Operations Research Interview Questions & Preparation

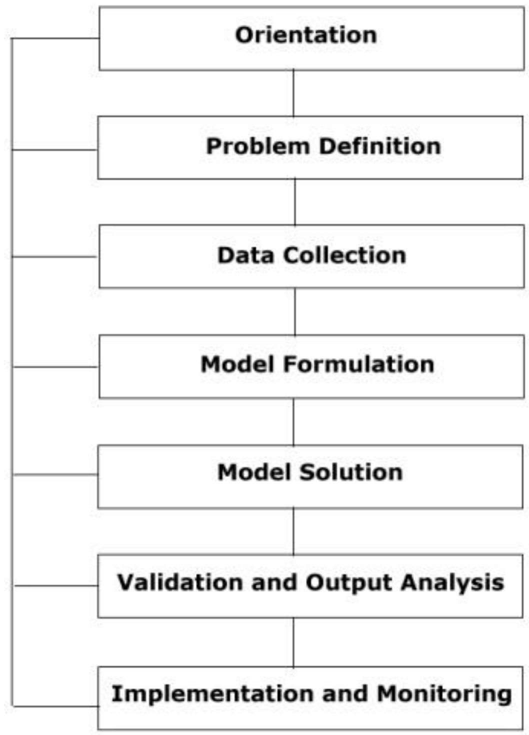
(ongoing)

Compiled by Tanmoy Das

Table of Contents

1 Basic of Operations Research

1. What are the steps of problem solving in OR?



https://sites.pitt.edu/~jrclass/or/or-intro.html

https://web.mit.edu/urban_or_book/www/book/chapter1/1.3.html

2. What do you do for infeasibility analysis?
3. Problem degeneration
4. Convex and concave functions
5. Mathematical programming formula (from verbal to math)
    a. Practise a few+++

