This is a cooperative game theory problem involving container terminals. The goal is to find a way for the terminals to cooperate to increase their collective and individual profits.

**Problem Description**

This problem involves several container terminals, each with unique cost and revenue structures. The key features are:

* **Marginal Cost (MC):** The cost to handle one additional container. It initially decreases as a terminal gains efficiency from a higher volume (economies of scale), reaches a minimum at a certain **Volume of Containers (VOC)**, and then increases due to congestion and resource constraints (diseconomies of scale).
* **Per-Container Charge:** The price a terminal charges per container. This charge decreases as volume increases, a strategy often used to attract more business, until it reaches a minimum and remains constant.
* **Profit Function:** A terminal's profit, which is total revenue minus total cost, initially increases with volume, reaches a maximum at a specific VOC, and then decreases.
* **Cooperation:** Terminals on the "right side" of their maximum profit VOC, meaning they are experiencing decreasing returns, can transfer vessels to terminals on the "left side" of their maximum profit VOC, where handling more containers would increase their profit. The transferring terminal receives compensation to make the transfer beneficial for both parties. The goal is for all terminals to increase their profits compared to their non-cooperative state.

**Conditions for Feasible Cooperation**

For cooperation to be feasible, the following marginal conditions must be met to ensure all terminals gain more profit than they would have without cooperation.

**Marginal Cost and Revenue Conditions**

The core principle is that a vessel should be handled by the terminal that can do so at the lowest **marginal cost**. A terminal willing to transfer a vessel must be able to compensate the receiving terminal while still retaining a net gain.

* **For the transferring terminal (Terminal A):** Its marginal cost (MCA​) is increasing, and it's operating beyond its optimal VOC. By transferring a vessel, its volume decreases, moving it closer to its optimal VOC and reducing its total cost. The compensation it pays must be less than the cost savings achieved from reducing its volume.
* **For the receiving terminal (Terminal B):** Its marginal cost (MCB​) is decreasing, and it's operating below its optimal VOC. By accepting a vessel, its volume increases, moving it closer to its optimal VOC and further decreasing its marginal cost. The compensation it receives must be greater than or equal to its marginal cost of handling the additional container (MCB​).

For cooperation to be profitable for both:

MCB​<Compensation<Cost Savings of Terminal A

In an idealized setting, for the system to maximize its collective profit, vessels should be redirected from terminals with a high marginal cost to terminals with a low marginal cost. This continues until the marginal costs across all cooperating terminals are equal. This is the condition for **allocative efficiency**.

**Profit Conditions**

The ultimate measure of feasibility is a positive change in profit for all parties. Let Πibefore​ be the profit of terminal i before cooperation and Πiafter​ be its profit after cooperation.

* **For all terminals i∈{1,2,...,n}:** Πiafter​>Πibefore​.

The total profit of the system must increase, and the compensation mechanism must distribute this increase in a way that makes all terminals better off. The transfer of a container from Terminal A to Terminal B is profitable for the system as a whole if the marginal revenue of that container is greater than the collective marginal cost.

The compensation structure is crucial and must be designed to satisfy the following:

1. **Individual Rationality:** Each terminal's profit after cooperation must be greater than its profit before cooperation.
2. **Collective Rationality:** The sum of all terminals' profits after cooperation must be greater than the sum of their profits before cooperation.

Here's the text formatted using Markdown that's suitable for conversion to a Word document via Pandoc. I've used standard Markdown conventions, avoiding complex features that might not translate well.

**Cooperative Game Among Container Terminals**

This document outlines a cooperative game theory problem involving container terminals. The goal is to maximize collective profit by reallocating vessels from less-efficient terminals to more-efficient ones, ensuring all terminals benefit.

**Problem Description**

Let's define the key functions for each terminal i:

* Vi​: Volume of containers handled by terminal i.
* Ci​(Vi​): Total cost function for terminal i.
* Ri​(Vi​): Total revenue function for terminal i.
* Πi​(Vi​)=Ri​(Vi​)−Ci​(Vi​): Profit function for terminal i.

