Part VI

LP Decomposition Approaches

Outline

1. delayed constraint generation
   - cutting-plane method
   - lagrangian relaxation
   - Benders decomposition
2. delayed column generation

cutting-plane method

- if an LP $(\bar{P}) : \bar{z} = max\{cx \mid A_j x \leq b_j \forall j \in J\}$ has many constraints, then generate and add them progressively starting from a restricted set $J_r \subseteq J$

cutting-plane algorithm

while $J_r \subsetneq J$
  solve $(\bar{P}_r) \to \bar{x}$
  find $J' \subseteq J$ s.t. $A_j \bar{x} > b_j, \forall j \in J'$
  augment J_r with J' or stop

- decomposition: restricted master LP $(\bar{P}_r)$ / the separation subproblem

cutting-plane method

the template paradigm

1. $t \leftarrow 0, J^0 \subseteq J$
2. $x^{t+1} \leftarrow \mathtt{opt}(\bar{P}^t) : \bar{z}^t = max\{cx \mid A_j x \leq b_j \forall j \in J^t\}$
3. find $J' \subseteq J$ s.t. $A_j x^{t+1} > b_j, \forall j \in J'$
4. if $J' = \emptyset$ STOP RETURN $\bar{z} = \bar{z}^t$
5. otherwise $J^{t+1} \leftarrow J^t \cup J'$ GOTO 2 WITH $t \leftarrow t+1$

- the procedure (usually) stops long before the constraints are all generated
- the separation subproblem must be solved quickly (exactly and/or heuristically)

example: Symmetric Traveling Salesman Problem

$$(P): z = \max \sum_{e \in E} p_e x_e, \quad \text{s.t.} \quad \sum_{e \in \delta(v)} x_e = 2 (\forall v \in V), \quad \sum_{e \in \delta(S)} x_e \geq 2 (\forall \emptyset \subsetneq S \subsetneq V), \quad x \in \{0,1\}^E.$$

one subtour constraint for each proper subset of vertices:
generate them on the fly when S is disconnected

example: Symmetric Traveling Salesman Problem

$$(P): z = \max \sum_{e \in E} p_e x_e, \quad \text{s.t.} \quad \sum_{e \in \delta(v)} x_e = 2 (\forall v \in V), \quad \sum_{e \in \delta(S)} x_e \geq 2 (\forall \emptyset \subsetneq S \subsetneq V), \quad x \in \{0,1\}^E.$$

one subtour constraint for each proper subset of vertices:
generate them on the fly when S is 2-disconnected

example: Symmetric Traveling Salesman Problem

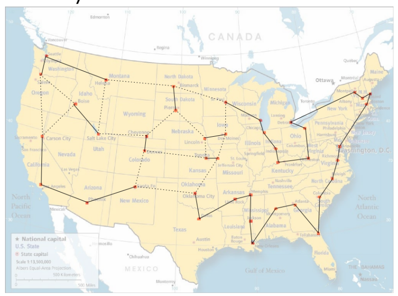
$$(P): z = \max \sum_{e \in E} p_e x_e, \quad \text{s.t.} \quad \sum_{e \in \delta(v)} x_e = 2 (\forall v \in V), \quad \sum_{e \in \delta(S)} x_e \geq 2 (\forall \emptyset \subsetneq S \subsetneq V), \quad x \in \{0,1\}^E.$$

one subtour constraint for each proper subset of vertices:
generate them on the fly until exist

example: Symmetric Traveling Salesman Problem

$$(P): z = \max \sum_{e \in E} p_e x_e, \quad \text{s.t.} \quad \sum_{e \in \delta(v)} x_e = 2 (\forall v \in V), \quad \sum_{e \in \delta(S)} x_e \geq 2 (\forall \emptyset \subsetneq S \subsetneq V), \quad x \in \{0,1\}^E.$$

one subtour constraint for each proper subset of vertices:
generate them on the fly



- $(\bar{P}_{42})$ solved with only 7 subtour constraints
- raising $\bar{z}$ from 641 to 697 (where optimum $z^* = 699$).