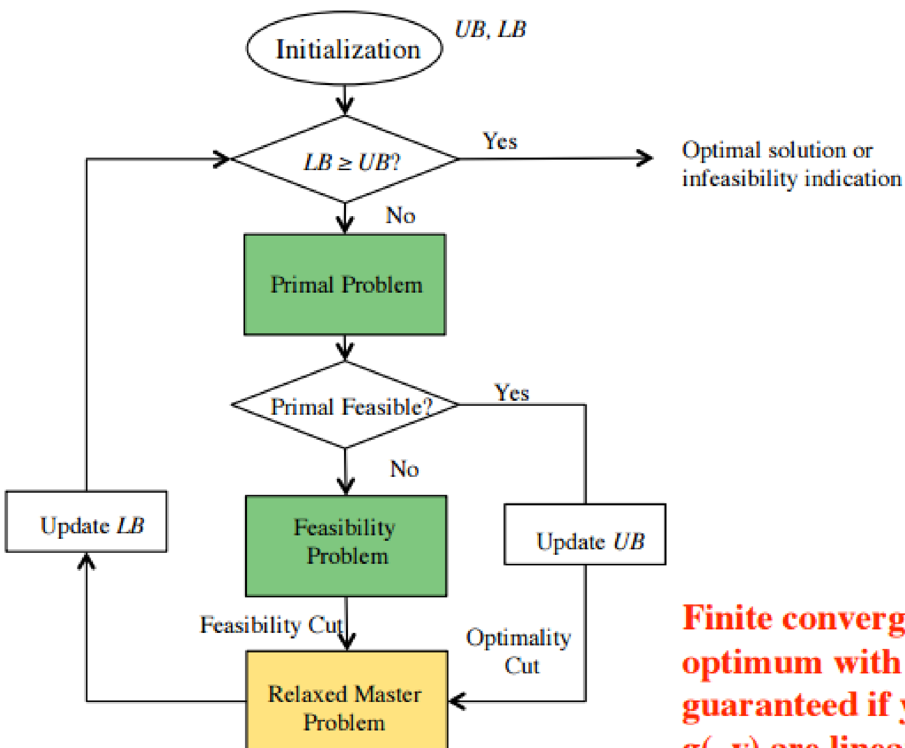
2 Algorithm

2.1 Integer Programming

2.1.1 B&B

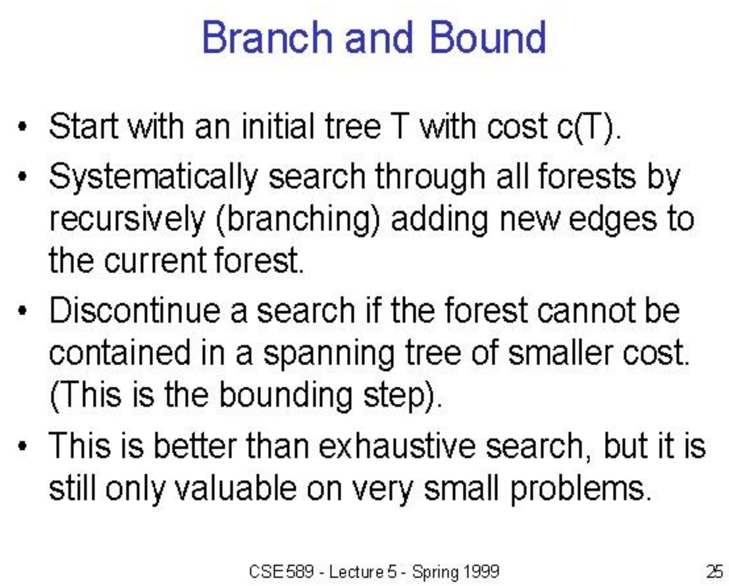
Enumerative (branch and bound, implicit enumeration) methods solve a discrete optimization problem by breaking up its feasible set into successively smaller subsets, calculating bounds on the objective function value over each subset, and using them to discard certain subsets from further consideration. The bounds are obtained by replacing the problem over a given subset with an easier (relaxed) problem, such that the solution value of the latter bounds that of the former. The procedure ends when each subset has either produced a feasible solution, or was shown to contain no better solution than the one already in hand. The best solution found during the procedure is a global optimum.

1. What are the steps in Benders Decomposition (UPS)

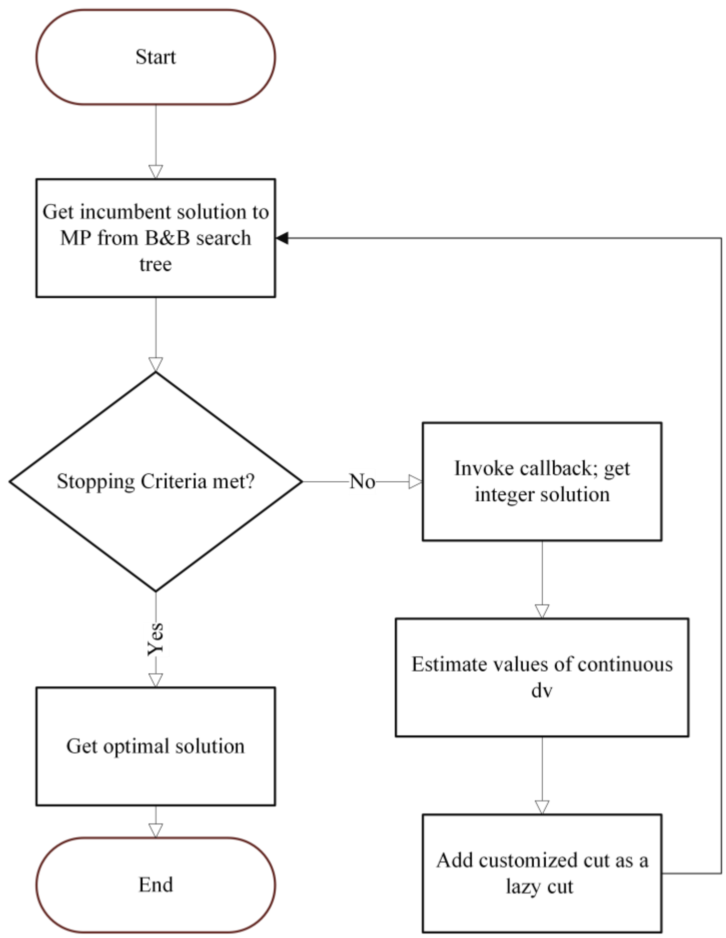


Finite convergence proof see to an optimum with a given tolerance is guaranteed if y is integer or f(.,y), g(.,y) are linear for all y in Y.