The marginal cost for terminal i is the derivative of its total cost function:

MCi​(Vi​)=dVi​dCi​(Vi​)​

The problem states that MCi​(Vi​) decreases up to a certain volume (VOC) and then increases. This implies a U-shaped marginal cost curve.

The marginal revenue for terminal i is the derivative of its total revenue function:

MRi​(Vi​)=dVi​dRi​(Vi​)​

The profit is maximized when marginal revenue equals marginal cost:

dVi​dΠi​(Vi​)​=MRi​(Vi​)−MCi​(Vi​)=0

Let's denote the profit-maximizing volume for terminal i as Vi∗​. Terminals on the "left side" of the profit function are operating at a volume Vi​<Vi∗​, meaning their marginal profit is positive (MRi​>MCi​). Terminals on the "right side" are operating at a volume Vi​>Vi∗​, meaning their marginal profit is negative (MRi​<MCi​).

**Conditions for Feasible Cooperation**

For cooperation to be feasible, a redistribution of containers must lead to a Pareto improvement, where no terminal is worse off and at least one is better off. This requires meeting the following marginal cost, revenue, and profit conditions.

**Marginal Cost and Revenue Conditions**

Let's consider a transfer of a single container from a transferring terminal, Terminal A, to a receiving terminal, Terminal B.

* **Terminal A (transferring):** This terminal is on the right side of its optimal volume, VA​>VA∗​. Its marginal cost is increasing, MCA​(VA​) is high. By transferring one container, its volume decreases, moving it towards its optimal point. The savings in total cost for Terminal A is the marginal cost it would have incurred, MCA​(VA​).
* **Terminal B (receiving):** This terminal is on the left side of its optimal volume, VB​<VB∗​. Its marginal cost is decreasing, MCB​(VB​) is low. By receiving one container, its volume increases, further reducing its marginal cost. The cost incurred by Terminal B is its current marginal cost, MCB​(VB​).

For cooperation to be collectively profitable, the total cost for handling a container should decrease. This happens if the marginal cost of the receiving terminal is less than the marginal cost of the transferring terminal:

MCB​(VB​)<MCA​(VA​)

The compensation (K) paid from Terminal A to Terminal B must satisfy the following condition to make the transfer individually rational for both:

MCB​(VB​)≤K≤MCA​(VA​)

* If K<MCB​(VB​), Terminal B is not compensated enough for its costs.
* If K>MCA​(VA​), Terminal A pays more than it saves.

The system-wide optimal allocation of containers is achieved when the marginal cost across all cooperating terminals is equalized. This is the equimarginal principle:

MC1​(V1′​)=MC2​(V2′​)=⋯=MCn​(Vn′​)

where Vi′​ is the new volume for terminal i after cooperation.

**Profit Conditions**

Let the total profits of the system before and after cooperation be Πtotalbefore​ and Πtotalafter​.

Πtotalbefore​=i=1∑n​Πibefore​andΠtotalafter​=i=1∑n​Πiafter​

The condition for collective feasibility is that the total profit of the system increases:

Πtotalafter​>Πtotalbefore​

For cooperation to be individually rational, the profit of each terminal must increase or remain constant:

Πiafter​≥Πibefore​for all terminals i

This is guaranteed by a well-designed compensation scheme. For a single transfer, the change in profit for Terminal A (ΔΠA​) and Terminal B (ΔΠB​) must be positive.

ΔΠA​=Cost Savings−K>0⟹K<MCA​(VA​)

ΔΠB​=K−Cost Incurred>0⟹K>MCB​(VB​)

These inequalities directly lead back to the condition MCB​<K<MCA​, ensuring that both parties benefit from the transaction. The total gain from the transfer is the difference in marginal costs:

Total Gain=MCA​(VA​)−MCB​(VB​)

This gain can then be divided between the two terminals via the compensation K, making cooperation a win-win situation.

