Lagrangian Relaxation for MAP Inference

SPFLODD

October 8, 2013

Outline

- An elegant example of a relaxation to TSP
- A common problem in NLP: finding consensus
- Basic Lagrangian relaxation
- Solving the problem with subgradient
- AD^3: an alternative approach to decomposition and optimization using the augmented Lagrangian

Traveling Salesman Problem

- Given: a graph (V, E) with edge weight function θ
- Tour: a subset of E corresponding to a path that starts and ends in the same place, and visits every other node exactly once.
- TSP: Find the maximum-scoring tour.
  - NP-hard

$$\max_{y \in \mathcal{Y}_{\text{tour}}} \sum_{e \in E} y_e \theta_e$$

Another Problem

- 1-tree: a tree on edges for {2, ..., |V|}, plus two edges from E that link the tree to vertex 1.
  - All tours are 1-trees.
  - All 1-trees where every vertex has degree 2 are tours.
  - Easy to solve.

Held and Karp (1971)

$$\mathcal{Y}_{\text{tour}} = \left\{ y : y \in \mathcal{Y}_{\text{1-tree}} \wedge \forall i \in \{1, \ldots, |V|\}, \sum_{e:i \in e} y_e = 2 \right\}$$

$$\max_{y \in \mathcal{Y}_{\text{tour}}} \sum_{e \in E} y_e \theta_e$$

transforming the constraints

$$\max_{y \in \mathcal{Y}_{\text{1-tree}}} \sum_{e \in E} y_e \theta_e \text{ s.t. } \forall i, \sum_{e:i \in e} y_e = 2$$

Lagrangian dual

$$L(u) = \max_{y \in \mathcal{Y}_{\text{1-tree}}} \sum_{e \in E} y_e \theta_e + \sum_{i=1}^{|V|} u_i \left(\sum_{e:i \in e} y_e - 2 \right)$$

LR Algorithm for TSP

1. Initialize $u^{(0)} = 0$
2. Repeat for k = 1, 2, ...:

$$y^{(k)} \leftarrow \arg \max_{y \in \mathcal{Y}_{\text{1-tree}}} \sum_{e \in E} y_e \theta_e + \sum_{i=1}^{|V|} u_i^{(k-1)} \left(\sum_{e:i \in e} y_e - 2 \right)$$

$$\forall i, u_i^{(k)} \leftarrow u_i^{(k-1)} - \delta_k \left(\sum_{e:i \in e} y_e - 2 \right)$$

If this converges to a solution that satisfies the constraints, it is a solution to the TSP.

Lagrangian Relaxation, More Generally

- Assume a linear scoring function that is "hard" to maximize.

$$\max_{\boldsymbol{y} \in \mathcal{Y}} \boldsymbol{\theta}^\top \boldsymbol{y}$$

- Rewrite the problem as something easier, with linear constraints (relaxation): $\mathcal{Y} = \{\boldsymbol{y} \in \mathcal{Y}' : \mathbf{A}\boldsymbol{y} = \mathbf{b}\}$

$$\max_{\boldsymbol{y} \in \mathcal{Y}'} \boldsymbol{\theta}^\top \boldsymbol{y}$$

$$\text{s.t. } \mathbf{A}\boldsymbol{y} = \mathbf{b}$$

- Tackle the dual problem:

$$\min_{\mathbf{u}} \max_{\boldsymbol{y} \in \mathcal{Y}'} \boldsymbol{\theta}^\top \boldsymbol{y} + \mathbf{u}^\top (\mathbf{A}\boldsymbol{y} - \mathbf{b})$$

Theory

- The dual function (of **u**) upper bounds the MAP problem.
- A subgradient algorithm can be applied to minimize the dual; it will converge in the limit.
- If the solution to the dual problem satisfies the constraints, it is also a solution to the primal (relaxed) problem ($\mathcal{Y}'$).
  - If the relaxation is *tight*, we also have a solution to the original primal problem ($\mathcal{Y}$).

