Section Notes 9

Midterm 2 Review

Applied Math / Engineering Sciences 121

Week of November 28, 2016

The following list of topics is an overview of the material that was covered in the lectures and sections (with some specific examples).

Contents

1 Lecture 11: Integer Programming

- Integer program: A linear program (objective and constraints linear) that has the additional requirement that all variables can only take integer values.
  - binary programs (all variables 0 or 1)
  - mixed integer programs (can include integer variables, binary variables, and continuous variables)
- Many potential applications
  - Weighted knapsack
  - Set covering
  - Traveling salesperson problem
- More powerful modeling tool than LPs.
- Modeling tricks (see Section 6 and 7 notes for more examples)
  - At least one (choose appropriate M_i)

$$\begin{aligned} f_1(x) &\le b_1 \\ f_2(x) &\le b_2 \end{aligned}$$

$$\begin{aligned} f_1(x) - M_1\alpha_1 &\le b_1 \\ f_2(x) - M_2\alpha_2 &\le b_2 \\ \alpha_1 + \alpha_2 &\le 1 \\ \alpha_1, \alpha_2 &\in \{0, 1\} \end{aligned}$$

  - Either-or (but not both)

$$\begin{aligned} f_1(x) &\le b_1 \\ f_2(x) &\ge b_2 \end{aligned}$$

$$\begin{aligned} f_1(x) - M_1\alpha_1 &\le b_1 \\ f_2(x) + M_2\alpha_2 &\ge b_2 \\ \alpha_1 + \alpha_2 &= 1 \\ \alpha_1, \alpha_2 &\in \{0, 1\} \end{aligned}$$

  - If (A)-then (B)
    Note that it is equivalent to model at least one of: not(A), B. Thus:

$$f_1(x) > b_1 \Rightarrow f_2(x) \le b_2$$

$$f_1(x) \leq b_1 \text{ (not A)}$$
$$f_2(x) \leq b_2 \text{ (B)}$$

$$\begin{aligned} f_1(x) - M_1\alpha_1 &\leq b_1 \\ f_2(x) - M_2\alpha_2 &\leq b_2 \\ \alpha_1 + \alpha_2 &\leq 1 \\ \alpha_1, \alpha_2 &\in \{0, 1\} \end{aligned}$$

- Big-M parameters
  - Select M to be as small as possible while still allowing constraints to be satisfied

2 Lecture 13: Branch and Bound I

- LP relaxations
- Weak duality of IPs
- Branch and bound algorithm (for a maximization problem):
  1. *Initialization* Put initial problem S into list of open problems, set lower bound $\underline{z} = -\infty$ (value of best incumbent found so far).
  2. *Termination* If no more problems in list, terminate.
  3. *Choosing a node* Chose one of the open problems S^i.
  4. *Optimizing* Solve LP relaxation of S^i, obtain dual bound $\overline{z}^i = LPvalue$.
  5. *Fathoming*
     - If LP relaxation is infeasible, then fathom this subproblem
     - If $\overline{z}^i \leq \underline{z}$, fathom by bound
     - If solution to LP relaxation is integer, update lower bound $\underline{z} := \max(\underline{z}, \overline{z}^i)$ and fathom by integrality (updating incumbent solution if necessary)
  6. *Branching* If cannot fathom the subproblem then create two new subproblems S_1^i and S_2^i and add them to the list.
  7. Return to step 2.

