Doubly Separable Models and Distributed Parameter Estimation

Hyokun Yun

Department of Statistics
Purdue University

April 23rd, 2014

Outline

Challenges for statisticians

Most statistical procedures require us to solve:

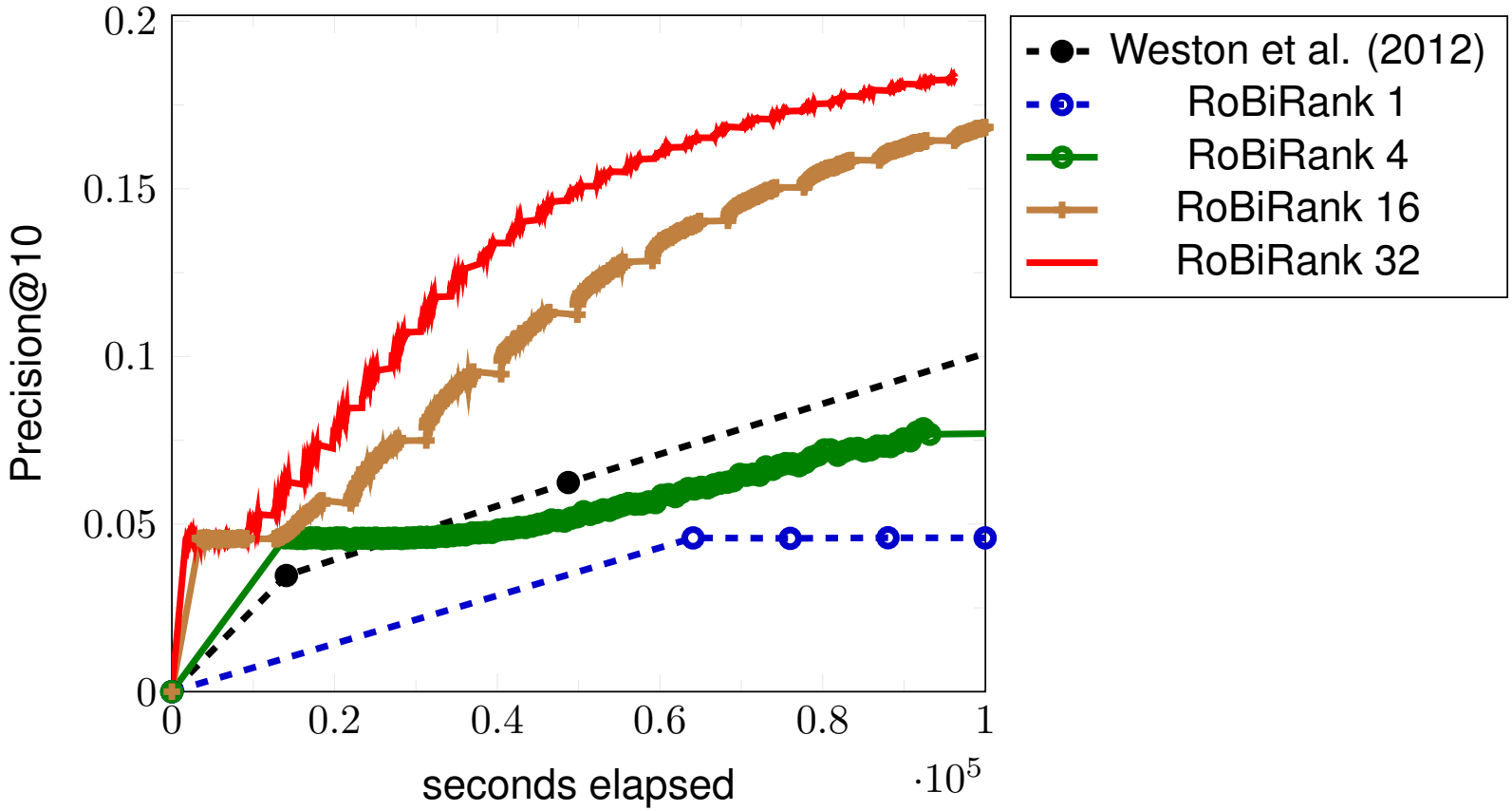
$$\min_{\theta} f(\theta) = \sum_{i=1}^{n} f_i(\theta).$$

Challenges:

- n is large
- $\dim(\theta)$ is large
- time is limited

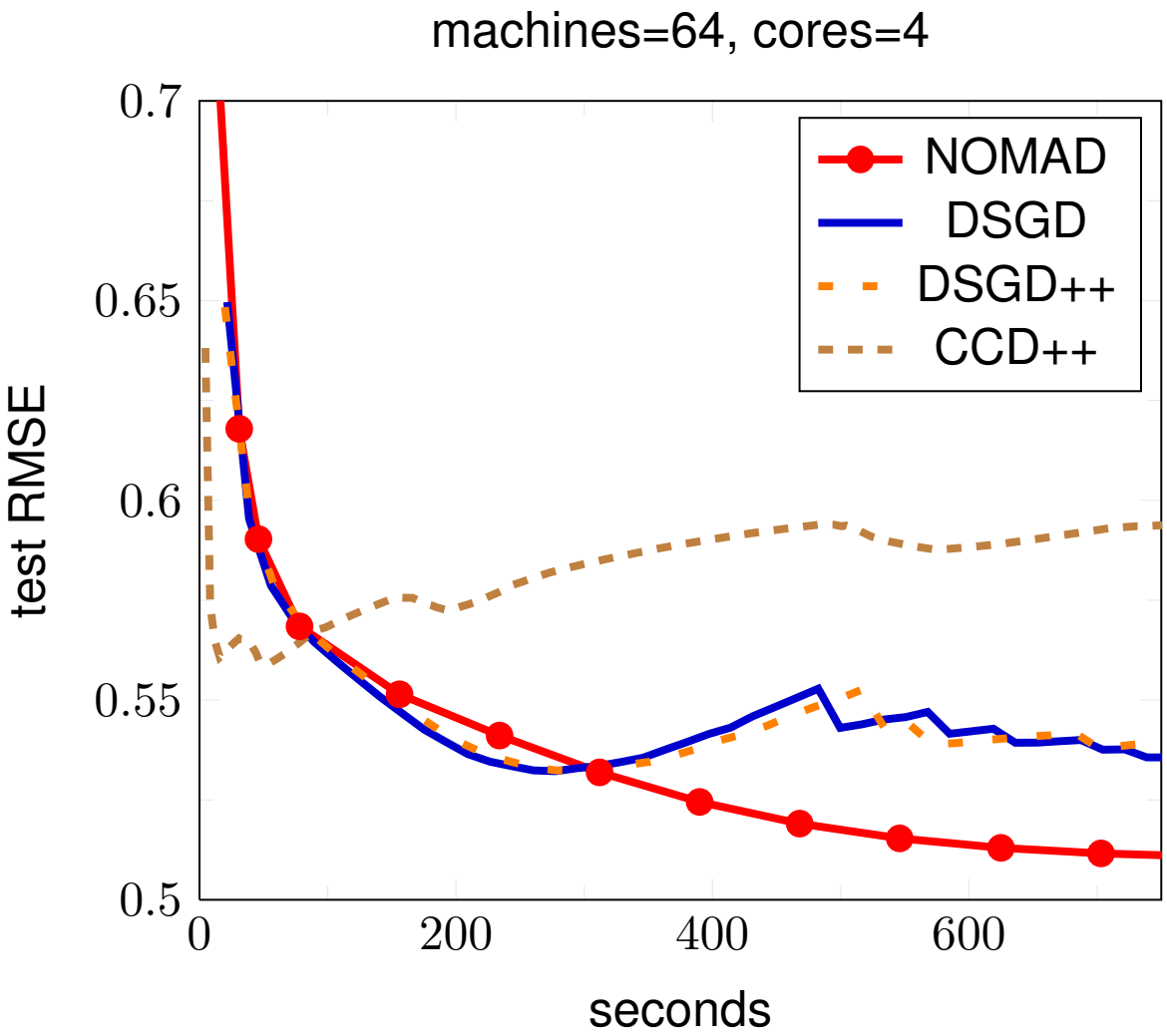
Scale of the Challenge

Learn a recommendation model from
1.1 million users, 386K songs, and 49 million records