Dual Decomposition (A Special Case of LR)

- Assume the objective decomposes into two parts, coupled only through the linear constraints:

$$\max_{\boldsymbol{y}\in\mathcal{Y},\boldsymbol{z}\in\mathcal{Z}} \boldsymbol{\theta}^\top \boldsymbol{y} + \boldsymbol{\psi}^\top \boldsymbol{z}$$

$$\text{s.t.} \;\; \mathbf{A}\boldsymbol{y} + \mathbf{C}\boldsymbol{z} = \mathbf{b}$$

- The relaxation:

$$\max_{\boldsymbol{y}\in\mathcal{Y},\boldsymbol{z}\in\mathcal{Z}} \boldsymbol{\theta}^\top \boldsymbol{y} + \boldsymbol{\psi}^\top \boldsymbol{z} \equiv \left(\max_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{\theta}^\top \boldsymbol{y}, \max_{\boldsymbol{z}\in\mathcal{Z}} \boldsymbol{\psi}^\top \boldsymbol{z}\right)$$

Dual Decomposition

$$\min_{\mathbf{u}} \max_{\boldsymbol{y}\in\mathcal{Y},\boldsymbol{z}\in\mathcal{Z}} \boldsymbol{\theta}^\top \boldsymbol{y} + \boldsymbol{\psi}^\top \boldsymbol{z} + \mathbf{u}^\top \left(\mathbf{A}\boldsymbol{y} + \mathbf{C}\boldsymbol{z} - \mathbf{b}\right)$$

1. Initialize **u**$^{(0)}$ = 0
2. Repeat for k = 1, 2, ...:

$$\boldsymbol{y}^{(k)} \leftarrow \max_{\boldsymbol{y}\in\mathcal{Y}} \boldsymbol{\theta}^\top \boldsymbol{y} + \mathbf{u}^{(k-1)\top}\mathbf{A}\boldsymbol{y}$$

$$\boldsymbol{z}^{(k)} \leftarrow \max_{\boldsymbol{z}\in\mathcal{Z}} \boldsymbol{\psi}^\top \boldsymbol{z} + \mathbf{u}^{(k-1)\top}\mathbf{C}\boldsymbol{z}$$

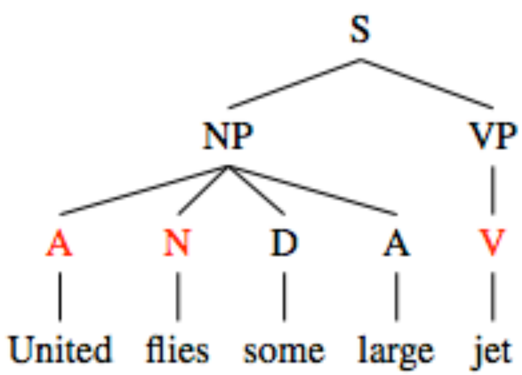
$$\mathbf{u}^{(k)} \leftarrow \mathbf{u}^{(k-1)} - \delta_k \left(\mathbf{A}\boldsymbol{y}^{(k)} + \mathbf{C}\boldsymbol{z}^{(k)} - \mathbf{b}\right)$$

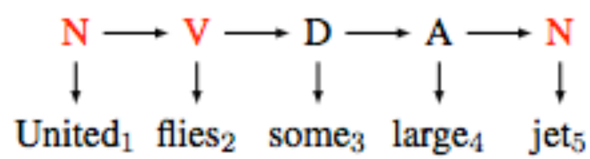
Consensus Problems in NLP

- Key example:
  - Find the jointly-best parse (under a WCFG) and sequence labeling (under an HMM); see Rush et al. (2010)
- Other examples:
  - Finding a lexicalized phrase structure parse that is jointly-best under a WCFG and a dependency model (Rush et al., 2010)
  - Decoding in phrase-based translation (Chang and Collins, 2011).

