Lecture 1: CRP: Container relocation problem: Vertical transportation system

Heuristic: find a solution quickly, but no guarantee of optimal

Bin packing problem: put objects into bins: Linear integer problem_

Para: list of items n, weight of items Wn, capacity of bins C, list of bins available u

Variables: yi=0 or 1 xij=0 or 1

VRP: vehicle routing problem: deliver parcels, with limited vehicles and capacity

Model1(2-index directed): min∑Cij*xij, s.t.∑xij(in arc)=1, ∑xij(out arc)=1, ∑x0j(o out)≤or=K, ∑xij≥r(S), xij∈{0,1}, where r(S)is solution of bin packing problem over a set S

Ad: polynomial variable number, strong lower bound from linear relaxation, direction of car

Dis: exponential constraint number, route orientation can be hard 2^K solutions to prove

Model2(2-index undirected): min∑Ce*Xe, s.t.∑xe=2, ∑xe(o out)=2K, ∑Xe≥2r(S), xo∈{012}

Ad: polynomial variable, linear relaxation, Dis: Exponential constraints, no flexible model to include realistic constraint, Requires solution for NP hard bin packing problem

RCI(Rounded Capacity Inequalities):___

MTZ(Miller-Tucker-Zemlin constraint): ui -uj +Qxij≤Q-qi, capacity constraint: qi≤ui≤Q

Model3(3 index):___

Ad: polynomial variables and constraint, flexibility to model truck specific constraints

Dis: a weak lower bound from Linear Relaxation, permutation of routes |K|!

Model4(set partitioning): Min∑Crλr, s.t∑airλr=1,∑λr=K, λr ∈{0,1}, where r∈ feasible set

Ad: polynomial constraint, strong lower bound, can model complex constraint, no symmetry problems

Dis: exponential variables, Route≈!(N-1) with 12 customer, ≈11!

Intra route constraint(VRP with time window): Tik −Tjk +Mxijk ≤M−tij earliest and latest arrive time, car waiting time, time for customer service, tij(Ti) time of day, soft time window

Fleet character: multiple depot, different capacity, route of trailers/trucks, split delivery,

Lecture 2: Branch and bound: LP relaxation z ->prune by infeasibility, ->if z*<z, prune by bound ->if x integer and z*<z, set z*=z ->select fractional variable and branch

Feasible solution (randomly solve 1) > optimal solution > linear relaxation (allow fraction)

Pruning: 1. no feasible solution, 2.by bound (LB is more than UB), 3.by optimality

每一次 branch 就是分数向上向下取整然后上下分割，然后排除法，直到找到最优

Best node first: Select the node that has the best bound first. Ad: lead to smallest BB tree, Dis: long time

Depth first search: Descend the enumeration tree quickly to find a feasible solution. Select the most branched node,

Ad: find feasible int quickly, Dis: bigger tree.

Branching strategies: Most fractional variable(close to 0.5), Strong branching 向上下取整 stronger bound

Branch-and-Cut: LP 松弛(LR)->生成切割平面->LP 松弛->if 解仍然不整数，分支创建子问题->repeat, Chvatal-Gomory cuts: ∑⌊vaj⌋*xj ≤ ⌊vb⌋两边都向下取整，域就不会排除整数

Branch-and-price: LRMP(Linear Relaxation Master Problem):Min∑cλ,s.t ∑aλ =1,1 ≥ λr ≥ 0 for MP it’s λr=1 or 0

When MP = RMP ? Optimal if the reduced costs of Non-Basic varibles is non negative, cr = c – cT*B^−1*Ar

Dummy variable: min MλDs.t. λD = 1,1 ≥ λD ≥ 0, If cannot eliminate dummy variable, the problem is infeasible. prune the problem and continue B&P, if can, will create negative reduced cost and add to RMP

Lecture 3: SPP(shorted path problems): principle of optimality: polynomial time O(|Arcs|)

D(i) is the shortest path distance from the origin to i. N(p) is the set of nodes connected to the node (p). cip is the cost of going from node i to p directly: D(p) = min {D(i) + cip}

M(k,i) be a bucket: M(k,i) = {L = (i,c,L)}, where c = Dk(i) (cost) and L = Lk−1

每到一个节点，他周围的节点就要全部更新，然后也要往回更新直到所有点都遍历

Negative cost cycle will make shortest path problem Unbounded!