lagrangian relaxation

- lagrangian relaxation is another decomposition approach which delays some feasibility part of the problem (the dualized constraints)
- let $X = \{x_1, \dots, x_T\}$ be a finite set of vectors and

$$(P): \ z = \max\{cx \mid Ex \leq d, x \in X\}$$

- dualizing constraints $Ex \leq d$, gives upper bound u:

$$(L): \ u = \min_{\mu \geq 0} z^\mu \quad \text{where} \quad (P^\mu): \ z^\mu = \max_{x \in X}(cx - \mu(Ex - d)).$$

solving the lagrangian dual

- we proved that u is the optimum value of the following LP:

$$(L^T): \ y^T = \min\{y \mid y \geq cx^t - \mu(Ex^t - d), \forall t = 1, \dots, T, \ \mu \in \mathbb{R}^J_+, y \in \mathbb{R}\}$$

- (L) can be solved by generating and adding constraints iteratively

cutting-plane algorithm

for $t = 1, \dots, T$
  solve $(L^t) \to (y^t, \mu^t)$
  find x^{t+1} s.t. $y^t < cx^{t+1} - \mu^t(Ex^{t+1} - d)$ or stop

- lagrangian subproblem $(P^{\mu^t}): \ z^{\mu^t} = \max_{x \in X}(cx - \mu^t(Ex - d))$ returns the most violated constraint

separation problem

solve $(P^{\mu^t}) \to (x^{\mu^t})$
if $y^t < z^{\mu^t}$ then return $x^{t+1} = x^{\mu^t}$
otherwise $y^t = z^{\mu^t} = u$

solving the lagrangian dual

cutting-plane algorithm for solving the lagrangian dual

1. $t \leftarrow 0, \mu^0 \geq 0, y^0 \leftarrow -\infty,$
2. $x^{t+1} \leftarrow \texttt{opt}(P^{\mu^t}), z^{\mu^t} = cx^{t+1} - \mu^t(Ex^{t+1} - d)$
3. if $y^t \geq z^{\mu^t}$ STOP RETURN $u = y^t$
4. otherwise $(y^{t+1}, \mu^{t+1}) \leftarrow \texttt{opt}(L^t)$ GOTO 2 WITH $t \leftarrow t + 1$

lagrangian relaxation

advantages of the decomposition

1. **issues separation**: (L) is divided into two distinguishable parts
   - the lagrangian subproblem (P^{μ^t}) solves the feasibility part $x \in X$
   - the master program (L^t) manages the optimization part
2. **implicit formulation**: (L) is progressively generated and solved
   - the procedure stops before the constraints are all generated
3. **non linear formulation**: no modeling restriction for $x \in X$
   - subproblem (P^{μ^t}) can be solved by any optimization method (simplex, B&B, dynamic programming, constraint programming, local search,...)

note: solving the lagrangian dual using a subgradient algorithm has the same advantages

lagrangian relaxation

- applies to any ILP for which the lagrangian subproblem (ILP) can be solved quickly
- the intermediary values u^t are not valid upper bounds (until the last one)
- each subproblem optimum z^{μ^t} is an upper bound of u (and of z): keeps the best one currently at each iteration
- each subproblem solution x^{μ^t} is an *almost* feasible solution for (P): often easy to derive a solution / a lower bound

example: 0-1 Multi-Knapsack Problem

$$(P) : z = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij}, \quad \text{s.t.} \quad \sum_{i \in I} w_i x_{ij} \leq c_j (\forall j \in J), \quad \sum_{j \in J} x_{ij} \leq 1 (\forall i \in I), \quad x \in \{0,1\}^{I \times J}.$$

consider the lagrangian dual $(P_3) : z_3 = \min\{z_3^\mu \mid \mu \in \mathbb{R}_+^I\}$ with

$$(P_3^\mu) : z_3^\mu = \max \sum_{i \in I} \sum_{j \in J} p_i x_{ij} - \sum_{i \in I} \mu_i (\sum_{j \in J} x_{ij} - 1), \quad \text{s.t.} \quad \sum_{i \in I} w_i x_{ij} \leq c_j (\forall j \in J), \quad x \in \{0,1\}^{I \times J}.$$

Questions

1. prove that $z_3 = \min\{z_3^\mu \mid 0 \leq \mu \leq p\}$
2. consider $p = (110, 150, 70, 80, 30, 5)$, $w = (40, 60, 30, 40, 20, 5)$, and $c = (65, 85)$; run the cutting-plane algorithm starting from the optimal dual values associated with the assignment constraints in $(\bar{P})$: $\mu^0 = (30, 30, 10, 0, 0, 0)$.