Example Run (k = 1)

$$\forall i \in \{1, \ldots, n\}, \forall N \in \mathcal{N}, \boldsymbol{y}[N, i, i] = \boldsymbol{z}[N, i]$$





$$\mathbf{u}[A, 1] = -1$$

$$\mathbf{u}[N, 2] = -1$$

$$\mathbf{u}[N, i]^{(1)} = \mathbf{u}[N, i]^{(0)} - \delta_k \left(\boldsymbol{y}[N, i, i]^{(1)} - \boldsymbol{z}[N, i]^{(1)} \right)$$

$$\mathbf{u}[V, 5] = -1$$

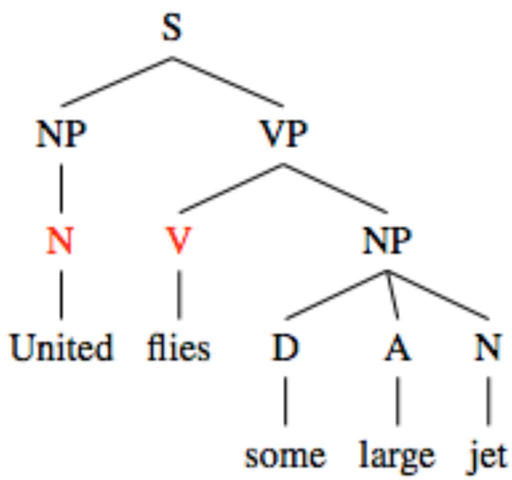
$$\mathbf{u}[N, 1] = 1$$

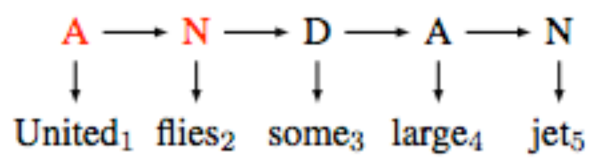
$$\mathbf{u}[V, 2] = 1$$

$$\mathbf{u}[N, 5] = 1$$

Example Run (k = 2)

$$\forall i \in \{1, \ldots, n\}, \forall N \in \mathcal{N}, \boldsymbol{y}[N, i, i] = \boldsymbol{z}[N, i]$$





$$\downarrow \mathbf{u}[\mathrm{N}, 1]$$

$$\mathbf{u}[N, i]^{(2)} = \mathbf{u}[N, i]^{(1)} - \delta_k \left(\boldsymbol{y}[N, i, i]^{(2)} - \boldsymbol{z}[N, i]^{(2)} \right)$$

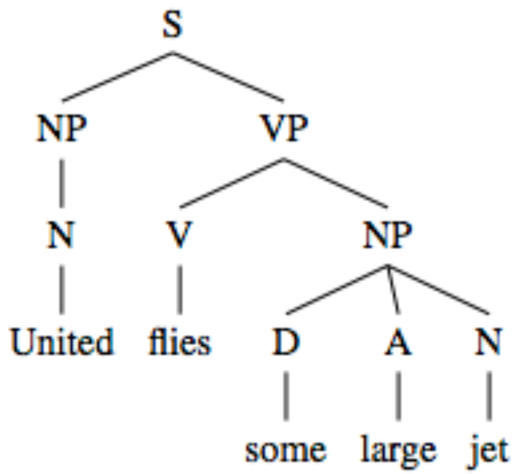
$$\downarrow \mathbf{u}[\mathrm{V}, 1]$$

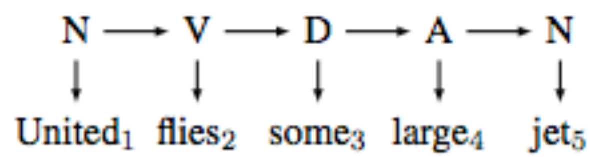
$$\uparrow \mathbf{u}[\mathrm{A}, 1]$$

$$\uparrow \mathbf{u}[\mathrm{N}, 1]$$

Example Run (k = 3)

$$\forall i \in \{1, \ldots, n\}, \forall N \in \mathcal{N}, \boldsymbol{y}[N, i, i] = \boldsymbol{z}[N, i]$$





“Certificate”

- Proof that we have solved the original problem: constraints hold.
  - This is easy to check given **y** and **z**.
- In published NLP papers so far, this happens most of the time (better than 98%).