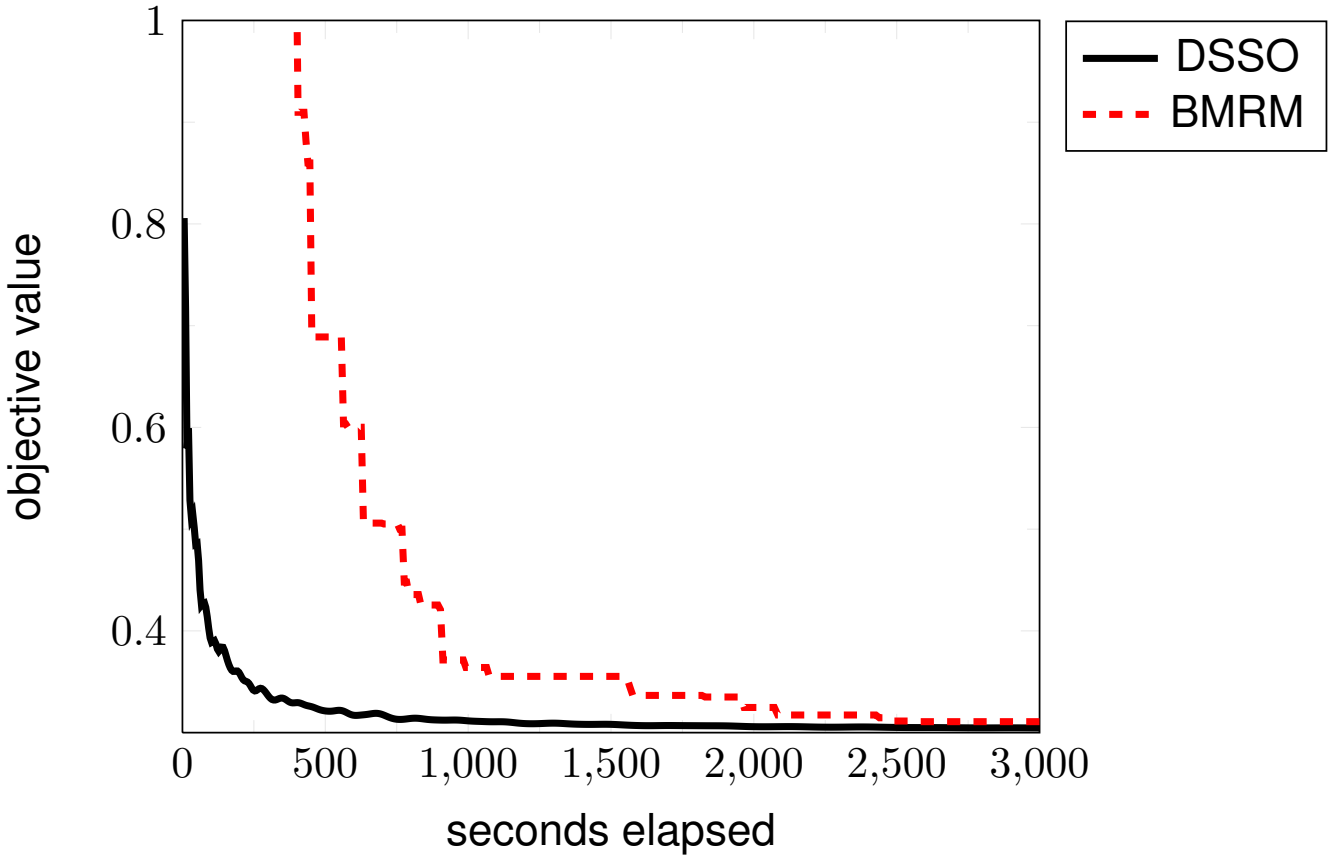
Scale of the Challenge

Complete a 50 million $\times$ 40K matrix with 2.7 billion entries



Scale of the Challenge

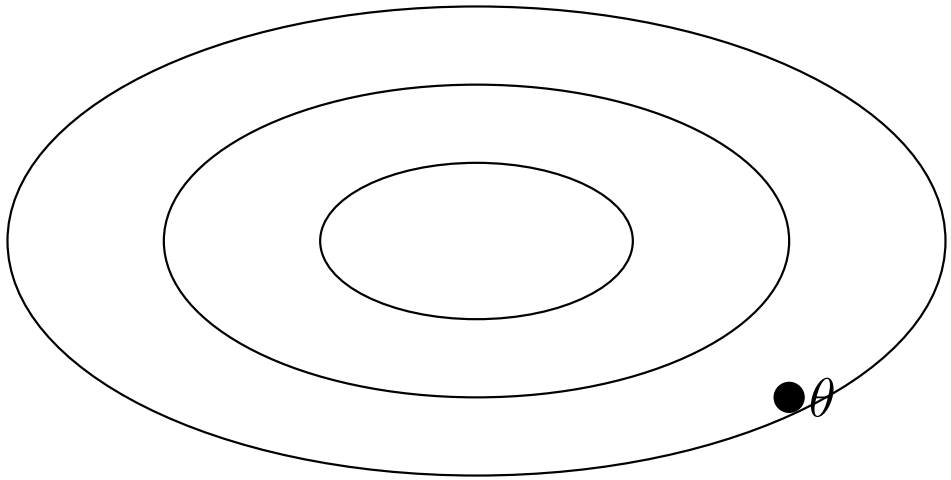
Learn a classification model from
8.4 million data points and 20 million features



Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta))$

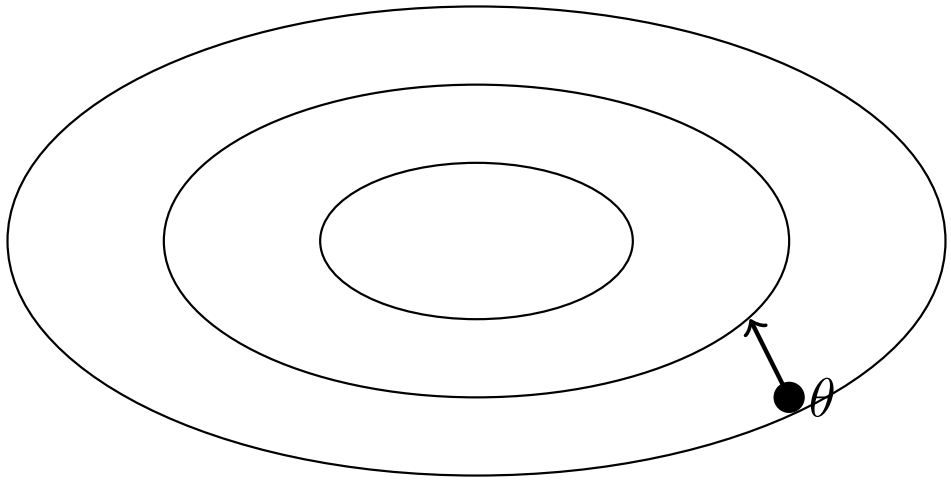


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta)$)