https://optimization.cbe.cornell.edu/index.php?title=Generalized_Benders_decomposition_%28GBD%29



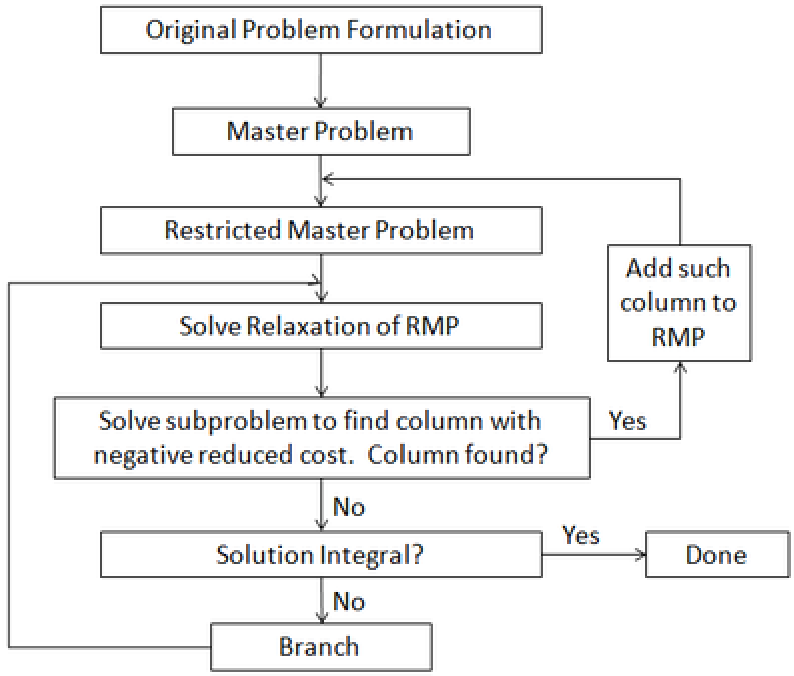
2.1.2 Branch & Cut



What is a cut?

What is incumbent solution?

2.1.3 B&P



https://www.wikiwand.com/en/Branch_and_price

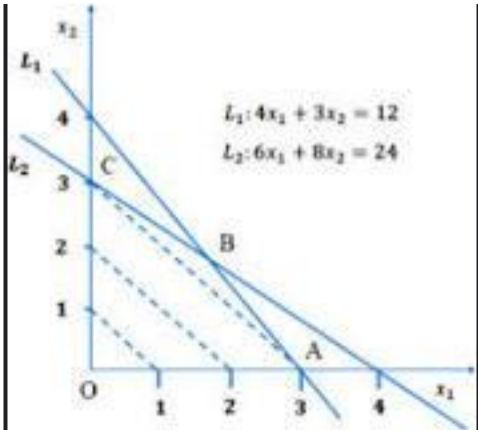
Restricted MP means??

Reduced cost: how much objective function value changes with one unit change of RHS

Shadow price: unit change of constraint → obj

2.1.4 Cutting plane

What is the cutting plane method?



The idea of this method is based on developing a sequence of linear inequalities, which is known as cuts. Then these cuts are used to reduce a part of the feasible region and give a feasible region of the ILP problem. The hyperplane boundary of a cut is called the cutting plane.

2.1.5 Column Generation

1. How to use column generation for VRP problem?
   a. Generic: Column generation or delayed column generation is an efficient algorithm for solving large linear programs.
      The overarching idea is that many linear programs are too large to consider all the variables explicitly. The idea is thus to start by solving the considered program with only a subset of its variables. Then iteratively, variables that have the potential to improve the objective function are added to the program.
      https://medium.com/@sean-patrick-kelley/how-to-implement-column-generation-for-vehicle-routing-bdb8027c957f