What can go wrong?

- It can take many iterations to converge.
- Oscillation between different solutions; failure to agree.
  - Suggested solution: add more variables for “bigger parts” and enforce agreement among them with more constraints.

What does this have to do with ILP?

- The linear constraints are expressed in terms of an integer-vector representation of the output space.
  - Just like when we treated decoding as an ILP.
- The subproblems *could* be expressed as ILPs, though we'd prefer to use poly-time combinatorial algorithms to solve them if we can.

Consensus Problems, Revisited

- What if we just have a hard combinatorial optimization problem?
  - There isn't always a straightforward decomposition.
- Martins et al. (2011): shatter a decoding problem into *many* "small" subproblems (instead of two "big" ones).
  - Instead of dynamic programming as a subroutine, LP relaxations of "small" subproblems.
  - Extra LP relaxation step.

Martins' Alternative Formulation

- Original problem:

$$\max_{\boldsymbol{y}_1 \in \mathcal{Y}_1, \ldots, \boldsymbol{y}_S \in \mathcal{Y}_S, \boldsymbol{w} \in \mathbb{R}^D} \sum_{s=1}^{S} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s$$

$$\text{s.t. } \forall s, \mathbf{A}_s \boldsymbol{w} = \boldsymbol{y}_s$$

- Convex relaxation:

$$\max_{\boldsymbol{y}_1 \in \text{conv}(\mathcal{Y}_1), \ldots, \boldsymbol{y}_S \in \text{conv}(\mathcal{Y}_S), \boldsymbol{w} \in \mathbb{R}^D} \sum_{s=1}^{S} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s$$

$$\text{s.t. } \forall s, \mathbf{A}_s \boldsymbol{w} = \boldsymbol{y}_s$$

- Dual:

$$\min_{\mathbf{u}_1, \ldots, \mathbf{u}_S} \max_{\boldsymbol{y}_1 \in \text{conv}(\mathcal{Y}_1), \ldots, \boldsymbol{y}_S \in \text{conv}(\mathcal{Y}_S), \boldsymbol{w} \in \mathbb{R}^D} \sum_{s=1}^{S} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \sum_s \mathbf{u}_s^\top (\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w})$$

Augmented Lagrangian (Hestenes, 1969; Powell, 1969)

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_S} \max_{\boldsymbol{y}_1 \in \mathrm{conv}(\mathcal{Y}_1),\ldots,\boldsymbol{y}_S \in \mathrm{conv}(\mathcal{Y}_S),\boldsymbol{w} \in \mathbb{R}^D} \sum_{s=1}^{S} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \sum_s \mathbf{u}_s^\top \left(\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\right) + \frac{\rho}{2} \sum_s \|\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\|_2^2$$

$$\uparrow$$

$$\min_{\mathbf{u}_1,\ldots,\mathbf{u}_S} \max_{\boldsymbol{y}_1 \in \mathrm{conv}(\mathcal{Y}_1),\ldots,\boldsymbol{y}_S \in \mathrm{conv}(\mathcal{Y}_S),\boldsymbol{w} \in \mathbb{R}^D} \sum_{s=1}^{S} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \sum_s \mathbf{u}_s^\top \left(\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\right)$$

Alternating Directions Method of Multipliers

(Gabay and Mercier, 1976; Glowinski and Marroco, 1975)

Dual Decomposition (AD^3)

- Alternate between updating **y** and **w**:

$$\forall s, \boldsymbol{y}_s \leftarrow \arg \max_{\boldsymbol{y}_s \in \mathrm{conv}(\mathcal{Y}_s)} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \mathbf{u}_s^\top \boldsymbol{y}_s + \frac{\rho}{2} \|\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\|_2^2$$

$$\boldsymbol{w} \leftarrow \arg \max_{\boldsymbol{w}} \sum_s \mathbf{u}_s^\top \mathbf{A}_s \boldsymbol{w} + \frac{\rho}{2} \sum_s \|\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\|_2^2$$