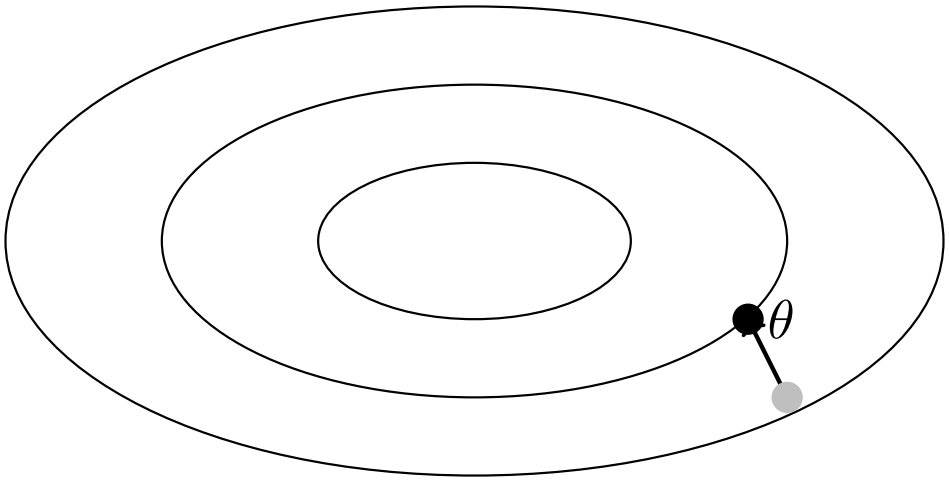


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta))$

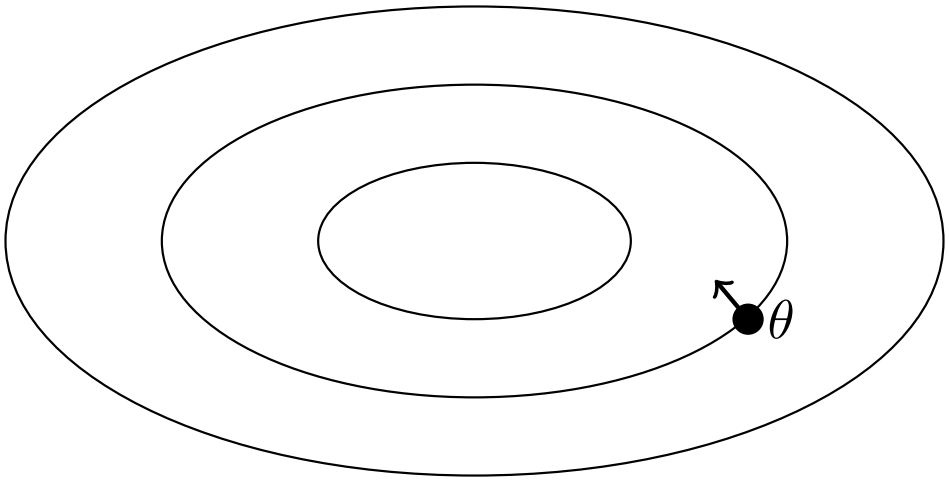


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta)$)

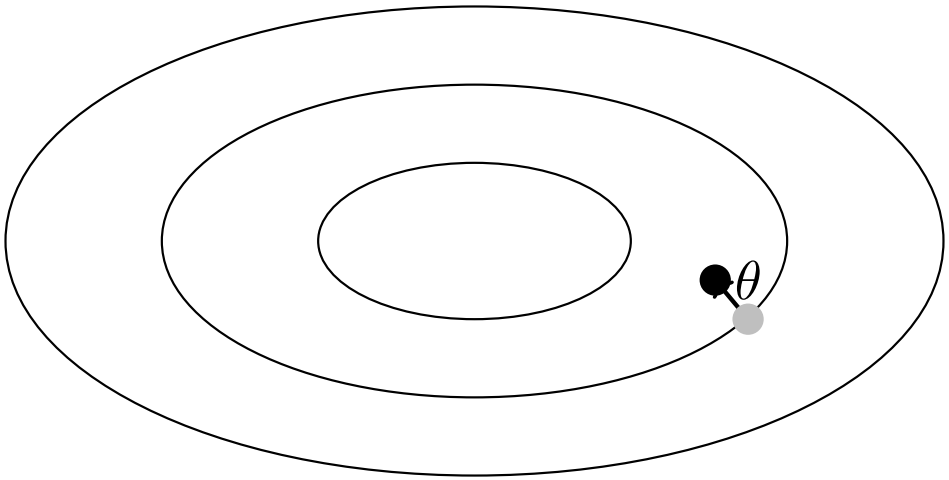


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta))$

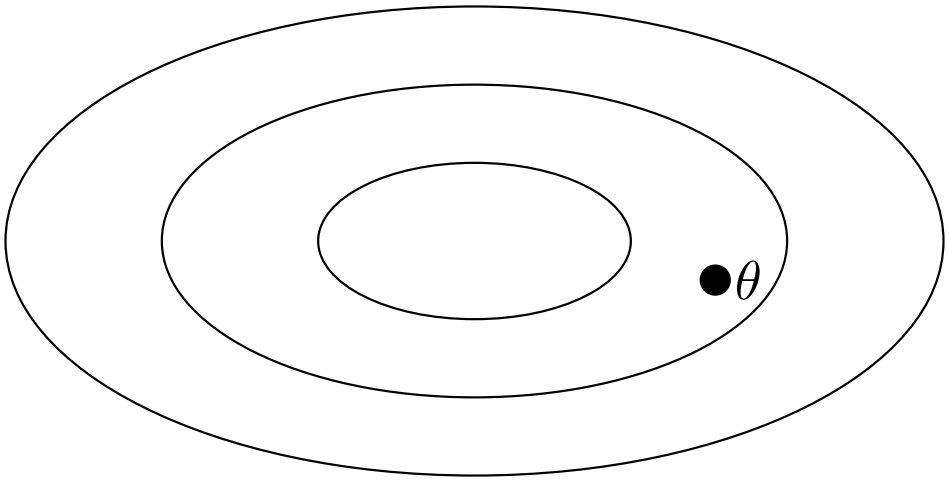


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta))$

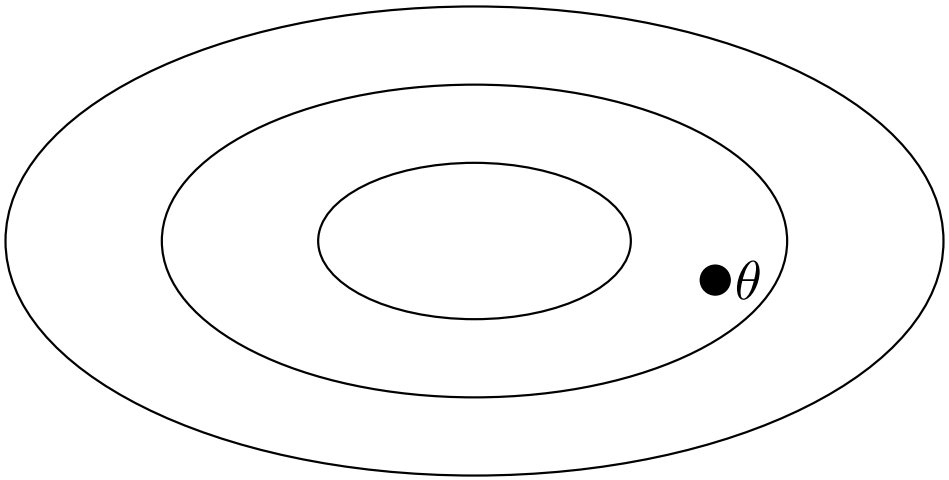


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Limitations of Batch Optimization

Gradient Descent Algorithm:

- Start with some θ
- Calculate the negative gradient $-\nabla_\theta f(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta))$

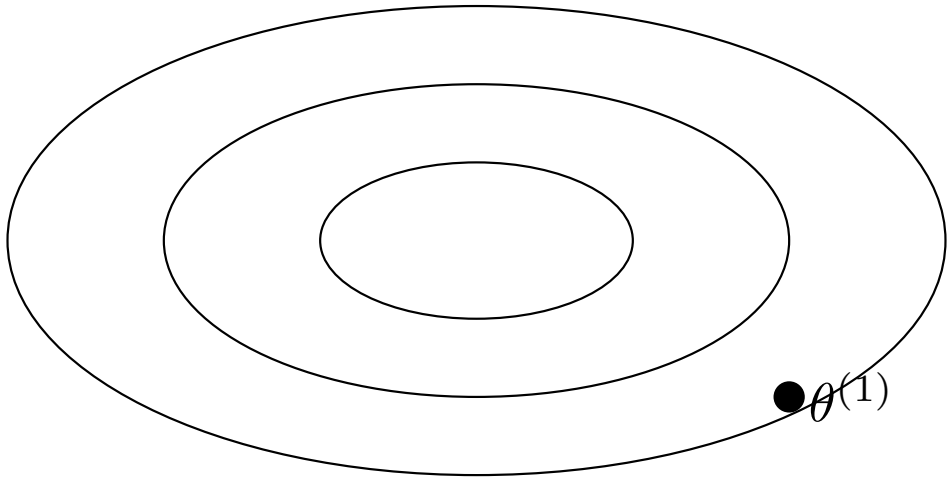


Challenge: $\nabla_\theta f(\theta) = \sum_{i=1}^{n} \nabla_\theta f_i(\theta)$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$