2.1.6 Other decomposition methods

2.1.6.1 Comparing algorithms

	Advantage	**Disadvantage**
B&B	Whenever the problem is small and the branching can be completed in a reasonable amount of time, the algorithm finds an optimal solution. By using the branch and bound algorithm, the optimal solution is reached in a minimal amount of time If the problem is simple enough BnB can look at the whole problem at once and provide an optimal solution, if the problem is enormous (as practical problems, opposed to simple ones, usually are) then solving pieces involves later recombining them and reiterating over them; that can lead to	

	small errors, or pieces that were useful in the final solution having been trimmed.	
B&C	Branch-and-cut methods combine branch-and-bound and cutting-plane methods. The cutting-planes are generated throughout the branch-and-bound tree. The underlying idea is to work on getting as **tight as possible bounds in each node** of the tree and thus reducing the number of nodes in the search tree.	Of course, there is an obvious trade-off. If many cuts are added at a node, re-optimization may slow down. In addition, keeping all the information in the tree is more difficult. On the other hand, the size of the search tree may be reduced significantly.
Lagrange relaxation		Difficulty in Solving: Lagrange's method can become computationally complex, especially for problems with **multiple constraints or variables**. Finding the appropriate Lagrange multipliers and s olving the resulting system of equations can be challenging
Column generation	Often a problem can be split into a master problem and the subproblem, that is called column generation. Branch-and-price is a hybrid of branch and bound and column generation methods	

decomposition methods? Benders, Column Generation, Branch and Bound, Branch and Cut, Branch and price, Dantzig wolf decomposition

Dantzig wolf decomposition

While there are several variations regarding implementation, the Dantzig–Wolfe decomposition algorithm can be briefly described as follows:

1. Starting with a feasible solution to the reduced master program, formulate new objective functions for each subproblem such that the subproblems will offer solutions that improve the current objective of the master program.
2. Subproblems are re-solved given their new objective functions. An optimal value for each subproblem is offered to the master program.
3. The master program incorporates one or all of the new columns generated by the solutions to the subproblems based on those columns' respective ability to improve the original problem's objective.

4. Master program performs *x* iterations of the simplex algorithm, where *x* is the number of columns incorporated.
5. If objective is improved, goto step 1. Else, continue.
6. The master program cannot be further improved by any new columns from the subproblems, thus return.

https://en.wikipedia.org/wiki/Dantzig%E2%80%93Wolfe_decomposition

2.2 Linear Programming

1. What are the different solution techniques for LPs?
   a. Graphical method, Simplex, interior point algorithms, and the Ellipsoid Method
   b.

2.2.1 Simplex

1. Optimality conditions in Simplex
   a. The entering variable in a maximization (minimization) problem is the non-basic variable having the most negative (positive) coefficient in the Z-row. The optimum is reached at the iteration where all the Z-row coefficient of the non-basic variables are non-negative (non-positive).
   b. The optimality condition is the reduced-cost condition. When reduced costs are positive, you have the optimum

Steps in Simplex

2. 3 rules in converting primal to dual?
3. When to use primal vs dual?

Steps in SIMPLEX algorithm

I. Standard form
II. Introducing slack variables
III. Creating the tableau
IV. Pivot variables
V. Creating a new tableau
VI. Checking for optimality
VII. Identify optimal values

Step 1: Establish a given problem. (i.e.,) write the inequality constraints and objective function.

Step 2: Convert the given inequalities to equations by adding the slack variable to each inequality expression.

Step 3: Create the initial simplex tableau. Write the objective function at the bottom row. Here, each inequality constraint appears in its own row. Now, we can represent the problem in the form of an augmented matrix, which is called the initial simplex tableau.

Step 4: Identify the greatest negative entry in the bottom row, which helps to identify the pivot column. The greatest negative entry in the bottom row defines the largest coefficient in the objective function, which will help us to increase the value of the objective function as fastest as possible.

Step 5: Compute the quotients. To calculate the quotient, we need to divide the entries in the far right column by the entries in the first column, excluding the bottom row. The smallest quotient identifies the row. The row identified in this step and the element identified in the step will be taken as the pivot element.

Step 6: Carry out pivoting to make all other entries in column is zero.

Step 7: If there are no negative entries in the bottom row, end the process. Otherwise, start from step 4.

Step 8: Finally, determine the solution associated with the final simplex tableau.

https://byjus.com/maths/linear-programming/#:~:text=The%20linear%20programming%20problem%20can,and%20graphical%20method%20in%20detail.