- Subgradient step for dual variables **u** is similar to before:

$$\forall s, \mathbf{u}_s^{(k)} \leftarrow \mathbf{u}_s^{(k-1)} - \delta_k \left(\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w} \right)$$

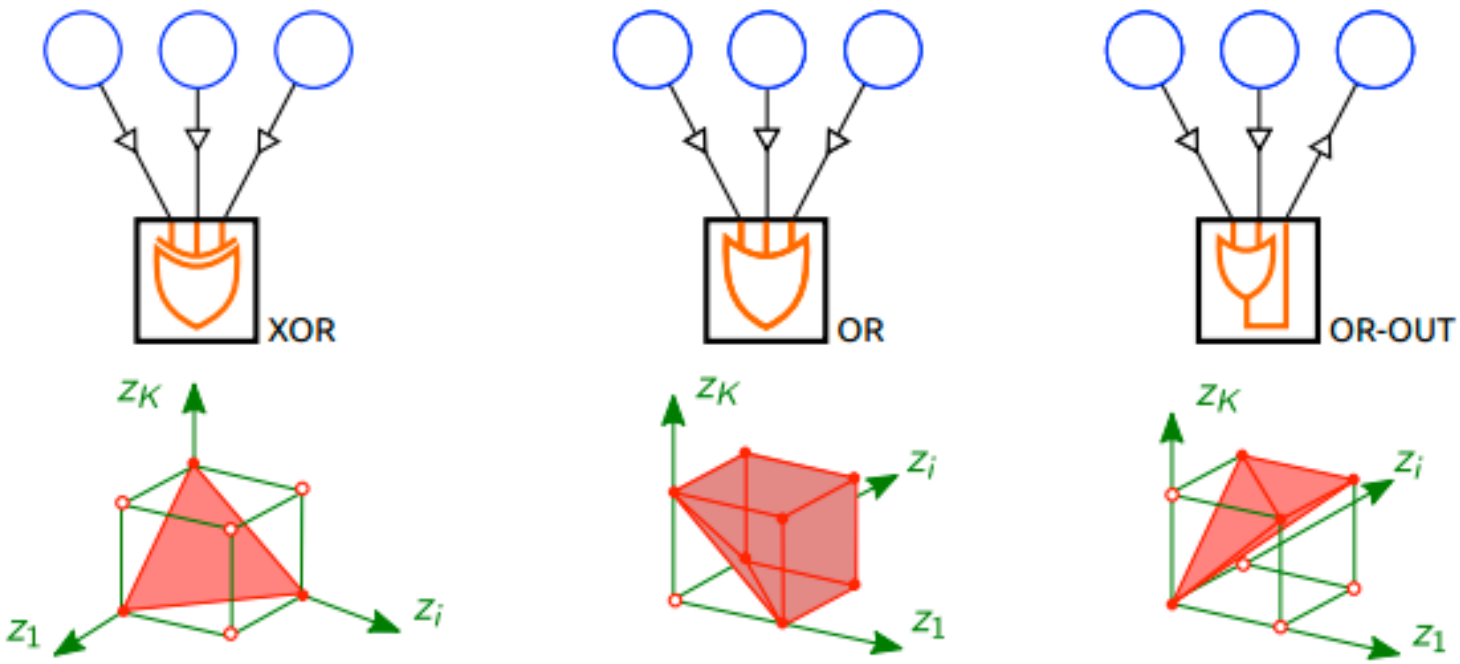
Massive Decomposition

- Most extreme: every factor (MN) or "part" is a separate subproblem.

$$\forall s, \boldsymbol{y}_s \leftarrow \arg \max_{\boldsymbol{y}_s \in \mathrm{conv}(\mathcal{Y}_s)} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \mathbf{u}_s^\top \boldsymbol{y}_s + \frac{\rho}{2} \|\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\|_2^2$$

- Some kinds of MN factors can be solved very efficiently ...