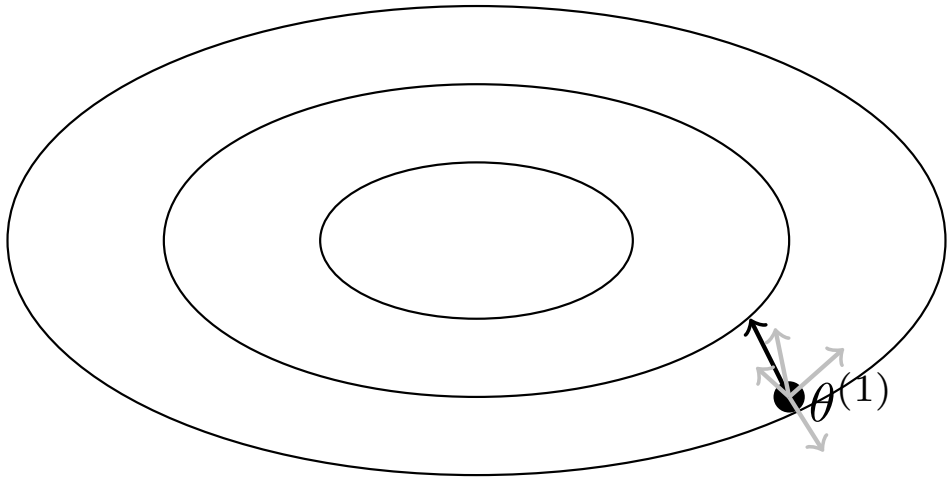


Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$

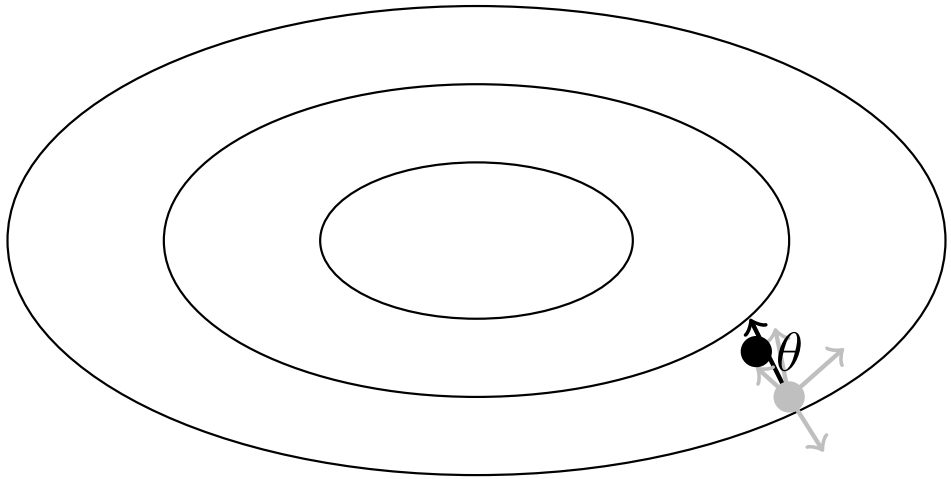


Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$

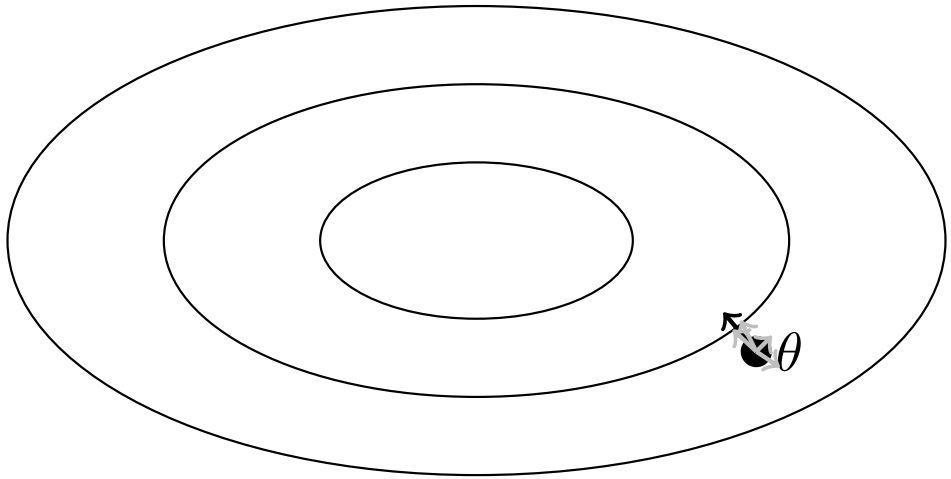


Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$

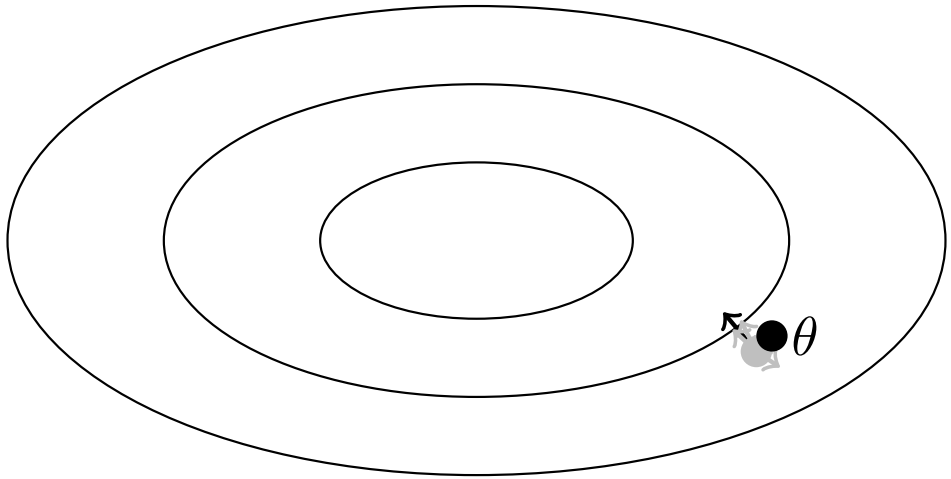


Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$

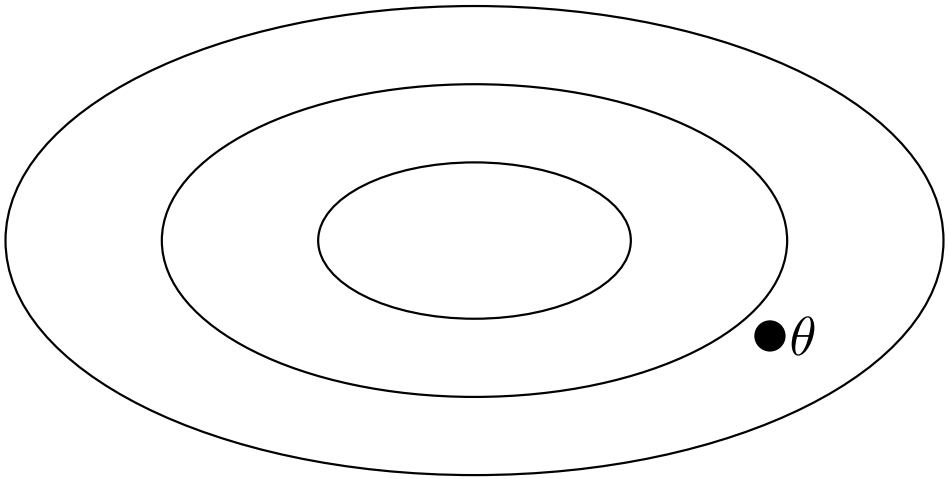


Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Attractiveness of Stochastic Gradient Descent