1. **Set up the problem.** That is, write the objective function and the inequality constraints.
2. **Convert the inequalities into equations.** This is done by adding one slack variable for each inequality.
3. **Construct the initial simplex tableau.** Write the objective function as the bottom row.
4. **The most negative entry in the bottom row identifies the pivot column.**
5. **Calculate the quotients. The smallest quotient identifies a row. The element in the intersection of the column identified in step 4 and the row identified in this step is identified as the pivot element.** The quotients are computed by dividing the far right column by the identified column in step 4. A quotient that is a zero, or a negative number, or that has a zero in the denominator, is ignored.
6. **Perform pivoting to make all other entries in this column zero.** This is done the same way as we did with the Gauss-Jordan method.
7. **When there are no more negative entries in the bottom row, we are finished; otherwise, we start again from step 4.**
8. **Read off your answers.** Get the variables using the columns with 1 and 0s. All other variables are zero. The maximum value you are looking for appears in the bottom right hand corner.

https://math.libretexts.org/Bookshelves/Applied_Mathematics/Applied_Finite_Mathematics_(Sekhon_and_Bloom)/04%3A_Linear_Programming_The_Simplex_Method/4.02%3A_Maximization_By_The_Simplex_Method

Sensitivity analysis and reduced cost

The opportunity/reduced cost of a given decision variable can be interpreted as the rate at which the value of the objective function (i.e., profit) will deteriorate for each unit change in the optimized value of the decision variable with all other data held fixed.

2.2.2 Dual Simplex

Why we use dual simplex method instead of simplex method?
In the dual simplex, new primal cuts correspond to new dual variables, which are initialized as nonbasic, and thus the previous solution is still dual feasible. Thus, dual simplex does not need to regain feasibility. So in a mixed-integer-programming context, dual simplex will usually outperform primal simplex.
In general it is easier to get dual feasibility than primal feasibility, and dual simplex appears to make more progress in many iterations.

Many commercial solvers also offer Barrier methods to solve LP(-Relaxations). The big drawback of interior point methods is that they can't really be warmstarted, and when solving mixed-integer-programmes using branch & bound are generally given a reasonable warm/hot-start (the solution of the previous LP Relaxation). source

2.3 NLP

1. Contraint optimisation (conditions, Lagrangien relaxation, Quasi-newton, Rampson)

Lagrangian relaxation is a technique well suited for problems where the constraints can be divided into two sets:

“good” constraints, with which the problem is solvable very easily •

“bad” constraints that make it very hard to solve.

The main idea is to relax the problem by removing the “**bad**” constraints and putting them into the objective function, assigned with weights (the Lagrangian multiplier). Each weight represents a penalty which is added to a solution that does not satisfy the particular constraint.

Quasi-Newton methods are methods used to either find zeroes or local maxima and minima of functions, as an alternative to Newton's method. They can be used if the Jacobian or Hessian is unavailable or is too expensive to compute at every iteration. The "full" Newton's method requires the Jacobian in order to search for zeros, or the Hessian for finding extrema. Some iterative methods that reduce to Newton's method, such as SLSQP, may be considered quasi-Newtonian.

https://en.wikipedia.org/wiki/Quasi-Newton_method

2.4 MIP

1. What's the difference between LP & MIP?
   a. Continuous DV vs mix of integer and continuous variables
   b. MIP is difficult to solve compared to LP
2. What are the different solution techniques for MIPs?
   a.
3. ++

Logical transformation

Piecewise linear transformation

2.4.1 Exact Method

2.4.2 Heuristics

For example, a greedy strategy for the travelling salesman problem (which is of high computational complexity) is the following heuristic: "At each step of the journey, visit the nearest unvisited city." This heuristic does not intend to find the best solution, but it terminates in a reasonable number of steps; finding

2.4.3 Meta-heuristics

2.4.3.1 *GA*

2.4.3.2 *PSO*

3 Problem Type

3.1 Knap-sack problem

3.2 Set Covering

$$\begin{aligned} \max \quad & \sum_j a_j y_j \\ \text{s.t} \quad & \sum_{i \in C_j} x_i \geq y_j \quad \forall j \\ & \sum_i x_i = k \\ & x_i, y_j \in \{0, 1\} \end{aligned}$$

https://optimization.cbe.cornell.edu/index.php?title=Set_covering_problem

https://en.wikipedia.org/wiki/Set_cover_problem

3.3 P-median

The "p-median" problem is a distance based optimization problem in which **p facilities** need to be located and assigned to the demand points such that each demand point is mapped to a single facility, and the sum of the weighted distance between all demand points and corresponding facilities is minimized.

