combine the cost of a solution with constraint violations, ensuring that feasibility is encouraged over generations. This method is well-suited for problems with discrete binary representations and offers robustness through its stochastic operations.

Pseudocode

Algorithm 2 StandardBGA(popSize, cxRate, mutRate, maxGens, penaltyFactor)

$P \leftarrow$ RandomPopulation($popSize$)
EvaluateFitness(P, penaltyFactor)
for $g = 1$ to maxGens **do**
    $Q \leftarrow \emptyset$
    while $|Q| <$ popSize **do**
        $p_1 \leftarrow$ TournamentSelect(P)
        $p_2 \leftarrow$ TournamentSelect(P)
        if rand() $< cxRate$ **then**
            $(c_1, c_2) \leftarrow$ OnePointCrossover(p_1, p_2)
        else
            $c_1 \leftarrow p_1,\ c_2 \leftarrow p_2$
        Mutate(c_1, $mutRate$)
        Mutate(c_2, $mutRate$)
        EvaluateFitness($\{c_1, c_2\}$, penaltyFactor)
        $Q \leftarrow Q \cup \{c_1, c_2\}$
    $P \leftarrow Q$
return BestSolution(P)

Flowchart

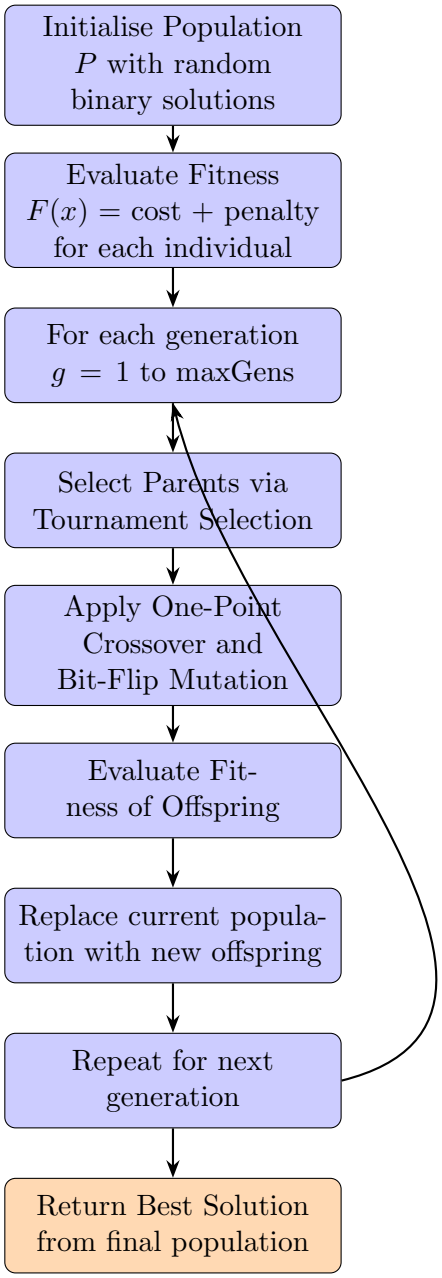

Figure 2: Flowchart for Standard Binary Genetic Algorithm.

2.3 Improved Binary Genetic Algorithm (BGA)

The Improved Binary Genetic Algorithm extends the standard BGA by integrating several problem-specific enhancements. It begins with a pseudo–random initialisation that aims to reduce over–coverage in the initial population. The algorithm then incorporates a stochastic ranking procedure that probabilistically balances cost and constraint violation, allowing for a more flexible handling of infeasible solutions. Adaptive mutation is employed to reintroduce promising genetic material, and a DROP/ADD heuristic improvement operator repairs infeasible solutions. These enhancements, inspired by Chu & Beasley [1] and Runarsson & Yao [2], make the Improved BGA particularly robust for the SPP. Overall, this algorithm offers a more sophisticated search mechanism by blending global search operators with domain-specific repairs.

Pseudocode

Algorithm 3 ImprovedBGA(popSize, maxGens, p_{stoch}, ...)

