A logo of a university

AI-generated content may be incorrect.  
  
Alexandria University  
CSED: Operations Research  
Faculty of Engineering  
Assignment # 1  
Computer and Systems Department  
LP Problems Solver

**REPORT**

**Team:**  
*Name/ Ziad Islam Hosny  
ID/ 22010778*

*Name/ Ali Eldeen Maher  
ID/ 22010934*

*Name/ Mohamed Mostafa  
ID/ 22011170*

**Code Explanation:**

**Standard Simplex Method:**

**Constructor (\_\_init\_\_)**

The class is initialized with:

* A: The coefficient matrix for constraints.
* b: The right-hand side (RHS) values of the constraints.
* Z: Coefficients of the objective function.
* urv: Array indicating unrestricted variables.
* objective: Indicates whether to maximize (1) or minimize (-1).
* m: Number of constraints.
* n: Number of decision variables.
* BV: List of basic variables (slack and unrestricted variables).
* steps: Stores intermediate steps of the solution process.

**Methods:**

**1. addingSlackVars()**

* This method introduces slack variables to convert inequality constraints into equalities.
* It appends an identity matrix to the coefficient matrix A to represent slack variables.
* The objective function coefficients (Z) are updated by appending zeros for slack variables.

**2. addURV()**

* This method handles unrestricted variables (urv).
* It duplicates each unrestricted variable by introducing an equivalent negative counterpart.
* The coefficient matrix A is modified to include these additional columns.
* The objective function (Z) is also adjusted accordingly.
* Updates the list of basic variables (BV) to account for newly introduced unrestricted variables.

**3. method()**

* Implements the Simplex Method iteratively to find the optimal solution.

1. **Determine Pivot Column:**
   * If maximizing, the pivot column is chosen as the one with the most negative value in Z.
   * If minimizing, it is the one with the most positive value in Z.
2. **Check for Unbounded Solution:**
   * If all elements in the chosen pivot column are negative or zero, the solution is unbounded.
3. **Determine Pivot Row:**
   * Calculates the minimum positive ratio of RHS (b[i]) to pivot column values (A[i, pivotCol]).
   * The row corresponding to this minimum ratio becomes the pivot row.
4. **Perform Row Operations:**
   * The pivot row is scaled so that the pivot element becomes 1.
   * Other rows are adjusted to make all other elements in the pivot column 0.
   * The same transformation is applied to the objective function Z.
5. **Update Basic Variables:**
   * The entering variable replaces the departing variable in the basic variable list (BV).
6. **Check Optimality:**
   * If all coefficients in Z are non-negative (for maximization) or non-positive (for minimization), the solution is optimal.
   * If not, the process repeats until optimality is reached.
7. **Extract Solution:**
   * Once optimality is reached, the values of decision variables are extracted from the tableau.
   * The final optimized value of the objective function (Z\_final) is returned along with the solution status.

**4. ansSetup()**

* Prepares a list of variable names (x0, S0) for reference.
* This helps in formatting and displaying the solution properly.

**5. addToSteps()**

* Records each step of the Simplex process to maintain a history of computations.
* The current tableau is formatted using the tabulate library.
* This allows for step-by-step tracking of the method's progress.

**Big-M Method:**

**Constructor (\_\_init\_\_)**

**The class extends Simplex and initializes with:**

* A, B, Z, urv, signs, objective.
* M: A large constant (penalty value for artificial variables).

**Key Variables:**

* self.artificial\_vars: Tracks artificial variable indices.
* self.BV: Stores basic variables.
* self.steps: Records iterative computations.

**Methods in BigMMethod**

**1. addingArtificialVars()**

* Introduces artificial variables for >= and = constraints.
* Appends columns to A for artificial variables.
* Modifies Z by adding a large penalty (M) for artificial variables.
* Updates the basic variable (BV) list.

**2. method()**

* Ensures all b values are positive.
* Adjusts the objective function Z by incorporating artificial variables with penalty M.
* Calls the Simplex method to find the optimal solution.
* Checks if artificial variables remain in the basis, indicating infeasibility.
* If feasible, runs Simplex again without artificial variables.

**3. ansSetup()**

* Prepares a list of variable names (x0, S0) for reference.
* This helps in formatting and displaying the solution properly.

**4. addToSteps()**

* Records each step of the Simplex process to maintain a history of computations.
* The current tableau is formatted using the tabulate library.
* This allows for step-by-step tracking of the method's progress.