3.4 MCLP

Problem Name	Aim	Objective	
Set covering problem	locate the minimum number of facilities required to "cover" all of the demand nodes.	minimizes the number of facilities located	Minimize $\sum_{j \in J} x_j$ subject to: $\sum_{j \in N_i} x_j \geq 1 \; \forall i \in I$ $x_j \in \{0,1\} \; \forall j \in J$
MCLP	Budget constraint, so build p number of facilities	Maximize demand	Maximize $\sum_{i \in I} h_i z_i$ subject to: $\sum_{j \in N_i} x_j - z_i \geq 0 \; \forall i \in I$ $\sum_{j \in J} x_j = p$ $x_j \in \{0,1\} \quad \forall j \in J$ $z_i \in \{0,1\} \quad \forall i \in I$

p-center	minimizing the maximum distance that demand is from its closet facility given that we are siting a pre-determined number of facilities	Minimize max distance	Maximize W subject to: $\sum_{j \in J} x_j = p$ $\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$ $y_{ij} - x_j \leq 0 \quad \forall i \in I, j \in J$ $W - \sum_{j \in J} h_i d_{ij} y_{ij} \geq 0 \ \forall i \in I$ $x_j \in \{0,1\} \quad \forall j \in J$ $y_{ij} \in \{0,1\} \quad \forall i \in I, j \in J$
p-median	Locations of p facilities to minimize the demand-weighted total distance between demand nodes and the facilities	minimize the demand-weighted total distance	Minimize $\sum_{i \in I} \sum_{j \in J} h_i d_{ij} y_{ij}$ subject to: $\sum_{j \in J} x_j = p$ $\sum_{j \in J} y_{ij} = 1 \quad \forall i \in I$ $y_{ij} - x_j \leq 0 \quad \forall i \in I, j \in J$ $x_j \in \{0,1\} \quad \forall j \in J$ $y_{ij} \in \{0,1\} \quad \forall i \in I, j \in J$

Maximal Covering Location Problem

$$\max \sum_{i \in I} h_i y_i$$

$$s.t.$$

$$\sum_{j \in N_i} x_j \geq y_i, \qquad i \in I$$

$$\sum_{j \in J} x_j = p,$$

$$x_j \in \{0,1\}, \qquad j \in J$$

$$y_i \in \{0,1\}, \qquad i \in I$$

https://en.wikipedia.org/wiki/Maximum_coverage_problem

3.5 Location Allocation

1. Can you write a simple facility allocation model? When we use location-allocation modeling?

Verbal:

Objective: Min cost, $FixedCost_f \, .open_f$

St. number of facility <= 10

Allocation_f,c <= availability_f

3.6 Network Optimization

3.6.1 VRP formulation

3.6.2 TSP formulation

$$
\begin{aligned}
\min \sum_{i=1}^{n} \sum_{j\neq i, j=1}^{n} c_{ij} x_{ij}&: \\
x_{ij} &\in \{0,1\} && i,j = 1,\ldots,n; \\
u_i &\in \mathbf{Z} && i = 2,\ldots,n; \\
\sum_{i=1, i\neq j}^{n} x_{ij} &= 1 && j = 1,\ldots,n; \\
\sum_{j=1, j\neq i}^{n} x_{ij} &= 1 && i = 1,\ldots,n; \\
u_i - u_j + (n-1)x_{ij} &\leq n-2 && 2 \leq i \neq j \leq n; \\
2 \leq u_i &\leq n && 2 \leq i \leq n.
\end{aligned}
$$

3.6.3 Minimum Spanning Tree

3.6.3.1 Kruskal

Learning source: link.

4 Computational Complexity

1. Please describe one **linear optimization problem/algorithm** and state its **computational complexity**.

Linear programming:

Linear programming is **an optimization technique for a system of linear constraints and a linear objective function**. An objective function defines the quantity to be optimized, and the goal of linear programming is to find the values of the variables that maximize or minimize the objective function.

