Introduction to Dual Decomposition for Inference

David Sontag, Amir Globerson, Tommi Jaakkola

Part I: dual decomposition and subgradient method

Ke Jiang

October 16, 2012

The Problem

MAP inference: finding an assignment $\mathbf{x} = (x_1, \ldots, x_n)$ which satisfies:

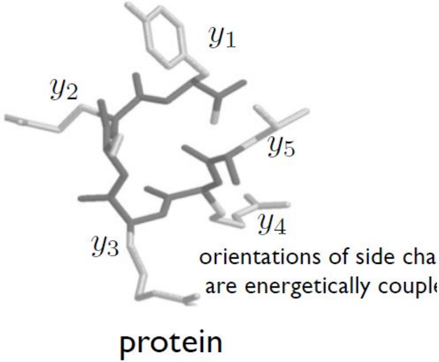
$$\mathrm{MAP}(\theta) = \max_x \left(\sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{x}_f) \right)$$

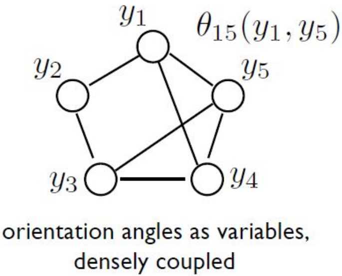
- $x_1, \ldots, x_n$ are a set of discrete variables, $V = \{1, \ldots, n\}$
- F is the set of subsets on these variables, where each subset corresponds to one of the factors
- $\theta_f(\mathbf{x}_f)$ are the functions on the factors, and $\theta_i(x_i)$ are the functions on the individual variables

Motivation - Protein Structure Prediction

Goal: recover energetically optimal amino acid side-chain orientations in a fixed protein backbone structure

$$\max_{y} \Big(\sum_{i \in V} \theta_i(y_i) + \sum_{(i,j) \in E} \theta_{ij}(y_i, y_j) \Big)$$

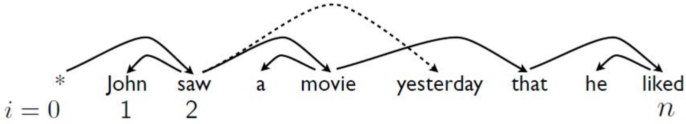




(Note: The graphs are adapted from Tommi Jaakkola's slides)

Motivation - Dependency Parsing

Goal: predict the dependency tree (highest scoring) that relates the words in the sentence



$$\max_x \Big((\theta_T(\mathbf{x})) + (\sum_{ij} \theta_{ij}(x_{ij}) + \sum_i \theta_{i|}(\mathbf{x}_{|i})) \Big) = \theta_1(\mathbf{x}) + \theta_2(\mathbf{x})$$

- $x_{ij} \in \{0, 1\}$ are the binary arc selection variables
- $\theta_T(\mathbf{x})$ enforces the selections must form a directed tree, with $\theta_T(\mathbf{x}) = -\infty$ for non-trees, $\theta_T(\mathbf{x}) = 0$ otherwise
- $\theta_{ij}(x_{ij})$ are the weight on the arcs
- $\theta_{i|}(\mathbf{x}_{|i})$ is the higher order interactions between the arc selections for a given word i, where $\mathbf{x}_{|i} = \{x_{ij}\}_{j \neq i}$ (all outgoing edges) is the modifier selections

(Note: The graph is adapted from Tommi Jaakkola's slides)

Algorithms for MAP Inference

- **Generally**, finding the MAP assignment of a graphical model is **NP-hard**, even if the local functions only depend on two variables (protein structure prediction)

Algorithms for MAP Inference

- **Generally**, finding the MAP assignment of a graphical model is **NP-hard**, even if the local functions only depend on two variables (protein structure prediction)
- **Simple dependencies**: some combinatorial algorithms can provide exact inference
  - dynamic programming: tree-strcutured Markov random field
  - maximum spanning tree: dependency parsing without higher order interactions

Algorithms for MAP Inference

- **Generally**, finding the MAP assignment of a graphical model is **NP-hard**, even if the local functions only depend on two variables (protein structure prediction)
- **Simple dependencies**: some combinatorial algorithms can provide exact inference
  - dynamic programming: tree-strcutured Markov random field
  - maximum spanning tree: dependency parsing without higher order interactions
- **Complex dependencies**: approximation algorithms often work well in practice using **relaxation** of the original MAP problem
  - pose it as a constrained optimization problem
  - relax some of the constraints in order to factor the problem into more independent subproblems

Dual Decomposition I

MAP inference as combined optimization

$$\max \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{y}_f^f)$$

such that $x_i = y_i^f$ for all $i = 1, \ldots, n, f \in F$, where y_i^f is the same discrete variable as x_i in factor f.

Dual Decomposition I

MAP inference as combined optimization

$$\max \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{y}_f^f)$$

such that $x_i = y_i^f$ for all $i = 1, \ldots, n, f \in F$, where y_i^f is the same discrete variable as x_i in factor f.

