Convergence Rates for Stochastic Dual Descent Algorithm

Ketan Rajawat

SPiN Lab,
Department of Electrical Engineering
Indian Institute of Technology Kanpur
Uttar Pradesh, India

June 21, 2016

This work was supported by the Indo-French Centre for the Promotion of Advanced Research-CEFIPRA.

Stochastic Resource Allocation Problem

$$(\mathbf{P}_1) \qquad (\mathbf{x}^\star, \{\mathbf{p}_t^\star\}_{t\in\mathbb{N}}) = \arg\max f_0(\mathbf{x}) \tag{1}$$

$$\text{s. t. } \mathbb{E}\left[\mathbf{s}_t(\mathbf{p}_t, \mathbf{x})\right] \geq 0 \tag{2}$$

$$\mathbf{x} \in \mathcal{X}, \mathbf{p}_t \in \mathcal{P}_t \tag{3}$$

- **Goal** is to determine the resource allocation variable $\mathbf{x} \in \mathbb{R}^n$ and policy $\mathbf{p}_t \in \mathbb{R}^p$ for all $t \in \mathbb{N}$

Dual Descent Based Solution

- **Dual descent using sample averages**

$$\{\mathbf{x}_t, \mathbf{p}_t\} \in \arg \max_{\mathbf{x}\in\mathcal{X}, \mathbf{p}(.)\in\mathcal{P}} f(\mathbf{x}) - \boldsymbol{\lambda}^T \mathbb{E}\left[\mathbf{s}_t(\mathbf{p}, \mathbf{x})\right] \tag{4a}$$

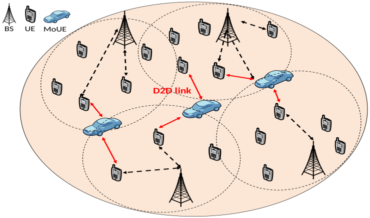
$$\boldsymbol{\lambda}_{t+1} = \left[\boldsymbol{\lambda}_t - \epsilon \mathbb{E}\left[\mathbf{s}_t(\mathbf{p}_t, \mathbf{x}_t)\right]\right]_+ \tag{4b}$$

- **Stochastic Dual Descent Algorithm**

$$(\mathbf{x}_t, \mathbf{p}_t) = \arg \max_{\mathbf{x}\in\mathcal{X}, \mathbf{p}\in\mathcal{P}_t} f_0(\mathbf{x}) + \boldsymbol{\lambda}_t{}^T \mathbf{s}_t(\mathbf{p}, \mathbf{x}) \tag{5a}$$

$$\boldsymbol{\lambda}_{t+1} = \left[\boldsymbol{\lambda}_t - \epsilon \mathbf{s}_t(\mathbf{x}_t, \mathbf{p}_t)\right]_+ \tag{5b}$$

System model for Edge Caching



Proposed Algorithm (Repeat for $t = 1, 2, \ldots,$)

Step 1. Collect download costs c_t^i and γ_t^i from mobile caches $i = 1, 2, \ldots, M_t$

Step 2. **Primal update:** Find "winning" cache i_t and allocated power p_t^i

$$r_t \in \underset{0 \le r \le r_{max}}{\arg\max}\ U(r) - \lambda_t r, \qquad \{p_t^i\}_{i=1}^{M_t} \in \underset{\{p^i\}_{i=1}^{M_t} \in \mathcal{P}_t}{\arg\max} \sum_{i=1}^{M_t} \left[\lambda_t R_i(p^i, \gamma_t^i) - c_t^i p^i\right].$$

Step 3. Download at rate $R_{i_t}(p_t^{i_t}, \gamma_t^i)$ from user i_t

Step 4. **Dual update:** $\lambda_{t+1} = \mathcal{P}_\Lambda\left(\lambda_t - \epsilon\left[R_{i_t}(p_t^{i_t}, \gamma_t^i) - r_t\right]\right)$