Of course. As a master of optimization, I understand that true mastery lies in the details. Here is a more comprehensive and highly detailed breakdown of the work, delving into the specific mathematical nuances and providing concrete examples for each stage of the memetic algorithm.

**1. Mathematical Formulation: An Exhaustive Analysis**

The foundation of our work is a non-linear, constrained optimization problem. The primary objective is to maximize the total collective profit of n cooperating container terminals.

**A. The Individual Terminal Profit Function**

Each terminal i has a unique profit function, Πi​(Vi​), which is the difference between its total revenue, Ri​(Vi​), and its total cost, Ci​(Vi​).

Πi​(Vi​)=Ri​(Vi​)−Ci​(Vi​)

Our **OptimizedTerminalModel** uses a more realistic, piecewise-linear marginal cost function, which is integrated to derive the total cost. This is a crucial improvement over simple cubic functions, as it accurately models the economies and diseconomies of scale.

* **Revenue Function**: Our model uses a linear revenue function, R(V)=p⋅V, where p is the fixed price per container. This simplifies the model but can be extended to a more complex demand curve where price decreases with volume.
* **Cost Function**: The total cost is a piece-wise polynomial. The marginal cost, MC(V), is linear in two segments based on the terminal's optimal utilization, uoptimal​.

MC(V)={MCstart​−slope1​⋅uMCmin​+slope2​⋅(u−uoptimal​)​if u≤uoptimal​if u>uoptimal​​

The total cost, C(V), is the integral of this marginal cost.

**B. The Optimization Problem and KKT Conditions**

The collective optimization problem is to find the set of volumes (V1​,V2​,…,Vn​) that maximizes the total system profit, Πtotal​.

maxΠtotal​=i=1∑n​Πi​(Vi​)

Subject to the constraints:

1. **Volume conservation**: ∑i=1n​Vi​=Vtotal​
2. **Non-negativity**: Vi​≥0∀i
3. **Individual Rationality**: Πi​(Vi​)≥Πinon−coop​∀i

The **Individual Rationality** constraint is the most complex. It is handled by embedding it directly into the optimization problem as a **penalty term** in the fitness function. This transforms the problem from a simple maximization to a search for a viable, stable solution within the feasible set of profitable outcomes for all terminals.

**2. The Memetic Algorithm: A Step-by-Step Breakdown**

Our memetic algorithm is a sophisticated heuristic that combines a global search for the best coalition with a powerful local search for fine-tuning.

**A. The Individual (Chromosome)**

An individual is a vector representing a potential volume allocation.

* **Example**: For a system with 3 terminals and a total volume of 1,000 containers, an individual might be [300, 450, 250]. The sum is 1,000, and all values are non-negative.

**B. The Fitness Function: The Heart of the Optimization**

The fitness function guides the entire evolutionary process. It is a composite function designed to reward collective profit while severely punishing violations of economic rationality.

Fitness(V)=(i=1∑n​Πi​(Vi​))−Rationality PenaltyM⋅i=1∑n​max(0,Πinon−coop​−Πi​(Vi​))​​−Non-Negative Profit Penalty10M⋅i=1∑n​max(0,−Πi​(Vi​))​​

* **Collective Profit**: The first term is the gross total profit.
* **Rationality Penalty**: The second term is a penalty for any terminal whose cooperative profit is less than its non-cooperative baseline. This is the **individual rationality** condition.
* **Profit Loss Penalty**: The third term is an even heavier penalty for any terminal operating at a negative profit, ensuring the algorithm avoids non-viable solutions.

**C. The Genetic Operators: Global Exploration**

These operators are the engine of the genetic algorithm, allowing it to explore a vast solution space.