**5. initialTableau()**

* Constructs the tableau manually by appending slack and artificial variables.
* Modifies the objective function (Z) by subtracting M times artificial variables.
* Sets up basic variables (BV).

**6. solve()**

* Adjusts Z using M penalties for artificial variables.
* Calls the Simplex method to solve the problem.
* Checks feasibility and returns the final solution.

**Two-Phase Method:**

**Constructor (\_\_init\_\_)**

The class is initialized with:

* A, b, Z, signs, objective, m, n, tableau, BV, Steps.
* artificialVars: Tracks the number of artificial variables introduced.

**Methods in TwoPhaseMethod Class**

**1. initialTableau()**

* Constructs the initial tableau by adding **slack** and **artificial variables**.
* If a constraint is ’ **>=**’, a **negative slack variable** and an **artificial variable** are introduced.
* If a constraint is ‘**=**’, only an **artificial variable** is introduced.
* The method modifies self.tableau to accommodate these variables.

**2. phaseOne()**

* This phase ensures feasibility by removing artificial variables.
* Constructs a **Phase 1 objective function** (z\_p1), which seeks to minimize the sum of artificial variables.
* Modifies Z to subtract artificial variable rows, ensuring they are driven to zero.
* Identifies basic variables corresponding to artificial variables.
* Uses the **Simplex Method** to optimize Phase 1 and eliminate artificial variables.
* Stores the modified tableau and the updated b values for Phase 2.

**3. phaseTwo()**

* Restores the original objective function Z after artificial variables are removed.
* Eliminates artificial variable columns from the tableau.
* Adjusts Z for unrestricted variables.
* Calls the **Simplex Method** again to find the optimal solution for the original problem.
* Outputs the final values of variables and the optimal value of Z.

**Preemptive Method for goal programming:**

**Constructor (\_\_init\_\_)**

The class is initialized with:

* A, b, urv, m, n, BV, steps.
* G: Goal coefficient matrix.
* Gb: Right-hand side (RHS) values for the goal constraints.
* signsk: Number of goal constraints.

**1. initialTableau()**

* Constructs the initial simplex tableau to incorporate both goal and general constraints.
* Adds slack and surplus variables to transform inequalities into equalities.
* Ensures the tableau structure is correctly set up for further calculations.

**2. setGoals()**

* Adjusts the goal constraints by introducing positive and negative deviation variables (S+ and S-).
* These variables track deviations from target goal values, ensuring a structured approach to prioritization.
* Modifies the goal matrix (G) accordingly.

**3. addURV()**

* Handles unrestricted variables by introducing counterpart variables with opposite signs.
* Updates the basic variables list to accommodate unrestricted variables.

**4. ansSetup()**

* Prepares variable names for tracking and visualization.
* Includes decision variables (x), deviation variables (S+, S-), and slack variables (S).

**5. addToSteps()**

* Records each computational step in a formatted table.
* Captures the evolving tableau and goal adjustments at different iterations.

**6. method() (Core Goal Programming Execution)**

* Implements the **Preemptive Goal Programming Algorithm** iteratively for each priority level.

**Steps in the method:**

1. **Adjust Goal Rows:**
   * Incorporates goal constraints into the simplex tableau.
   * Updates the right-hand side values (Gb).
2. **Column Selection for Pivoting:**
   * Identifies columns with positive coefficients in the goal function (G).
   * Sorts columns in descending order of their contribution to goal achievement.
3. **Row Selection for Pivoting:**
   * Calculates pivot row based on minimum positive ratio of RHS to pivot column.
   * Ensures feasible movement along constraints.
4. **Row Operations:**
   * Performs row transformations to maintain simplex feasibility.
   * Eliminates deviation in higher-priority goals first.
5. **Iterative Adjustment:**
   * Repeats the process until the highest-priority goals are satisfied.
   * Moves to lower-priority goals only if higher ones are optimized.
6. **Extract Solution:**
   * Returns the final optimized values for decision variables.
   * Outputs deviation values for goal constraints.

**Sample runs:**

**A screenshot of a computer

AI-generated content may be incorrect.Standard Simplex Method:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Big-M Method:**

**A screenshot of a computer

AI-generated content may be incorrect.Two-Phase Method:**

**A screenshot of a computer program

AI-generated content may be incorrect.** A screenshot of a computer

AI-generated content may be incorrect.

**A screenshot of a computer

AI-generated content may be incorrect.Preemptive Method for goal programming:**

**A screenshot of a computer screen

AI-generated content may be incorrect.** 