XOR, OR, OR-with-Output Solvable in O(K log K)



AD^3 and "Big" Subproblems?

- Return to Rush and Collins' example.
  - One subproblem is "WCFG" and one is "HMM tagger."

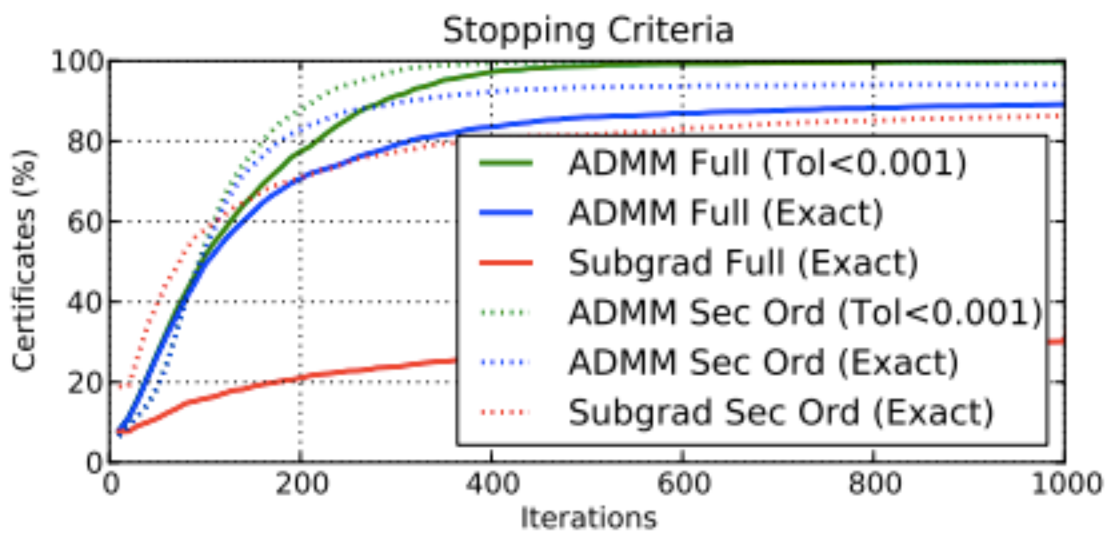
$$\forall s, \boldsymbol{y}_s \leftarrow \arg \max_{\boldsymbol{y}_s \in \mathrm{conv}(\mathcal{Y}_s)} \boldsymbol{\theta}_s^\top \boldsymbol{y}_s + \mathbf{u}_s^\top \boldsymbol{y}_s + \frac{\rho}{2} \|\boldsymbol{y}_s - \mathbf{A}_s \boldsymbol{w}\|_2^2$$

  - In dependency parsing, "max arborescence" might be a subproblem.
  - Why can't we use AD^3?

Pros and Cons

- Con: Subproblems are now *quadratic*.
  - Linear decoders as subroutines?
- Con: Fractional solutions.
- Pro: Better stopping criteria: residuals.
  - Primal residuals measure amount by which primal constraints are violated.
  - Dual residuals measure amount by which dual optimality is violated.
- Pro: Certificates as before (for each s, $\mathbf{A}_s\mathbf{w} = \mathbf{y}_s$)

Convergence of AD^3 vs. Subgradient



Dependency parsing:

- ADMM = AD^3
- Sec Ord = Second order model for which subgradient optimization is possible
- Full = second order model with all-siblings, directed paths, and non-projective arcs

Take-Home Messages

- Dual decomposition is useful for consensus problems.
  - Subgradient DD when there are a few subproblems with good specialized solvers.
  - AD^3 when you've got a big problem with lots of hard and soft constraints. (There is a library.)
- Attractive guarantees (cf. beam search).
- Only MAP inference.

References

- "A tutorial on dual decomposition and Lagrangian relaxation for inference in natural language processing," by A. Rush and M. Collins, *JAIR* 45:305-362, 2013.
- "Alternating directions dual decomposition" by A. Martins et al., arXiv 1212.6550.