* **Selection (Tournament Selection)**: We select the best individuals to be parents. In tournament selection, a small random subset of the population is chosen, and the individual with the highest fitness within that subset is selected. This balances exploration with exploitation.
* **Crossover (Arithmetic Crossover)**: Two parents, Vp​ and Vq​, are blended to create an offspring Voffspring​.
  + **Example**: Let Vp​=[300,450,250] and Vq​=[400,300,300]. With a random blending factor of α=0.6, the offspring is:Voffspring​=0.6⋅[300,450,250]+0.4⋅[400,300,300]=[340,390,270]. The new vector is then normalized to ensure its sum is 1,000.
* **Mutation (Dynamic Exchange)**: We introduce random perturbations. Instead of a simple random change, we randomly select two terminals and shift a small volume between them. This maintains the total volume constraint while exploring new points. The mutation rate is **adaptive**, increasing when the population is stagnant to encourage diversity.

**D. The Local Search: Targeted Refinement**

This is the "memetic" component, a powerful local optimizer that polishes newly created individuals. It applies the KKT equimarginal principle to fine-tune a solution.

* **Mechanism**: The local search iteratively transfers small amounts of volume from terminals with low marginal profit to those with high marginal profit. This continues until the marginal profits across all terminals are equalized, indicating a local optimum has been reached.

**3. The Grand Coalition and Sub-Coalition Search**

The final piece of the puzzle is the search for a stable coalition structure. The code does not simply assume all terminals will cooperate. Instead, it systematically evaluates the profitability of sub-coalitions to identify the most stable arrangement.

* **The Grand Coalition**: The algorithm first finds the optimal solution for all terminals cooperating. The total profit is a benchmark for all other arrangements.
* **The Sub-Coalition Search**: The code then explores the space of potential sub-coalitions, specifically focusing on partitions of the grand coalition into two smaller groups. For each partition, it runs the memetic algorithm on the individual sub-coalitions and sums their total profits.
* **Stability Condition**: A stable partition is one where the sum of the profits of all sub-coalitions is greater than the grand coalition's profit. This indicates that the grand coalition is not stable and that a more profitable arrangement exists for its members.

I can provide even more detail. This breakdown will cover the fine-grained mathematical and algorithmic aspects of the solution, including the specific forms of the functions and the step-by-step mechanics of the memetic algorithm, providing a level of detail that a true master would expect.

**1. Mathematical Formulation: The Full Scope**

The mathematical model is a **non-linear programming problem**. Its complexity arises from the cubic and piecewise nature of the cost functions.

**A. The Profit Function**

The profit function for each terminal is a complex blend of revenue and cost. Our OptimizedTerminalModel uses specific forms for these functions that are integrated to maintain consistency.

* Total Revenue (R(V)): The revenue function is a simple linear form with a price component.

R(V)=p⋅V where p is the price per container. For our OptimizedTerminalModel, this is derived from the revenue function's form: R(u)=(a⋅u+b)⋅C, where u is utilization. This translates to R(V)=(aCV​+b)V. The marginal revenue is a linear function of volume, MR(V)=C2a​V+b.

* **Total Cost** (C(V)): The cost function's complexity stems from its relationship to the marginal cost, which is a key economic indicator. The marginal cost curve is U-shaped, reflecting economies of scale at low volumes and diseconomies of scale at high volumes.

MC(V)={MCstart​−slope1​⋅CV​MCmin​+slope2​⋅(CV​−uoptimal​)​if CV​≤uoptimal​if CV​>uoptimal​​

The total cost C(V) is the integral of this marginal cost curve with respect to volume. This ensures economic consistency between the marginal and total cost functions.

**B. The KKT Conditions and the Equimarginal Principle**

The KKT conditions are the foundation for the local search component. The core insight is that for a solution to be optimal, the marginal profit of all active terminals must be equal to the shadow price of the total volume constraint, λ.

MRi​(Vi​)−MCi​(Vi​)=λ∀i∈{1,…,n}

This forms a system of equations that defines the optimal allocation. The memetic algorithm's local search is a heuristic solver for this system, iteratively moving volume from terminals with low marginal profit to those with high marginal profit, thereby driving them toward a common λ.