Stochastic Gradient Descent Algorithm:

- Start with some θ
- Choose i from $1, 2, \ldots, n$
- Calculate the negative gradient $-n \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot n \cdot \nabla_\theta f_i(\theta))$



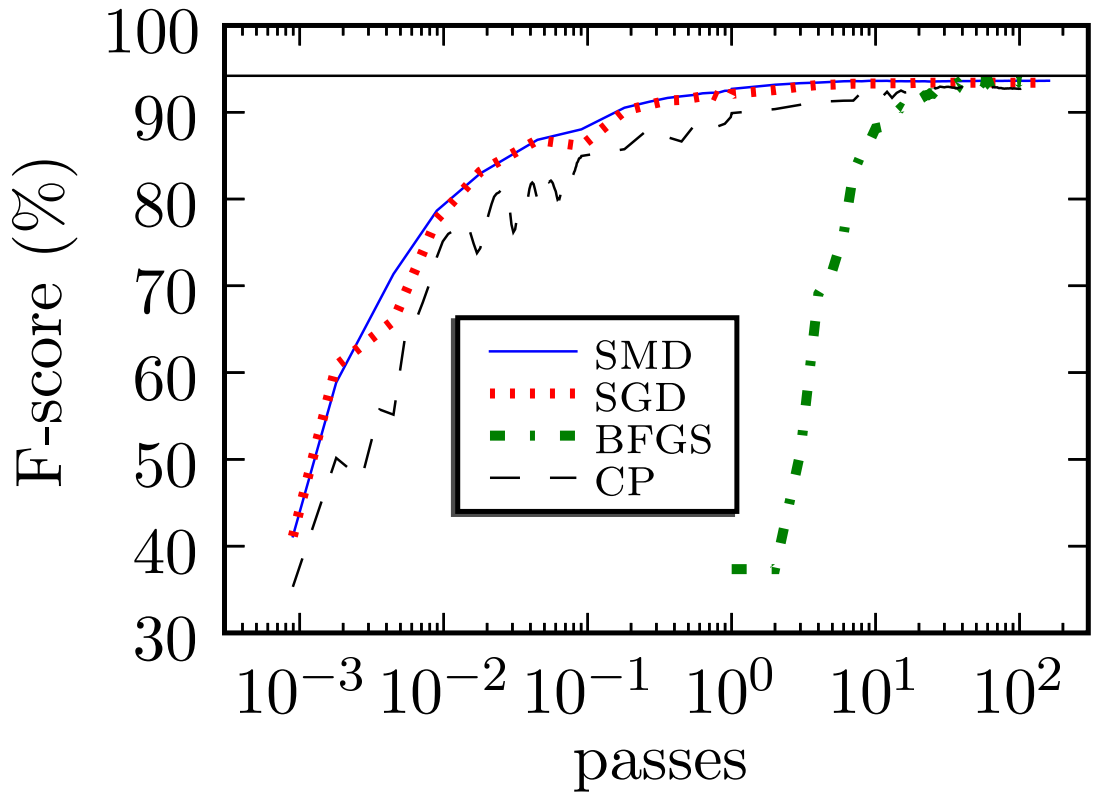
Rational: $\nabla_\theta f(\theta) = \mathbb{E}\left[n \cdot \nabla_\theta f_i(\theta)\right]$

Comparison of GD and SGD

Asymptotic analysis of Bottou and Bousquet (2011):

	GD	2GD	SGD
Time per iteration:	n	n	1
Time to error ε:	$\frac{1}{\varepsilon^{1/\alpha}} \log^2 \frac{1}{\varepsilon}$	$\frac{1}{\varepsilon^{1/\alpha}} \log \frac{1}{\varepsilon} \log\log \frac{1}{\varepsilon}$	$\frac{1}{\varepsilon}$

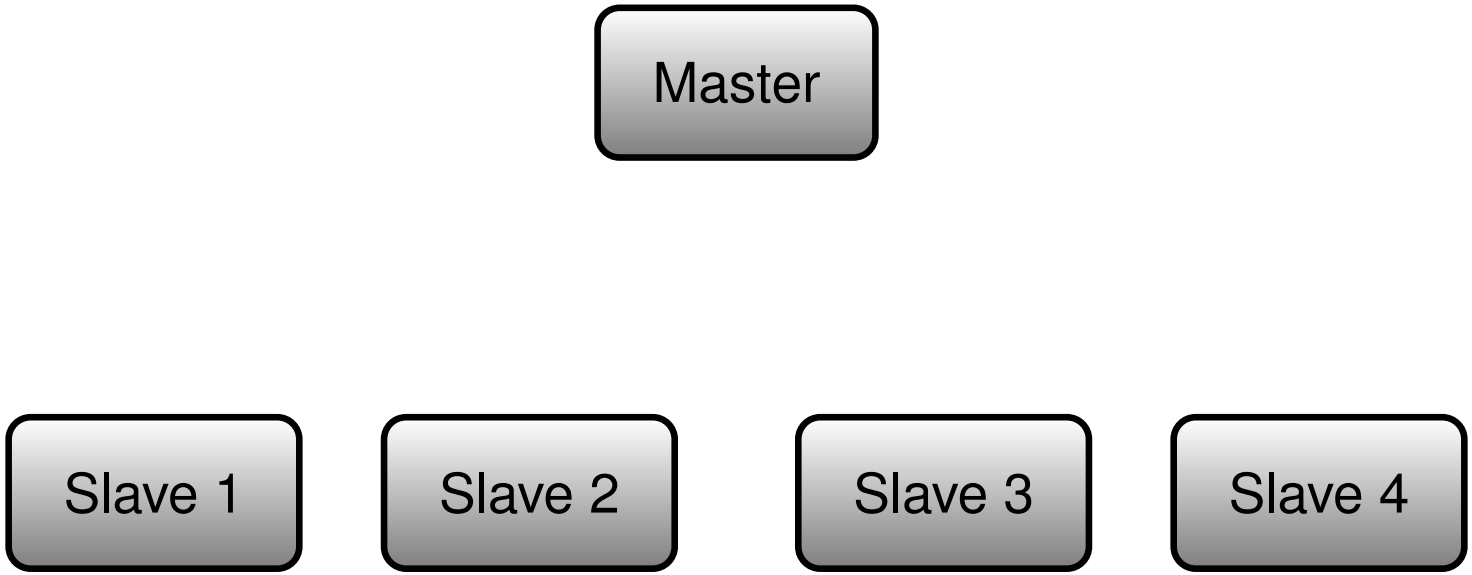
Empirical results:



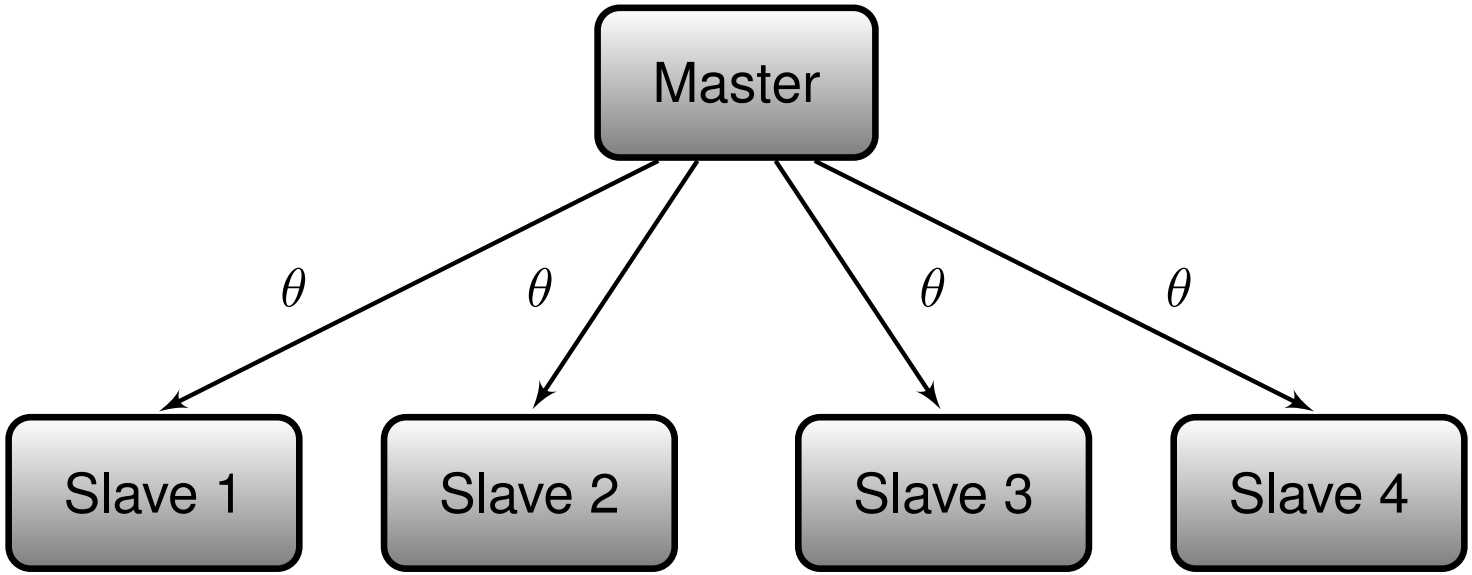
Another Perspective: Parallelism

(Image from https://www.jlab.org/news/releases/jefferson-lab-boasts-virginias-fastest-computer)