3 Lecture 14 (and 12): Branch and Bound II and Formulation Strength

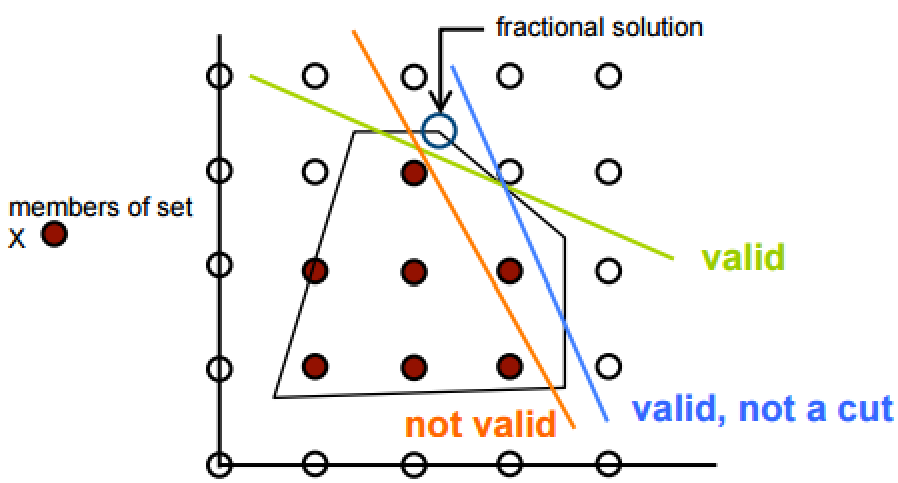
- Node selection decision

- Depth-first
- Best-bound first

- Dual simplex to pivot with new constraints
- Branching decision (e.g. most-fractional variable, strong branching)
- Formulation strength
  - If P_1 and P_2 are valid formulations for an IP (meaning that they include all feasible, integer solutions and only them), and $P_1 \subset P_2$, then we say P_1 is a stronger formulation than P_2.
  - Convex hull (every extreme point of the polyhedron is a feasible, integer solution of the IP)
  - The most important reason for preferring stronger formulations is that they allow for more search nodes to be fathomed by bound (important for avoiding enumeration!).

4 Lecture 15: Cutting plane methods

- Valid inequality: a constraint that is satisfied by all integer, feasible solutions of an IP
- Cut: a valid inequality that separates the current fractional solution (i.e., it is not satisfied by this fractional solution)



- Chvátal-Gomory inequalities
  - Apply to "≤" inequalities and a feasible set X that is a subset of the non-negative integers.

  - Obtained by (i) forming a positive, linear combinations of inequalities, (ii) taking the floor of coefficients on LHS, and (iii) taking the floor of the coefficients RHS. This last step (iii) is valid given that the left hand side value will always be integer since X is a subset of integers.

- Cutting plane method: generating a sequence of cuts, each cut separating the current, fractional optimal solution.
- Gomory's cutting plane algorithm
  - Requires the IP to have only integer coeffs and integer RHS values, and non-negative, integer variables.
  - Repeatedly generates a cut from one of the fractional rows in the current optimal tableau.
  - Gomory's algorithm is complete for IPs with integer coefficients and RHS values.

5 Lecture 16: More cuts, Branch and Cut

- Cover inequalities
  - Defined for a 0-1 knapsack set: $X = \{x \in \{0,1\}^n : \sum_{j=1}^n a_j x_j \leq b\}$, for positive coefficients $a_j \geq 0$ and positive RHS value $b \geq 0$.
  - A set $C \subseteq \{1, \ldots, n\}$ is a *cover* if $\sum_{j \in C} a_j > b$. A cover is *minimal* if $C \setminus j$ is not a cover for any $j \in C$.
  - If C is a cover, then the *cover inequality* $\sum_{j \in C} x_j \leq |C| - 1$ is valid.
  - An auxiliary problem can be used to find a separating cover inequality
- *Extended cover inequalities*: extend C by adding variables $j \notin C$ for which $a_j \geq a_i$ for all $i \in C$. Also valid.
- *Lifted cover inequalities*: make the multiplier $\alpha_j x_j$ ($\alpha_j > 0$) on variable x_j in an (extended) cover inequality as large as possible, such that the inequality is still valid. Again, solve an auxiliary problem to find α_j.
- Lifted cover inequalities are complete for IPs with constraints that are 0-1 knapsack sets.
- *Branch-and-cut* method: if cannot fathom a node, then add cuts to automatically strengthen the formulation. Still branch and bound, but also generate cuts ('global' cuts if at the root node, 'local' cuts otherwise.)
- Other tricks: Preprocessing, Primal heuristics, SOS branching