**2. The Memetic Algorithm: A Grandmaster's Approach**

Our memetic algorithm is a highly refined tool that meticulously combines global and local optimization.

**A. Chromosome Representation and Population Dynamics**

* **Chromosome**: A solution is represented as a vector of volumes, V=(V1​,V2​,…,Vn​). The sum of the components must equal Vtotal​.
* **Dynamic Population Size**: Unlike static algorithms, our solution adapts. If the algorithm detects a stagnation in fitness improvement, it dynamically increases the population size to broaden its search and escape local optima. Once progress resumes, the population size reverts to its initial value.

**B. Fitness Function: A Multi-layered Objective**

The fitness function is a carefully constructed measure of a solution's quality that ensures economic viability.

Fitness(V)=(i=1∑n​Πi​(Vi​))−i=1∑n​[penalty1​+penalty2​]

* **Individual Rationality Penalty**: The first penalty, penalty1​, is proportional to the profit deficit.

penalty1​=M⋅max(0,Πinon−coop​−Πi​(Vi​))

This is a sharp, linear penalty that makes any solution violating individual rationality highly undesirable.

* **Viability Penalty**: The second penalty, penalty2​, is for negative profits.

penalty2​=10M⋅max(0,−Πi​(Vi​))

This penalty is an order of magnitude larger to force the algorithm to seek only solutions where all terminals are financially solvent.

**C. The Local Search: A Precision Tool**

The local search is the most critical part of the algorithm's performance. It works as a precision tool to refine a raw solution.

1. **Marginal Profit Calculation**: For a given solution, the marginal profit (MPi​=MRi​(Vi​)−MCi​(Vi​)) is calculated for each terminal.
2. **Equimarginal Drive**: The algorithm identifies the average marginal profit (MPˉ) and iteratively adjusts volumes based on the difference, (MPi​−MPˉ).
   * Terminals with a marginal profit greater than the average (MPi​>MPˉ) are deemed "under-utilized" and receive more volume.
   * Terminals with a marginal profit less than the average (MPi​<MPˉ) are "over-utilized" and have their volume reduced.
3. **Proportional Adjustment**: The volume adjustment is proportional to the marginal profit difference, ensuring that the greatest changes are made where the greatest inefficiencies exist.

ΔVi​∝−(MPi​−MPˉ)

This is a targeted hill-climbing process that rapidly converges to a local optimum.

**3. The Grand Coalition and Sub-Coalition Search**

The solution moves beyond a simple centralized model to address the fundamental game theory question of stability.

* **Stability Condition**: A coalition is stable if no member can improve its profit by leaving. Our code searches for a partition of the terminals that is "stable," meaning no sub-coalition could make more profit on its own than by participating in the larger group.
* **Recursive Search**: The CooperativeProfitMaximizer employs a recursive search. It evaluates the profitability of the grand coalition. If it is not the most profitable arrangement (e.g., a sub-coalition could earn more together), the search continues to explore partitions until a stable set of coalitions is found. The use of memoization ensures that the profit of any given sub-coalition is only calculated once, dramatically improving efficiency.

**1. The Mathematical Formulation: An Exhaustive Analysis**

The mathematical model is a **non-linear programming problem**. Its complexity arises from the cubic and piecewise nature of the cost functions.

**A. The Profit Function**

The profit function for each terminal is a complex blend of revenue and cost. Our OptimizedTerminalModel uses specific forms for these functions that are integrated to maintain consistency.

* **Total Revenue** (R(V)): The revenue function's form is derived from the revenue function's form: R(u)=(a⋅u+b)⋅C, where u is utilization. This translates to R(V)=(aCV​+b)V. The marginal revenue is a linear function of volume, MR(V)=C2a​V+b.
* **Total Cost** (C(V)): The cost function's complexity stems from its relationship to the marginal cost, which is a key economic indicator. The marginal cost curve is U-shaped, reflecting economies of scale at low volumes and diseconomies of scale at high volumes.