example: 0-1 Multi-Knapsack Problem

Proof $z_3 = \min\{z_3^\mu \mid 0 \leq \mu \leq p\}$

We prove that for each multiplier μ, there exists $0 \leq \mu' \leq p$ s.t. $z^{\mu'} \leq z^\mu$: let $\mu'_i = \min(\mu_i, p_i)$ and associate to any optimum solution x' of $(P^{\mu'})$, the solution defined by $x_{ij} = x'_{ij}$ if $\mu_i \leq p_i$, and $x_{ij} = 0$ otherwise. Then, x is feasible for $(P^{\mu'})$ and (P^μ) and:

$$\begin{aligned}
z^{\mu'} &= \sum_{i \in I} \sum_{j \in J} (p_i - \mu'_i) x'_{ij} + \sum_{i \in I} \mu'_i \\
&= \sum_{i \mid p_i > \mu_i} \sum_{j \in J} (p_i - \mu_i) x'_{ij} + \sum_{i \mid p_i > \mu_i} \mu_i + \sum_{i \mid p_i \leq \mu_i} p_i \\
&= \sum_{i \mid p_i > \mu_i} \sum_{j \in J} (p_i - \mu_i) x_{ij} + \sum_{i \mid p_i > \mu_i} \mu_i + \sum_{i \mid p_i \leq \mu_i} p_i \\
&\leq \sum_{i \mid p_i > \mu_i} \sum_{j \in J} (p_i - \mu_i) x_{ij} + \sum_{i \mid p_i > \mu_i} \mu_i + \sum_{i \mid p_i \leq \mu_i} \mu_i \\
&\leq z^\mu
\end{aligned}$$

example: 0-1 Multi-Knapsack Problem

cutting-plane: step 0

- start with the dual solution $\mu^0 = (30, 30, 10, 0, 0, 0)$ and $y^0 = -\infty$
- solve:

$$\begin{aligned}
(P^{\mu^0}) : z^{\mu^0} &= \max \sum_{j \in J} 80x_{1j} + 120x_{2j} + 10x_{3j} + 80x_{4j} + 30x_{5j} + 5x_{6j} + 70 \\
s.t. & 40x_{11} + 60x_{21} + 30x_{31} + 40x_{41} + 20x_{51} + 5x_{61} \leq 65 \\
& 40x_{12} + 60x_{22} + 30x_{32} + 40x_{42} + 20x_{52} + 5x_{62} \leq 85 \\
& x \in \{0,1\}^{6 \times 2}.
\end{aligned}$$

- get $x_1^1 = (0, 1, 0, 0, 0, 1)$, $x_2^1 = (1, 0, 0, 1, 0, 1)$ and $z^{\mu^0} = 125 + 165 + 70 = 360$
- as $y^0 < z^{\mu^0}$, then (y^0, μ^0) violates constraint:

$$y \geq (110 - \mu_1) + (150 - \mu_2) + (80 - \mu_4) + 2(5 - \mu_6) + \sum_{i=1}^{6} \mu_i = 350 + \mu_3 + \mu_5 - \mu_6$$

example: 0-1 Multi-Knapsack Problem

cutting-plane: step 1

- solve $L^1 = \min\{y | y \geq 350 + \mu_3 + \mu_5 - \mu_6, y \in \mathbb{R}, 0 \leq \mu \leq p\}$:
- get $\mu^1 = (0,0,0,0,0,5)$ and $y^1 = 345$
- solve:

$$\begin{aligned}(P^{\mu^1}) : z^{\mu^1} = \max \sum_{j \in J} & 110x_{1j} + 150x_{2j} + 70x_{3j} + 80x_{4j} + 30x_{5j} + 5 \\ s.t. & 40x_{11} + 60x_{21} + 30x_{31} + 40x_{41} + 20x_{51} + 5x_{61} \leq 65 \\ & 40x_{12} + 60x_{22} + 30x_{32} + 40x_{42} + 20x_{52} + 5x_{62} \leq 85 \\ & x \in \{0,1\}^{6\times 2}.\end{aligned}$$

- get $x_1^2 = (0,1,0,0,0,0)$, $x_2^2 = (1,0,0,1,0,0)$ and $z^{\mu^1} = 150 + 190 + 5 = 345$
- as $y^1 \geq z^{\mu^1}$ then $u = y = z^{\mu^1} = 345$ is the optimum value of (L).