Equivalent Lagrangian formulation

$$\begin{aligned}\max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) = & \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{y}_f^f) \\ & + \sum_{f \in F} \sum_{i \in f} \sum_{\hat{x}_i} \delta(\hat{x}_i, f, i)(1[x_i = \hat{x}_i] - 1[y_i^f = \hat{x}_i])\end{aligned}$$

such that $x_i = y_i^f$ for all $i = 1, \ldots, n, f \in F$.

Dual Decomposition II

Dual problem (not considering the agreement constraints):

$$\begin{aligned} L(\delta) &= \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \\ &= \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f : i \in f} \delta(x_i, f, i) \Big) \\ &\quad + \sum_{f \in F} \max_{\mathbf{y}} \Big(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \Big) \end{aligned}$$

Dual Decomposition II

Dual problem (not considering the agreement constraints):

$$
\begin{aligned}
L(\delta) &= \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \\
&= \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f : i \in f} \delta(x_i, f, i) \Big) \\
&\quad + \sum_{f \in F} \max_{\mathbf{y}} \Big(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \Big)
\end{aligned}
$$

We decompose the original problem into smaller subproblems:

- each subproblems can be solved exactly and efficiently
- the decomposition is then subsequently optimized with respect to δ to encourage the agreement on shared variables
- the decomposition can also be seen as a reparametrization of the original problem, and searching over the set of reparametrizations of the factors θ

Formal Guarantees I - Upper Bound

Upper Bound

For any value of δ,

$$L(\delta) \geq \min_{\delta} L(\delta) \geq \sum_{i \in V} \theta_i(x_i^*) + \sum_{f \in F} \theta_f(\mathbf{y}_f^{f*})$$
$$= \mathrm{MAP}(\theta)$$

where $x_i^* = y_i^{f*}$ are the optimal combined solution.

Proof: $L(\delta) = \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \geq \max_{\mathbf{x},\mathbf{y}: x_i = y_i^f} L(\delta, \mathbf{x}, \mathbf{y}) = \mathrm{MAP}(\theta)$.

Formal Guarantees I - Upper Bound

Upper Bound

For any value of δ,

$$L(\delta) \geq \min_{\delta} L(\delta) \geq \sum_{i \in V} \theta_i(x_i^*) + \sum_{f \in F} \theta_f(\mathbf{y}_f^{f*})$$
$$= \mathrm{MAP}(\theta)$$

where $x_i^* = y_i^{f*}$ are the optimal combined solution.

Proof: $L(\delta) = \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \geq \max_{\mathbf{x},\mathbf{y}: x_i = y_i^f} L(\delta, \mathbf{x}, \mathbf{y}) = \mathrm{MAP}(\theta)$.

Now, the dual problem is to find the tightest upper bound by optimizing the Lagrangian multipliers: solving $\min_\delta L(\delta)$.

Formal Guarantees II - Optimality

Optimality

If there exists δ such that

$$\mathbf{x}_\delta = \mathbf{y}_\delta^f$$

then $\mathbf{x}_\delta$ gives the optimal solution to $\text{MAP}(\theta)$.

where $x_{\delta,i} = \arg\max_{\mathbf{x}} \left(\theta_i(x_i) + \sum_{f:i\in f} \delta(x_i, f, i)\right)$,

$\mathbf{y}_\delta^f = \arg\max_{\mathbf{y}^f} \left(\theta_f(\mathbf{y}_f^f) - \sum_{i\in f} \delta(y_i^f, f, i)\right)$

Proof:

- $L(\delta) = \sum_{i\in V} \theta_i(x_{\delta,i}) + \sum_{f\in F} \theta_f(\mathbf{y}_\delta^f) \geq \text{MAP}(\theta)$
- By the optimality of $\text{MAP}(\theta)$, $\sum_{i\in V} \theta_i(x_{\delta,i}) + \sum_{f\in F} \theta_f(\mathbf{y}_\delta^f) \leq \text{MAP}(\theta)$

Formal Guarantees II - Optimality

Optimality

If there exists δ such that
$$\mathbf{x}_\delta = \mathbf{y}_\delta^f$$
then $\mathbf{x}_\delta$ gives the optimal solution to $\mathrm{MAP}(\theta)$.
where $x_{\delta,i} = \arg\max_{\mathbf{x}} \left(\theta_i(x_i) + \sum_{f:i\in f} \delta(x_i, f, i)\right)$,
$\mathbf{y}_\delta^f = \arg\max_{\mathbf{y}^f} \left(\theta_f(\mathbf{y}_f^f) - \sum_{i\in f} \delta(y_i^f, f, i)\right)$

Proof:

- $L(\delta) = \sum_{i\in V} \theta_i(x_{\delta,i}) + \sum_{f\in F} \theta_f(\mathbf{y}_\delta^f) \geq \mathrm{MAP}(\theta)$
- By the optimality of $\mathrm{MAP}(\theta)$, $\sum_{i\in V} \theta_i(x_{\delta,i}) + \sum_{f\in F} \theta_f(\mathbf{y}_\delta^f) \leq \mathrm{MAP}(\theta)$

This ensures we have the exact solution to the MAP inference. The dual solution δ is said to provide a certificate of optimality in this case.