MC(V)={MCstart​−slope1​⋅CV​MCmin​+slope2​⋅(CV​−uoptimal​)​if CV​≤uoptimal​if CV​>uoptimal​​

The total cost C(V) is the integral of this marginal cost curve with respect to volume. This ensures economic consistency between the marginal and total cost functions.

**B. The KKT Conditions and the Equimarginal Principle**

The KKT conditions are the foundation for the local search component. The core insight is that for a solution to be optimal, the marginal profit of all active terminals must be equal to the shadow price of the total volume constraint, λ.

MRi​(Vi​)−MCi​(Vi​)=λ∀i∈{1,…,n}

This forms a system of equations that defines the optimal allocation. The memetic algorithm's local search is a heuristic solver for this system, iteratively moving volume from terminals with low marginal profit to those with high marginal profit, thereby driving them toward a common λ.

**2. The Memetic Algorithm: A Grandmaster's Approach**

Our memetic algorithm is a highly refined tool that meticulously combines global and local optimization.

**A. Chromosome Representation and Population Dynamics**

* **Chromosome**: A solution is represented as a vector of volumes, V=(V1​,V2​,…,Vn​). The sum of the components must equal Vtotal​.
* **Dynamic Population Size**: Unlike static algorithms, our solution adapts. If the algorithm detects a stagnation in fitness improvement, it dynamically increases the population size to broaden its search and escape local optima. Once progress resumes, the population size reverts to its initial value.

**B. Fitness Function: A Multi-layered Objective**

The fitness function is a carefully constructed measure of a solution's quality that ensures economic viability.

Fitness(V)=(i=1∑n​Πi​(Vi​))−i=1∑n​[penalty1​+penalty2​]

* **Individual Rationality Penalty**: The first penalty, penalty1​, is proportional to the profit deficit.

penalty1​=M⋅max(0,Πinon−coop​−Πi​(Vi​))

This is a sharp, linear penalty that makes any solution violating individual rationality highly undesirable.

* **Viability Penalty**: The second penalty, penalty2​, is for negative profits.

penalty2​=10M⋅max(0,−Πi​(Vi​))

This penalty is an order of magnitude larger to force the algorithm to seek only solutions where all terminals are financially solvent.

**C. The Local Search: A Precision Tool**

The local search is the most critical part of the algorithm's performance. It works as a precision tool to refine a raw solution.

1. **Marginal Profit Calculation**: For a given solution, the marginal profit (MPi​=MRi​(Vi​)−MCi​(Vi​)) is calculated for each terminal.
2. **Equimarginal Drive**: The algorithm identifies the average marginal profit (MPˉ) and iteratively adjusts volumes based on the difference, (MPi​−MPˉ).
   * Terminals with a marginal profit greater than the average (MPi​>MPˉ) are deemed "under-utilized" and receive more volume.
   * Terminals with a marginal profit less than the average (MPi​<MPˉ) are "over-utilized" and have their volume reduced.
3. **Proportional Adjustment**: The volume adjustment is proportional to the marginal profit difference, ensuring that the greatest changes are made where the greatest inefficiencies exist.

ΔVi​∝−(MPi​−MPˉ)

This is a targeted hill-climbing process that rapidly converges to a local optimum.

**3. The Grand Coalition and Sub-Coalition Search**

The solution moves beyond a simple centralized model to address the fundamental game theory question of stability.

* **Stability Condition**: A coalition is stable if no member can improve its profit by leaving. Our code searches for a partition of the terminals that is "stable," meaning no sub-coalition could make more profit on its own than by participating in the larger group.
* **Recursive Search**: The CooperativeProfitMaximizer employs a recursive search. It evaluates the profitability of the grand coalition. If it is not the most profitable arrangement (e.g., a sub-coalition could earn more together), the search continues to explore partitions until a stable set of coalitions is found. The use of memoization ensures that the profit of any given sub-coalition is only calculated once, dramatically improving efficiency.