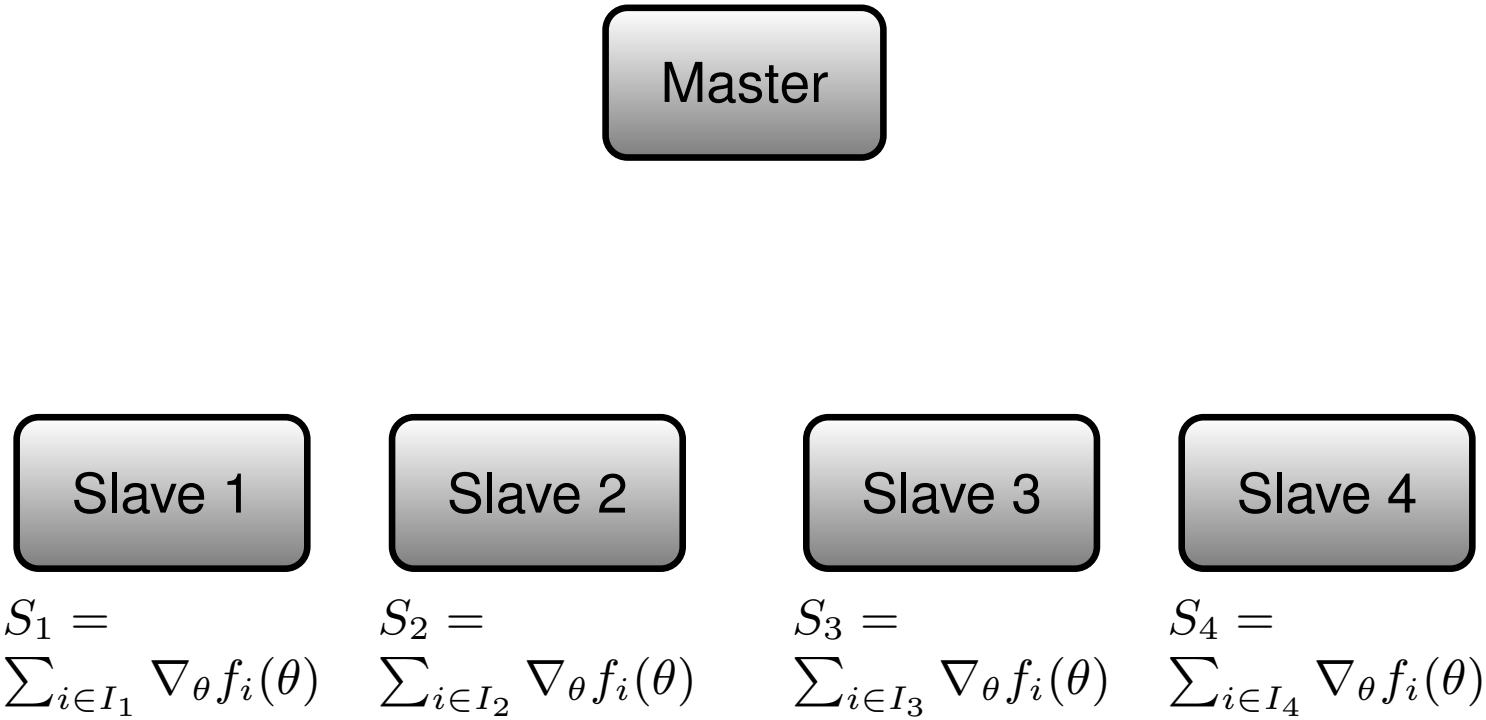
Parallelization of Gradient Descent



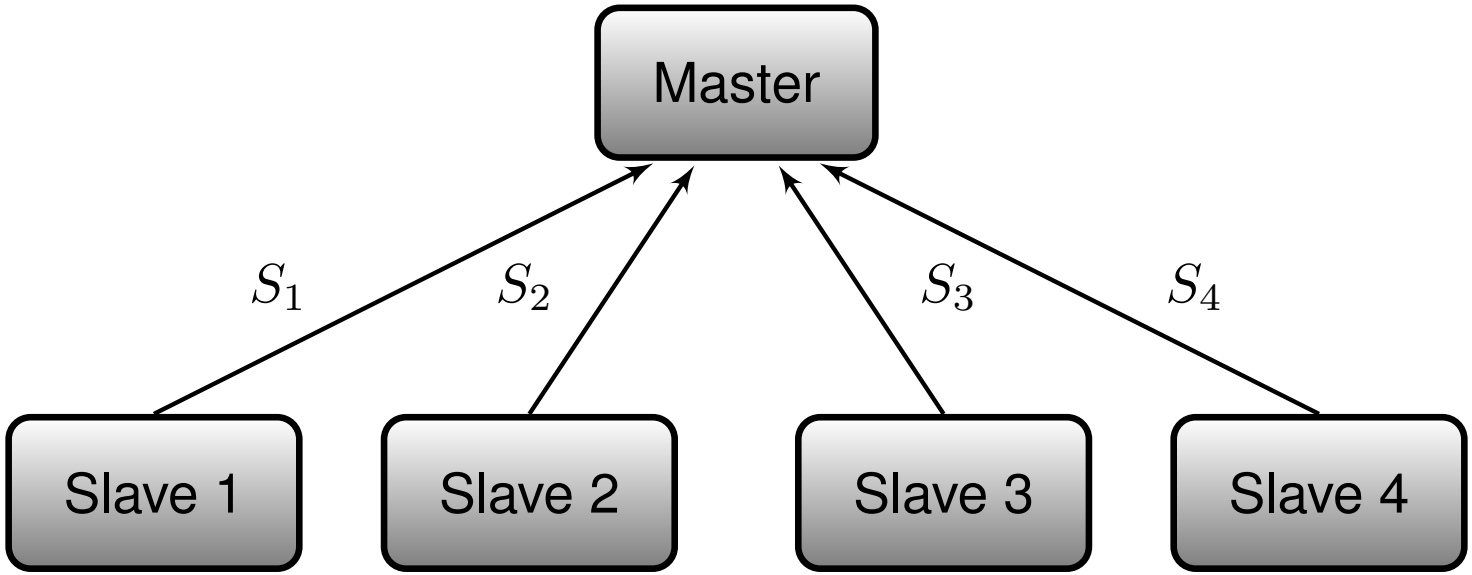
Parallelization of Gradient Descent



Parallelization of Gradient Descent

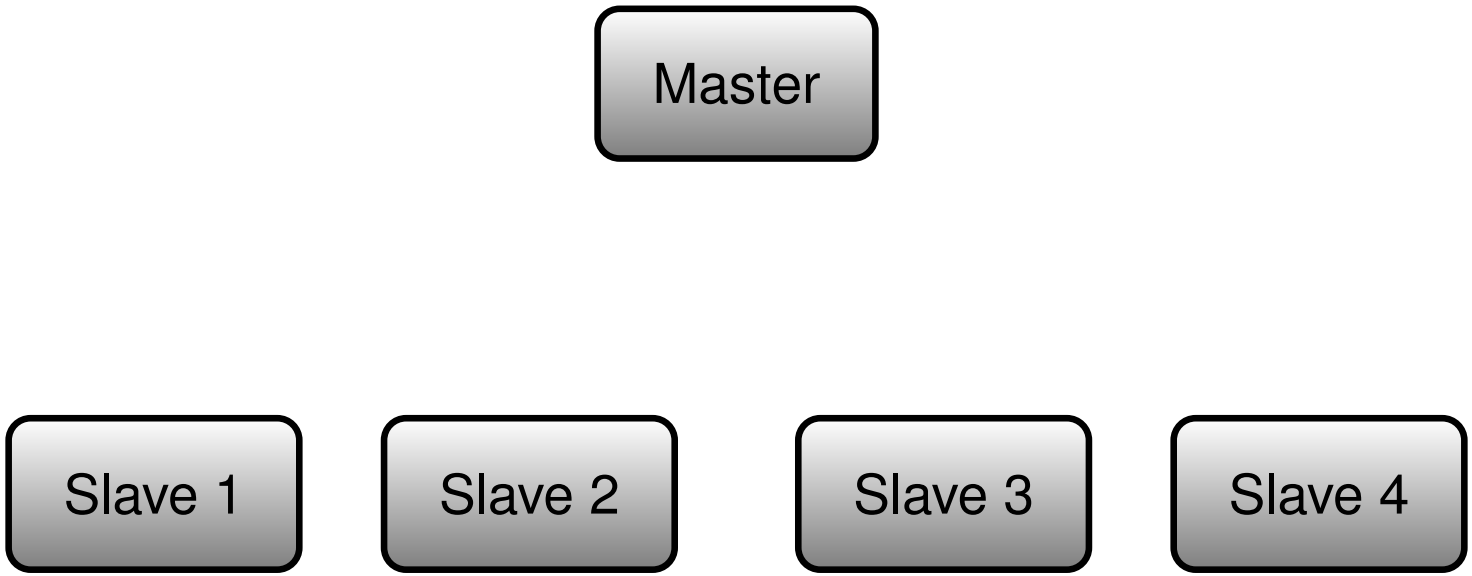


Parallelization of Gradient Descent



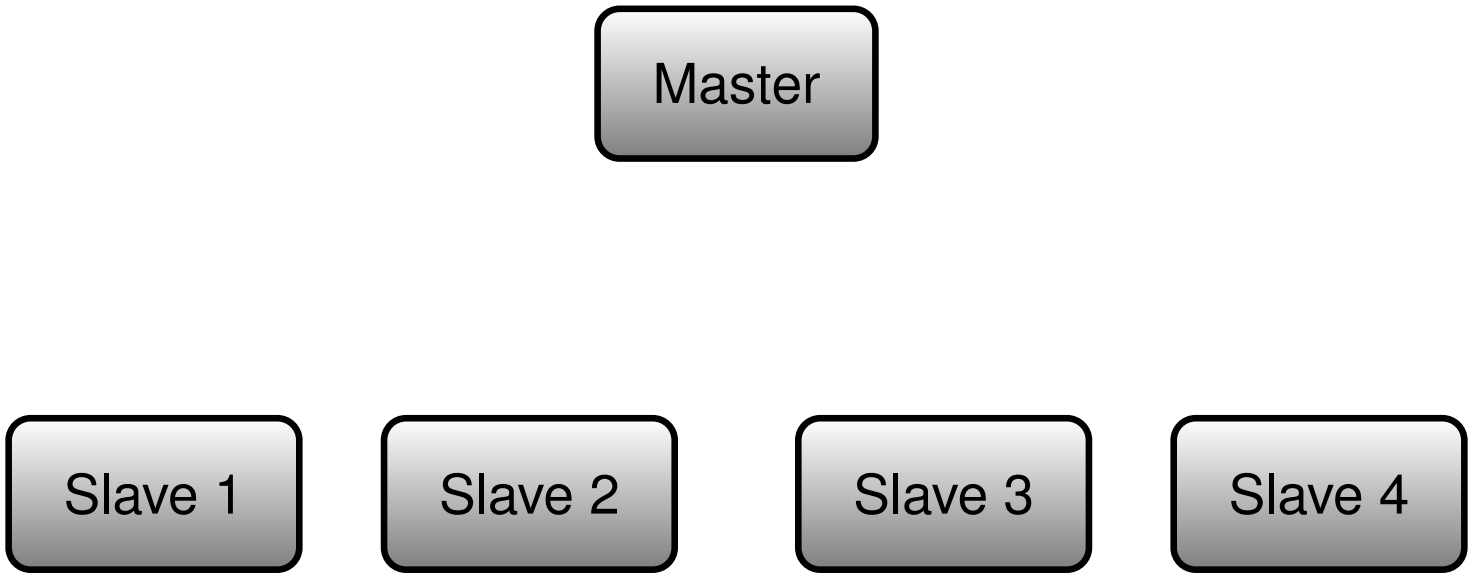
Parallelization of Gradient Descent

$$\nabla_\theta f(\theta) = S_1 + S_2 + S_3 + S_4$$



Parallelization of Gradient Descent

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta f(\theta)$$



Difficulty for Stochastic Gradient Descent

Recall:

- Start with some θ
- Choose i from $1, 2, \ldots, m$
- Calculate the negative gradient $-m \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot m \cdot \nabla_\theta f_i(\theta)$

Difficulty for Stochastic Gradient Descent

Recall:

- Start with some θ
- Choose i from $1, 2, \ldots, m$
- **Calculate the negative gradient** $-m \cdot \nabla_\theta f_i(\theta)$
- Move towards the direction: $\theta \leftarrow \theta - \eta \cdot m \cdot \nabla_\theta f_i(\theta)$

Our Approach

- For the matrix completion problem, efficient parallel SGD is known (Gemulla et. al, 2011)
- Question 1: Can it be genearlized for larger class of problems?
- Question 2: Is the class interesting?

Outline

Outline

Separability

The nicest situation occurs when the objective function is *separable*:

$$f(\theta_1, \theta_2, \ldots) = f_1(\theta_1) + f_2(\theta_2) + \cdots$$

However, this is too much to hope for.

Double Separability

A function is *doubly separable* if it can be written:

$$f(\underbrace{w_1, w_2, \ldots, w_m, h_1, h_2, \ldots, h_n}_{\theta}) = \sum_{(i,j)\in\Omega} f_{ij}(w_i, h_j).$$

It is easy to see that this is a weaker condition.