Dual Optimization

Lagrangian dual:

$$
\begin{aligned}
L(\delta) &= \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \\
&= \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f : i \in f} \delta(x_i, f, i) \Big) \\
&\quad + \sum_{f \in F} \max_{\mathbf{y}} \Big(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \Big)
\end{aligned}
$$

Dual Optimization

Lagrangian dual:

$$
\begin{aligned}
L(\delta) &= \max_{\mathbf{x},\mathbf{y}} L(\delta, \mathbf{x}, \mathbf{y}) \\
&= \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f: i \in f} \delta(x_i, f, i) \Big) \\
&\quad + \sum_{f \in F} \max_{\mathbf{y}} \Big(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \Big)
\end{aligned}
$$

Goal: finding the tightest upper bound

$$
\min_{\delta} L(\delta)
$$

Subgradient

$$L(\delta) = \sum_{i \in V} \max_{x_i} \Big(\theta_i(x_i) + \sum_{f: i \in f} \delta(x_i, f, i) \Big) + \sum_{f \in F} \max_{\mathbf{y}} \Big(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \Big)$$

Properties:

- $L(\delta)$ is convex and continuous in δ (global minima guaranteed)
- $L(\delta)$ is non-differentiable (due to the max operator)

Subgradient

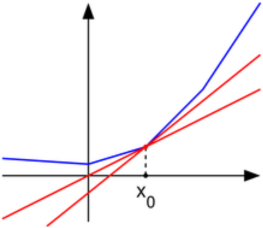
$$L(\delta) = \sum_{i \in V} \max_{x_i} \left(\theta_i(x_i) + \sum_{f : i \in f} \delta(x_i, f, i) \right) + \sum_{f \in F} \max_{\mathbf{y}} \left(\theta_f(\mathbf{y}_f^f) - \sum_{i \in f} \delta(y_i^f, f, i) \right)$$

Properties:

- $L(\delta)$ is convex and continuous in δ (global minima guaranteed)
- $L(\delta)$ is non-differentiable (due to the max operator)

Subgradient: A subgradient of a convex function $L(\delta)$ at δ is a vector g_δ such that for all δ',

$$L(\delta') \geq L(\delta) + g_\delta \cdot (\delta' - \delta)$$



Subgradient Calculation

Subgradient: a slightly loose formulation is

$$1[x_{\delta,i} = \hat{x}_i] - 1[y^f_{\delta,i} = \hat{x}_i]$$

recall the original Lagrangian formulation and the definition of subgradient.

$$\begin{aligned} L(\delta, \mathbf{x}, \mathbf{y}) = & \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{y}^f_f) \\ & + \sum_{f \in F} \sum_{i \in f} \sum_{\hat{x}_i} \delta(\hat{x}_i, f, i)(1[x_i = \hat{x}_i] - 1[y^f_i = \hat{x}_i]) \end{aligned}$$

Subgradient Calculation

Subgradient: a slightly loose formulation is

$$1[x_{\delta,i} = \hat{x}_i] - 1[y^f_{\delta,i} = \hat{x}_i]$$

recall the original Lagrangian formulation and the definition of subgradient.

$$\begin{aligned} L(\delta, \mathbf{x}, \mathbf{y}) = & \sum_{i \in V} \theta_i(x_i) + \sum_{f \in F} \theta_f(\mathbf{y}^f_f) \\ & + \sum_{f \in F} \sum_{i \in f} \sum_{\hat{x}_i} \delta(\hat{x}_i, f, i)(1[x_i = \hat{x}_i] - 1[y^f_i = \hat{x}_i]) \end{aligned}$$

Specifically,

- $g(x_{\delta,i}, f, i) = 1$, if $x_{\delta,i} \neq y^f_{\delta,i}$
- $g(y^f_{\delta,i}, f, i) = -1$, if $x_{\delta,i} \neq y^f_{\delta,i}$
- $g(x_i, f, i) = 0$, all others

Subgradient Algorithms

Subgradient descent: at iteration $t + 1$

$$\delta^{t+1}(x_i, f, i) = \delta^t(x_i, f, i) - \alpha_t g^t(x_i, f, i)$$

where α_t is a step-size that may depend on t.

Subgradient Algorithms

Subgradient descent: at iteration $t + 1$

$$\delta^{t+1}(x_i, f, i) = \delta^t(x_i, f, i) - \alpha_t g^t(x_i, f, i)$$

where α_t is a step-size that may depend on t.

Convergence: for any sequence $\alpha_0, \alpha_1, \alpha_2, \ldots$ such that

$$\lim_{t \to \infty} \alpha_t = 0, \qquad \sum_{t=0}^{\infty} \alpha_t = \infty,$$

we have

$$\lim_{t \to \infty} L(\delta^t) = \min_{\delta} L(\delta).$$