example: 0-1 Multi-Knapsack Problem

from the lagrangian solution to an optimal solution

- we get $x_1^2 = (0,1,0,0,0,0)$, $x_2^2 = (1,0,0,1,0,0)$ and $u = 345$
- note that x^2 is feasible for (P): $x_{i1}^2 + x_{i2}^2 <= 1$ for all i
- but the complementary slackness condition $\mu_6 > 0$ and $x_{61}^2 + x_{62}^2 = 1$ is not satisfied $\Rightarrow x^2$ is not optimal
- actually it is easy to transform x^2 into an optimal solution by tempting to enforce this condition:
  solution $x_1^* = (0,1,0,0,0,1)$, $x_2^* = (1,0,0,1,0,0)$ has cost 345 and is then optimal.

lagrangian decomposition

- lagrangian decomposition is a way of exhibiting coupling constraints

$$(P) : z = \max\{cx \mid Ax \leq b, Ex \leq d, x \in \mathbb{Z}_+^n\}$$

- consider a symmetric set of variables $y \in \mathbb{R}^n$ and add constraints $x = y$ then:

$$(P) : z = \max\{cx \mid Ax \leq b, Ey \leq d, x = y, x \in \mathbb{Z}_+^n, y \in \mathbb{R}_+^n\}$$

- dualize the equality constraints $x = y$

Benders decomposition

- separate a problem having two sets of variables: the easy variables y and the hard variables x

$$(P) : \quad z = \max\{cx + hy \mid Ax + Gy \leq b, x \in X \subseteq \mathbb{Z}_+^n, y \in \mathbb{R}_+^p\}$$

- consider the dual of the subproblem when the x variables are fixed:

$$(D(x)) : \quad z_x = \min\{u(b - Ax) \mid u \in Q\} \quad \text{with} \quad Q = \{u \in \mathbb{R}_+^m \mid uG \geq h\}$$

- Q is a polyhedron, then any point is a linear combination of extreme points $u^k, k \in \{1, \ldots, K\}$ and extreme rays v^j
- reformulate (P) as:

$$(PM) : \ z = \max\{cx + w \mid w \leq u^k(b - Ax)\ \forall k, v^j(b - Ax) \leq 0\ \forall j, x \in X, w \in \mathbb{R}\}$$

- this can be solved using delayed constraint generation

solving Benders decomposition

$$(MP):\ z = \max\{cx+w \mid w \leq u^k(b-Ax)\ \forall k, v^j(b-Ax) \leq 0\ \forall j, x \in X, w \in \mathbb{R}\}$$

cutting-plane algorithm

1. solve the restricted master program $\rightarrow (\bar{x}, \bar{w})$
2. solve $(D(\bar{x}))$: if unbounded ($\bar{x}$ does not belong to any feasible solution of (P)) get an extrem ray v^j s.t. $v^j(b-Ax) > 0$
3. otherwise get u^k an optimum solution with cost $z_{\bar{x}} = u^k(b - A\bar{x}) \leq \bar{z}$.
4. if $\bar{z} = z_{\bar{x}}$ then $(\bar{x}, \bar{z})$ is optimum for (MP).

- the convergence of this algorithm may be rather slow
- but it gives an LB and an UB at each iteration: $c\bar{x} + \bar{z} \geq z \geq c\bar{x} + z_{\bar{x}}$

Outline

1 delayed constraint generation

2 delayed column generation
- column generation for LP
- example: cutting-stock problem
- Dantzig-Wolfe decomposition
- Branch-and-Price

column generation for LP

- the column generation approach is dual to the cutting-plane approach
- when a program has too many variables and relatively few constraints then generate and add the variables progressively to the model
- slave subproblem: finds variables and columns (LP-matrix coefficients) to add to improve the restricted master program
- applies to LP only (but part of integrality conditions can be modeled within the subproblem)

column generation for LP

$$(\bar{P}): \bar{z} = max\{\sum_{i \in I} c_i x_i \mid \sum_{i \in I} a_{ij} x_i = b_j\ \forall j \in J, x_i \geq 0\ \forall i \in I\}$$

- if $|J| << |I|$ then any optimal (basic) solution $\bar{x}$ contains many 0 (nonbasic) values

the revised simplex algorithm

1. choose initial basis $B = I \setminus N$ (A_B: square nonsingular submatrix of A)
2. get the basic solution: $x_N = 0, x_B = A_B^{-1}b - A_B^{-1}A_N x_N = A_B^{-1}b$ with cost $z = c_B A_B^{-1} b + (c_N - c_B A_B^{-1} A_N) x_N = c_B A_B^{-1} b$
3. choose entering nonbasic variable $i \in N$: find column (A_i, c_i) with positive reduced cost $\bar{c}_i = c_i - c_B A_B^{-1} A_i > 0$ or stop (x is optimal)
4. choose a leaving basic variable $b \in B$ then update basis $B \leftarrow B \cup \{i\} \setminus \{b\}$

