**Genetic Algorithm for Resource Allocation Problem**

A manufacturing company wants to use a genetic algorithm to optimize the allocation of 15 different production tasks across 8 different manufacturing cells, minimizing total production time while balancing workload across cells.

a) Design and explain an appropriate chromosome representation for this resource allocation problem. Your answer should include how the tasks would be encoded and what a valid chromosome would look like.

For the resource allocation problem of assigning 15 tasks to 8 manufacturing cells, an appropriate chromosome representation would be:

A chromosome of length 15, where each gene position corresponds to a specific task (task 1 through task 15), and the value at each position represents the manufacturing cell (1-8) to which that task is assigned.

For example, a valid chromosome might look like:

[3, 1, 6, 2, 8, 4, 7, 5, 2, 6, 3, 1, 8, 4, 5]

In this representation:

* ﻿﻿Task 1 is assigned to manufacturing cell 3
* ﻿﻿Task 2 is assigned to manufacturing cell 1
* ﻿﻿Task 3 is assigned to manufacturing cell 6
* ﻿﻿And so on...

This encoding is appropriate because:

1.It has a fixed length equal to the number of tasks (15)

2. ﻿﻿﻿Each gene directly maps to a specific task, maintaining a one-to-one correspondence

3. The value domain for each gene is constrained to integers 1-8, representing the

available manufacturing cells

4. The representation ensures every task is assigned exactly once to a manufacturing cell

5. The representation allows multiple tasks to be assigned to the same cell, which is realistic in a manufacturing environment.

b) In a genetic algorithm solving this 15-task resource allocation problem, explain the validation rules that determine if a chromosome is a valid solution. Your answer should include three specific constraints that must be satisfied.

For the 15-task resource allocation problem, a chromosome must satisfy these three key constraints to be considered valid:

1. ﻿﻿﻿Completeness constraint: All 15 tasks must be assigned to a manufacturing cell. The chromosome must have exactly 15 genes, with each position corresponding to one task, ensuring no tasks are left unassigned.
2. ﻿﻿﻿Domain constraint: Each gene value must represent a valid manufacturing cell (integers from 1 to 8). Any value outside this range would repr@sent an invalid assignment, as manufacturing cells 1-8 are the only available resources.
3. Capacity constraint: No manufacturing cell should be assigned more tasks than its maximum capacity. If, for example, each cell can handle at most 3 tasks simultaneously due to resource limitations, then no more than 3 positions in the shromosome should contain the same cell number.

These validation rules ensure that solutions are feasible within the physical and operational constraints of the manufacturing environment.

c) Describe two suitable crossover methods specifically designed for resource allocation problems. Explain how each method works and why they are appropriate for this task allocation problem.

1. Uniform Crossover with Repair Mechanism:

Uniform crossover with repair works as follows for the resource allocation problem:

* ﻿﻿For each gene position (task), randomly select which parent contributes its assignment value
* ﻿﻿Create offspring by combining these selections
* ﻿﻿Apply a repair mechanism if the resulting chromosome violates workload  
  balance constraints

For example, if Parent 1 is [3, 1,6,2,8,4,7,5,2,6,3, 1,8,4,5] and Parent 2 is [1,5,3,7,2,8,4,6,5, 1,2,3,4,7,8], a uniform crossover might initially produce:

[3,5,6,7,8,8,7,5,5, 1,3, 1,4,7,5]

If this creates imbalance (e.g., cell 5 now has too many tasks), the repair mechanism would reallocate some tasks from overloaded cells to underutilized cells while minimizing disruption to the crossover result.

This approach is appropriate because:

• It maintains the direct mapping between tasks and assignments

• The repair mechanism ensures workload balance constraints are satisfied

• It effectively combines beneficial assignment patterns from both parents

2. Two-Point Crossover with Feasibility Preservation:

Two-point crossover with feasipility preservation works as follows:

* ﻿﻿Randomly select two crossover points in the chromosome
* ﻿﻿Copy the segment between these points from one parent
* ﻿﻿Fill the remaining positions from the other parent while preserving feasibility

For example, with crossover peints after positions 5 and 10:

• From Parent 1[3, 1,6,2,8,4,7,5,2,6,3, 1,8,4,5]

* ﻿﻿From Parent 2 1,5,3, 7,2,8,4,6,5, 1,2,3,4,7,81
* ﻿﻿The offspring would get: [1,5,3, 7,2,4,7,5,2,6,2,3,4,7,8]

If this creates workload imbalance, minor adjustments are made to redistribute tasks

White maintaining most of the inherited pattern.

This method is appropriate because:

* ﻿﻿It preserves contiguous assignments from parents, which often represent efficient subtask groupings
* ﻿﻿It maintains the workload balancing characteristic by adjusting only when  
  necessary

• It allows effective exploration of the solution space by combining large solution

segments

d) Describe two mutation operators that could be used for this resource allocation problem. Explain how each would work and what benefit they provide to the search process.

1. Swap Mutation:

Swap mutation for the resource allocation problem works as follows:

* ﻿﻿Randomly select two positions (tasks) in the chromosome
* ﻿﻿Exchange the manufacturing cell assignments between these two tasks

For example, if the original chromosome is:

[3, 1,6,2,8,4,7,5,2,6,3, 1,8,4,5]

And positions 3 and 9 are selected, the result after mutation would be:

[3, 1,2,2,8,4,7,5,6,6,3, 1,8,4,5]

This means Task 3 is now assigned to Cell 2 and Task 9 is now assigned to Cell 6.

Benefits:

• Maintains the overall workload distribution across cells (same number of tasks

per cell)

Introduces small, controlled changes that allow for fine-tuning of solutions Helps the algorithm escape local optima by exploring neighboring solutions

2. Cell Reassignment Mutation:

Cell reassignment mutation works as follows:

* ﻿﻿Randomly select a task (position in the chromosome)
* ﻿﻿Randomly select a new manufacturing cell different from the current assignment

Change the tasks assignment to the new cell

For example, if the original chromosome is:

[3,1,6,2,8,4,7,5,2,6,3,1,8,4,5]

And position 7 is selected with a new random cell value of 2, the result would be:

[3, 1, 6,2,8,4,2,5,2,6,3,1,8,4,5]

This means Task 7 is now assigned to Cell 2 instead of Cell 7.

Benefits:

• Allows exploration of different workload distributions across manufacturing cells

\*Helps balance the workload when certain cells become overloaded

• Provides diversity in the population by introducing new assignment

combinations

• Particularly effective when certain cells have specialized capabilities for specific

e) Explain how you would design a fitness function for this resource allocation problem. Include what it would measure and how it would evaluate solution quality.

A suitable fitness function for the manufacturing task allocation problem would:

1. Primary objective - Minimize total production time: Calculate the sum of processing times for all tasks based on their cell assignments. Each task may have different processing times in different cells based on the cell's

specialization. For example: Fitness = 1 / (E (\*sum symbol)

processingTime|task il[cell assigned to task i])

1. ﻿﻿﻿Secondary objective - Balance workload across cells: Calculate the standard deviation of the workload across all cells and penalize solutions with high  
   imbalance. Incorporate a penalty factor: Fitness = Fitness \* (1 - a \*  
   StandardDeviation(cell workloads)) Where a is a weighting factor determining the importance of load balancing.
2. ﻿﻿﻿Handling constraints: Apply penalty for any violated constraints such as  
   exceeding cell capacity. Final Fitness = Fitness - (B \* number of violations)  
   Where B is a penalty coefficient.