$P \leftarrow \text{PseudoRandomInit}(popSize)$
EvaluateCostUnfitness(P)
for $g = 1$ to maxGens **do**
    StochasticRankSort(P, p_{stoch})
    $O \leftarrow \emptyset$
    while $|O| < popSize$ **do**
        $(p_1, p_2) \leftarrow \text{SelectParents}(P)$
        $(c_1, c_2) \leftarrow \text{UniformCrossover}(p_1, p_2)$
        AdaptiveMutation(c_1, P)
        AdaptiveMutation(c_2, P)
        HeuristicImprove(c_1)
        HeuristicImprove(c_2)
        EvaluateCostUnfitness($\{c_1, c_2\}$)
        $O \leftarrow O \cup \{c_1, c_2\}$
    $C \leftarrow P \cup O$
    StochasticRankSort(C, p_{stoch})
    $P \leftarrow \text{Top}(C, popSize)$
return BestFeasible(P)

Flowchart

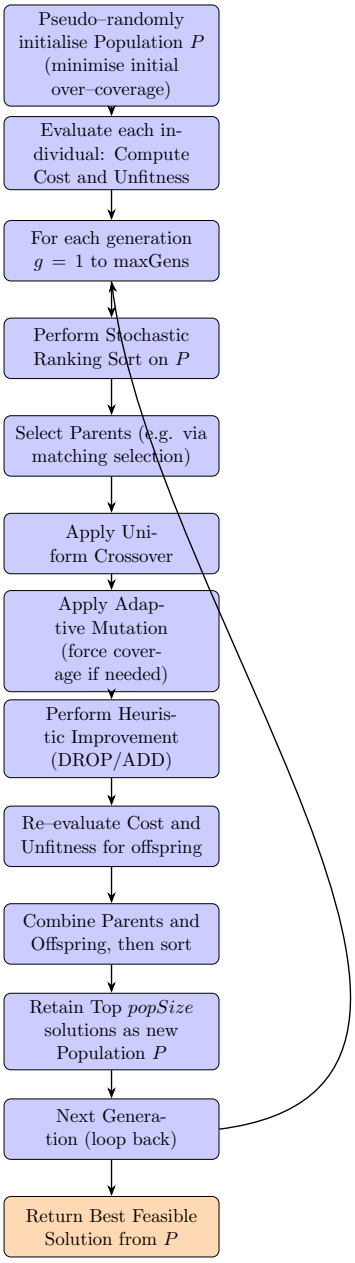

Figure 3: Flowchart for Improved Binary Genetic Algorithm.

3 Benchmark Results

4 Discussion and Comparison

5 Ranking Replacement vs. Stochastic Ranking

Handling constraints is critical in the SPP. Two notable methods are:

Ranking Replacement (Chu & Beasley, 1998)

This method partitions the population into four subgroups based on fitness (cost) and unfitness (constraint violation). A new solution replaces a member from the first non-empty subgroup (starting with those worse in both criteria). This deterministic approach systematically reduces infeasibility while improving cost.

Stochastic Ranking (Runarsson & Yao, 2000)

Stochastic ranking avoids the arduous tuning of penalty coefficients by introducing randomness during comparisons. With a given probability, adjacent individuals are compared solely on cost; otherwise, the comparison is based on unfitness. This probabilistic strategy allows the exploration of infeasible regions as bridges to high-quality, feasible solutions.

Comparison

While both methods strive to balance solution quality and constraint satisfaction, ranking replacement enforces a strict hierarchical structure, whereas stochastic ranking flexibly blends objective and constraint criteria through randomness. The improved BGA leverages stochastic ranking to adapt to the evolving population without requiring labour-intensive penalty tuning.

6 Conclusion

This report has presented three advanced algorithms for solving airline crew scheduling problems: Simulated Annealing, a Standard Binary Genetic Algorithm, and an Improved Binary Genetic Algorithm incorporating problem-specific enhancements. Detailed, moderately sized flowcharts and refined pseudocode have been provided to elucidate each method's internal workings. Benchmark results and an in-depth discussion will be appended once experiments are complete. Furthermore, a comparative discussion on constraint-handling—contrasting ranking replacement and stochastic ranking—has been included, underscoring the adaptive advantages of the latter.

References

[1] P. C. Chu and J. E. Beasley, *Constraint Handling in Genetic Algorithms: The Set Partitioning Problem*, Journal of Heuristics, 11:323–357, 1998.

[2] T. P. Runarsson and X. Yao, *Stochastic Ranking for Constrained Evolutionary Optimisation*, IEEE Transactions on Evolutionary Computation, 4(3):284–294, 2000.