Complexity

The complexity of an optimization algorithm depends on the following factors:

1- Number of iterations, 2- Number of individuals in the population, 3-complexity of the objective function, 4- If you sort the individuals, the complexity of sorting should be added

Example O(N*M*P*Q) where N is #1, M is #2, P is #3, Q is #4

The better estimation for #1 and #2 is computing the NFE(number of function evaluations) source: https://www.researchgate.net/post/How-can-I-calculate-the-computational-complexity-of-any-optimization-algorithm

The computational complexity, in general, depends on the optimization algorithm and the technique that you use. In some algorithms, the complexity can be measured by the time that the CPU needs to run the algorithm, others consider the computational complexity as the number of nested loops (for loops and others) per run and can be written as O(x), where x is your nested loops.

5 Uncertainty

5.1 Robust optimization

Why we always use “worst-case”

https://math.stackexchange.com/questions/646380/reasons-for-the-worst-case-scenario-in-robust-optimization

less worst case, then not much robust, right?

If we add worst situation, and train model accordingly, it will be robust for any other less-worse conditions, as well.

Other perspective: lightly robust

- An optimal (feasible) solution is robust if it stays optimal (feasible) under any realization of the data

5.2 Stochastics Optimization

6 Large Scale Optimization

1. Tell me about large-scale optimization
2. How do you solve them?
3. What heuristics you will apply for large-scale?

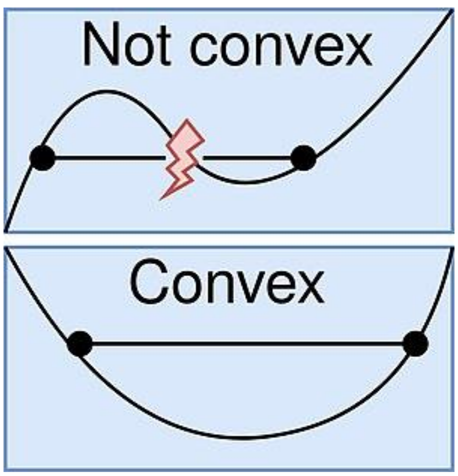
7 Gurobi

1. How you set parameters of B&B
2. How to handle infeasibility?
    a. Model.computeIIS (Irreducible Inconsistent Subsystem)
        i. This set is still infeasible
        ii. feasible if single constraint or bound removed
    b. How to manually check infeasibility?
3. Warm start
    a. How to use pretrained optimization model as initial sol

IIS

8 Miscellaneous

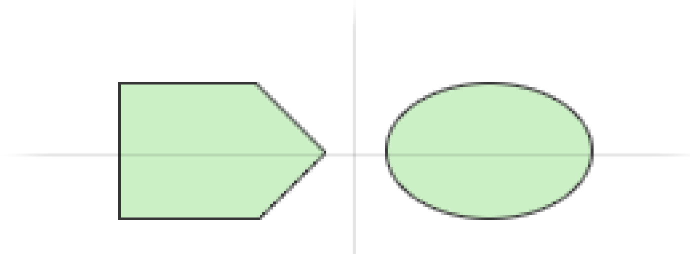
8.1 Convex function



8.1.1.1 ***Convex Optimization Problems***

A *convex optimization problem* is a problem where all of the constraints are convex functions, and the objective is a convex function if minimizing, or a concave function if maximizing.

Linear functions are convex, so linear programming problems are convex problems. Conic optimization problems -- the natural extension of linear programming problems -- are also convex problems. In a convex optimization problem, the feasible region -- the intersection of convex constraint functions -- is a convex region, as pictured below.


Convex

9 MCQs

1. If the feasible set of an optimization problem is unbounded, which of the following is true? (C3AI-T)
    a. No finite optimum point exists
    b. It has an infinite number of feasible points
    c. The existence of finite optimum points can not be assured
    d. None of the above.

Nash, "an **equilibrium** point is an n-tuple such that each player's mixed strategy maximizes his payoff if the strategies of the others are held

How many routes possible in the traveling salesman problem with n cities?

$(n-1)!$