SPPRC(SPP with resource constraint): Time constraints: earliest and latest arrival, the time consumed,

Capacity constraints: ∑qi≤Q, qi is demand of each node

Lextend = (ie,ce,Te,Qe,Le) a label that we want to extend from ie to jn, check resource constra

REF(Resource Extension Functions):arc (i,j), capacity fQ = Qi + qj, time fT = Ti + tij
Feasibility checks: Before extend a label, check resource constraints: capacity and time
If arrive early: Te +t(ie,jn) <ejn; we wait, REF modified to fT =max{ejn, Te +t(ie,jn)}
Dominance: L1 = (i1, c1, T1, Q1, L1) dominates L2 if i1=i2, c1≤c2, T1≤T2, Q1≤Q2
ESPPRC(Elemetary SPP with Resource Constraints): prevent negative cost cycles
Let Ve: set of unreachable nodes, if node in set Ve, unfeasible. Lextend = (ie,ce,Te,Qe,Ve,Le)
Feasibility checks: jn ∈ Ve, or Te +t(ie,jn) >bjn, or Qe + qjn > Q, then extension not feasible
Dominance: L1 = (i1, c1, T1, Q1, V1, L1) dominates L2 i1=i2,c1≤c2,T1≤T2,Q1≤Q2,V1⊆V2
relax the elemetarity constraints, get SPPRC as a relaxation(lower bound) then remove 1by1
Master Problem(The set-covering formulation): Minimize ∑cr*λr, s.t.∑air*λr ≥1, λr ∈ {0, 1}
Relaxation: k-cycle: cycles containing more than k nodes are allowed(better than SPPRC)
Ng-route: Only allow cycles formed by distant customers, Ngi= 5 : Ngi contains 5 nearest
Unreachable: V is replaced by a smaller set U, Un = Ue ∩ Ng (Only common element)
Lecture4: Heuristics: reasons to use: need rapid solution, hard for B and B, MIP ineffective
Greedy: Construction heuristic: Nearest Neighbor Heuristic: select a city randomly, then its nearest unvisited city time complexity of O(n^2),it building a solution from scratch
Sorted Edges Heuristic: Sort the edges in nondecreasing order(begin from cheapest)
Local Search: k-opt: removing k edges from solution and reconnect: time complex O(n^k)_
Metaheuristics: SA(Simulated Annealing): Probability of accepting a new solution:____
cooling factor of α: Tk+1 ←α×Tk, algorithm stop when: Tk ≤ stop, therefore α^k∗T0 =stop
Deterministic: NN, Sorted edges, k-opt, Stochastic: Simulated annealing,NN random start
Lecture5: Relocate heuristic: N(w) customer 从 route 1 换到 route 2; 2-opt: exchange edges
Small Neighborhood: k≤3, for 2-opt in TSP and Relocate in VRP, time complex: O(n^2)
LNS(Large Neighborhood Search): In a VRP with 100 customers, removes 15%： 100!/15!*85!
Acceptance criteria: Hill-climber: only improving, Threshold accept, Simulated annealing
Destroy method: too little: small neighborhood, local optima; too much: constructive heuristic
Rebuild method: Exact Method: reconstruct destroyed solution, Heuristic: escape local optima
ALNS(Adaptive Large Neighborhood Search):allow multiple destroy and rebuild methods
Matheuristics: combine exact methods like MIP and metaheuristics, i.e. LNS, Variable fixing: in BB algorithm, find a variable that is fractional and set to one, Column generation
RMP(Restricted Master problem): Minimize ∑cr*λr r ∈ Ω'小 set s.t∑air*λr ≥1, λr ∈ { 0 , 1 }
Column generation: solve RMP->dual variable->solve pricing(ESPPRC)->routes with negative reduce cost ->solve RMP, exact pricing 太慢，可先 Heuristic Pricing 再 exact pricing
Pricing Heuristic: Develop multiple heuristics 从最快到最慢一个一个试，最后用 exact
Column Generation: Variable fixing: 1Solve the MP, find a fractional, e.g., 0.95, and set equal to 1, 2. pricing until no negative column, 3. Select another fractional,4. Until feasible
VNS(Variable Neighborhood Search): change of neighborhood with descent and perturbation
Nk(x) is the set of solutions in the kth neighborhood of x, x′ ∈ X is local minimum
Neighborhood change: x′ a local solution in Nk, if f(x')<f(x), then x=x', k set to initial(1)
VND(Variable neighborhood descent):deterministic, find the best neighbor in Nk(x)
BVNS(basic VNS): stochastic part: shake function: generate random x', deterministic part: best improvement function for local search, General VNS: choose VND for local search
MDVRP(multi-depot VRP): 有很多个顶点; Randomized VND:随机排序 neighbor list

流程图：