6 Lecture 17: Markov Chains

- Stochastic process and Markov chain

- Markovian property
- Stationary transition probabilities

- States $S = \{0, \ldots, m-1\}$, $m \times m$ transition matrix P with p_{ij} for probability of transition from $i \in S$ to $j \in S$.
- Definitions:
  - Accessible, communicate, class
  - Irreducible (= one class)
  - Class properties:
    * Recurrent or transient (not every state in a chain can be transient)
    * Periodic (d, the largest integer such that the chain can only return to a state at multiples of d is greater than 1) or aperiodic ($d = 0$)
  - Ergodic: an irreducible (thus, recurrent) chain that is aperiodic
  - n-step transition $p_{ij}^{(n)}$ is conditional probability of state j after n steps from state i. Compute as $P^{(n)} = P^n$.
  - *Steady-state probability* $\pi_j = \lim_{n\to\infty} p_{ij}^{(n)} > 0$ exists in an ergodic chain (and is independent of i). The probability the process is in state j after a large number of transitions.
  - Computed via *balance equations*:

$$\pi_j = \sum_{i=0}^{m-1} \pi_i \cdot p_{ij} \quad \forall\, j = 0, ..., m-1 \tag{1}$$

$$\sum_{j=0}^{m-1} \pi_j = 1 \tag{2}$$

7 Lecture 18: Markov Decision Processes

- MDP model
  - states $S = \{0, \ldots, m-1\}$
  - actions $A = \{0, \ldots, n-1\}$
  - reward function $R(s, a)$ (immediate reward for action a in state s)
  - transition function $P(s, a, s') \in [0, 1]$ (conditional prob of transitioning to state s' given action a in state s)
- Policy μ: mapping from states to actions
  - Deterministic (each state maps to single action)

- Stochastic (each state maps to distribution on actions)

- The MDP defines the problem to solve. Given an MDP, we need to solve and find the optimal policy.
- For a particular policy, an MDP reduces to a Markov chain. An *ergodic MDP* is one for which the associated Markov chain is ergodic for every possible policy.
- The expected total reward over the next t time periods when following policy μ from state s is:

$$V_\mu^{(t)}(s) = \mathbf{E}_{s^1,\ldots,s^{t-1}}\left[R(s^0,\mu(s^0)) + R(s^1,\mu(s^1)) + \ldots + R(s^{t-1},\mu(s^{t-1}))\right], \quad (3)$$

  where $s^0 = s$, and s^τ denotes the state reached under policy μ after τ steps.
- The optimal policy from state s maximizes the *MDP value function* $V_\mu(s)$. The value function defines the objective value of policy μ from state s.
- There are two possible definitions:
  - *Limit-average reward*:

$$V_\mu(s) = \lim_{t\to\infty}\left[\frac{1}{t}V_\mu^{(t)}(s)\right], \quad (4)$$

    which is only well defined when state s is in a recurrent, aperiodic class. For an ergodic MDP, the limit-average reward for policy μ from state s can be computed as

$$V_\mu(s) = \sum_{s'} R(s',\mu(s'))\pi_\mu(s'), \quad (5)$$

    where $\pi_\mu(s')$ is the steady state probability of state s' under policy μ. Note: this does not depend on state s.
  - *Expected discounted reward* (for a *discount factor* γ, $0 < \gamma < 1$):

$$V_\mu(s) = \mathbf{E}_{s^1,s^2,\ldots}\left[R(s^0,\mu(s^0)) + \gamma\cdot R(s^1,\mu(s^1)) + \gamma^2\cdot R(s^2,\mu(s^2)) + \ldots + \right], \quad (6)$$

    where $s^0 = s$, and s^τ denotes the state reached under policy μ after τ steps.
- An LP can be formulated to maximize either the limit-average reward (for an ergodic MDP) or the expected-discounted reward, in order to find an optimal policy. Optimal policies are deterministic.
- Fundamental Theorem of MDPs: under the discounted reward criterion, there exists a policy that is uniformly optimal (i.e., it is optimal from every state).
- This 'uniform optimality' is also true under the limit-average reward criterion for an ergodic MDP.