- column generation is a variant of the revised simplex algorithm where the set of nonbasic variables x_N and columns (A_N, c_N) is maintained implicitely

column generation for LP

the column generation algorithm

1. start from an initial feasible column subset $I_r \subseteq I$ and solve the restricted LP $(\bar{P}_r)$
2. find entering improving (nonbasic) variable $i \in I \setminus I_r$ with positive reduced cost $\bar{c}_i > 0$
3. update $I_r \leftarrow I_r \cup \{i\}$ or stop ($\bar{x}_r$ is optimal for $(\bar{P})$)

- the slave subproblem (step 2) is called the pricing problem
- implicit formulation: the process stops before all columns are generated

dual interpretation

- the restricted master program $(\bar{P}_r)$ is $(\bar{P})$ with additional constraints $x_i = 0 \; \forall i \in I \setminus I_r$: solution $(\bar{x}_r, 0)$ is feasible for $(\bar{P})$
- the restricted dual $(\bar{D}_r)$ is a relaxation of

$$(\bar{D}): \; \bar{u} = min\{by \mid A_i y \geq c_i \; \forall i \in I, y \in \mathbb{R}^J\}$$

- $\bar{x}_r$ is optimal for $(\bar{P})$ if the dual solution $\bar{y}_r$ of $(\bar{D}_r)$ is feasible for $(\bar{D})$:

$$\bar{z} = \bar{u} \leq \bar{u}_r = \bar{z}_r \leq \bar{z}$$

- the pricing problem is then to find constraints violated in the dual:

$$(SP) = \{i \in I \mid A_i \bar{y}_r < c_i\} = \{i \in I \mid \bar{c}_i = c_i - c_B A_B^{-1} A_i = c_i - A_i \bar{y}_r > 0\}$$

- $\bar{z}_r$ approximates $\bar{z}$ from below (improved at each iteration)

pricing problem

find an improving column of (A, c) s.t. $c_i - A_i \bar{y}_i > 0$

- like lagrangian relaxation, it allows to separate a set of easy constraints from the rest of a large LP
- easy constraints involve a small fraction of the variables, or they have a special structure which can be solved quickly using a special algorithm
- can also be approximated (until no column is found)
- no modelling restriction
- no objective function but we aim at finding the most promising column: i.e. that maximizes $c_i - A_i \bar{y}_i$
- solving pricing problem at optimality gives an UB
- in practice, generate several columns at each iteration

example: cutting-stock problem

cutting-stock problem

Paper is produced in large rolls (*raws*) with a fixed width W. Customers desire to have rolls (*finals*) in a variety of smaller widths: b_k rolls of width $w_k \; \forall k = 1, \ldots, m$. Determine how to cut the raws into final rolls so that the fewest number of raw rolls are used.

Questions

- find a trivial UB
- show that the bin-packing formulation gives a LB not better than $\sum_k b_k w_k / W$
- consider a set-packing formulation by enumerating all possible patterns that may be cut from a raw
- consider only maximal patterns
- derive a feasible solution from this LP relaxation

Dantzig-Wolfe decomposition

- many problems has set-packing / -covering / - partitioning formulations to which column generation applies
- Dantzig-Wolfe decomposition is a systematic approach to separate a set of easy constraints from the rest of a large ILP

Dantzig-Wolfe decomposition

$$(\bar{P}) : \bar{z} = max\{cx \mid Ax \leq b, x \in conv(X)\}$$

let $x_1^*, \dots, x_s^*$ be the extreme points of the (finite) polytope of X then by convexity:

$$\bar{z} = max\{(cx^*)\lambda \mid (Ax^*)\lambda \leq b, \sum_{k=1}^{s} \lambda_s = 1, \lambda \in \mathbb{R}_+^s\}$$

- the extreme points (or rays) can be progressively generated using column generation
- this approach is dual to the solving the lagrangian dual by constraint generation

branch-and-price

- branch-and-price: extension of branch-and-bound where the LP relaxation is solved by column generation at each node
- the branching strategy must be compatible with the pricing problem: i.e. not modify its structure
- for binary variables: use GUB constraints
- for integer variables (volume or quantity), integrality is often not required
- a truncated branch-and-price: solve the column generation at the root node then run the b&b on the generated columns alone