- Separable $\Rightarrow$ Doubly Separable
- Doubly Separable $\not\Rightarrow$ Separable

Double Separability

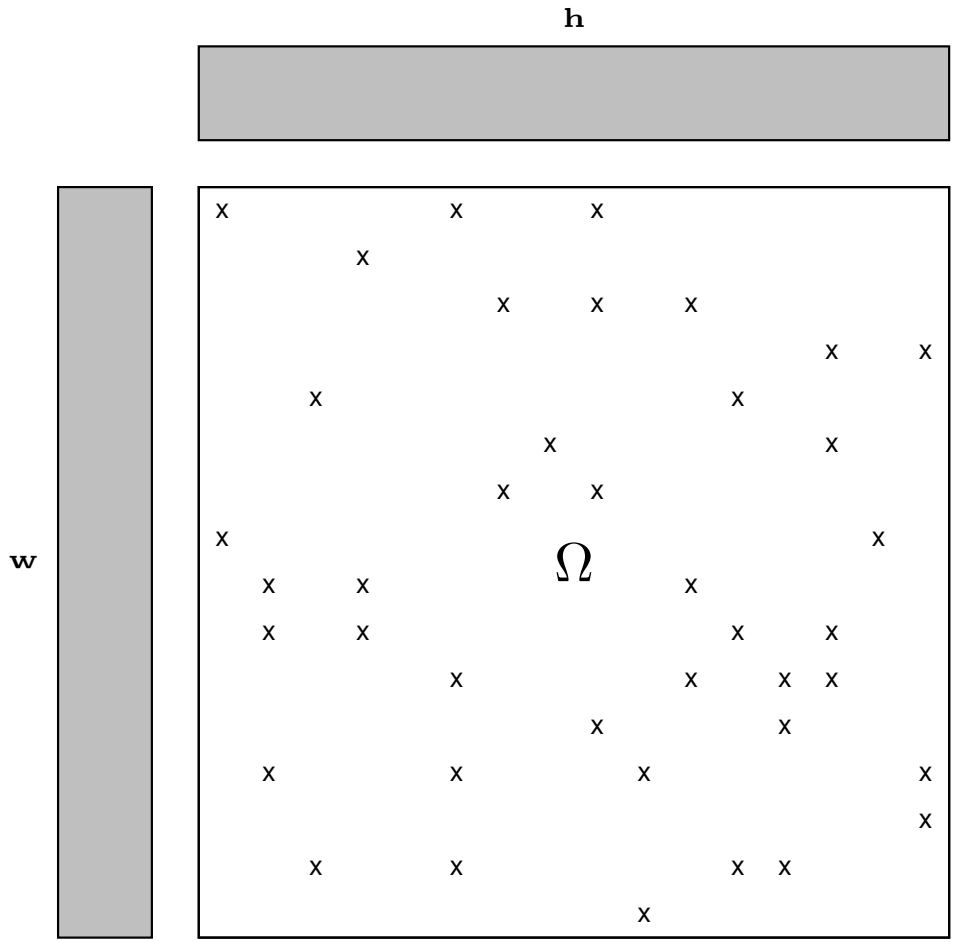
A function is *doubly separable* if it can be written:

$$f(\underbrace{w_1, w_2, \ldots, w_m}_{\mathbf{w}}, \underbrace{h_1, h_2, \ldots, h_n}_{\mathbf{h}}) = \sum_{(i,j)\in\Omega} f_{ij}(w_i, h_j).$$

It is easy to see that this is a weaker condition.

- Separable $\Rightarrow$ Doubly Separable
- Doubly Separable $\not\Rightarrow$ Separable

Double Separability



Minimization Problem

Solve:

$$\min_{\theta} f(\theta) = \sum_{(i,j)\in\Omega} f_{ij}(w_i, h_j).$$

Gradient Descent executes:

$$\theta \leftarrow \theta - \eta \cdot \nabla_{\theta} \sum_{(i,j)\in\Omega} \nabla_{\theta} f_{ij}(\theta)$$

Minimization Problem

Solve:

$$\min_{\theta} f(\theta) = \sum_{(i,j)\in\Omega} f_{ij}(w_i, h_j).$$

Gradient Descent executes:

$$\theta \leftarrow \theta - \eta \cdot \nabla_{\theta} \sum_{(i,j)\in\Omega} \nabla_{\theta} f_{ij}(\theta)$$

Stochastic Gradient Descent (SGD) samples $(i, j) \in \Omega$ and executes:

$$\theta \leftarrow \theta - \eta \cdot |\Omega| \cdot \nabla_{\theta} f_{ij}(w_i, h_j)$$

Minimization Problem

Solve:

$$\min_{\theta} f(\theta) = \sum_{(i,j)\in\Omega} f_{ij}(w_i, h_j).$$

Gradient Descent executes:

$$\theta \leftarrow \theta - \eta \cdot \nabla_{\theta} \sum_{(i,j)\in\Omega} \nabla_{\theta} f_{ij}(\theta)$$

Stochastic Gradient Descent (SGD) samples $(i, j) \in \Omega$ and executes:

$$\theta \leftarrow \theta - \eta \cdot |\Omega| \cdot \nabla_{\theta} f_{ij}(w_i, h_j)$$

This can be simplified as:

$$w_i \leftarrow w_i - \eta \cdot |\Omega| \cdot \nabla_{w_i} f_{ij}(w_i, h_j)$$
$$h_j \leftarrow h_j - \eta \cdot |\Omega| \cdot \nabla_{h_j} f_{ij}(w_i, h_j)$$

Saddle-point Problem

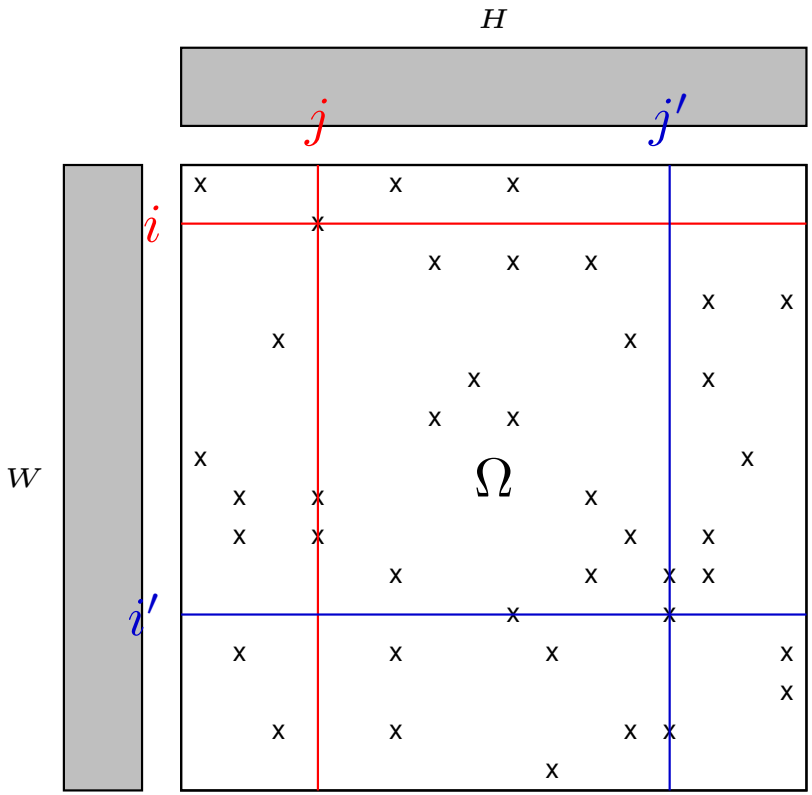
Solve:

$$\min_{\mathbf{w}} \max_{\mathbf{h}} f(\mathbf{w}, \mathbf{h}) = \sum_{(i,j) \in \Omega} f_{ij}(w_i, h_j).$$

Stochastic Saddle-point Optimization (SSO) samples $(i, j) \in \Omega$ and executes:

$$w_i \leftarrow w_i - \eta \cdot |\Omega| \cdot \nabla_{w_i} f_{ij}(w_i, h_j)$$
$$h_j \leftarrow h_j + \eta \cdot |\Omega| \cdot \nabla_{h_j} f_{ij}(w_i, h_j)$$

Low Dependency of Updates



If $i \neq i'$ and $j \neq j'$,

$$\begin{cases} w_i & \leftarrow w_i - \eta \cdot |\Omega| \cdot \nabla_{w_i} f_{ij}(w_i, h_j) \\ h_j & \leftarrow h_j - \eta \cdot |\Omega| \cdot \nabla_{h_j} f_{ij}(w_i, h_j) \end{cases}$$

and

$$\begin{cases} w_{i'} & \leftarrow w_{i'} - \eta \cdot |\Omega| \cdot \nabla_{w_i} f_{i'j'}(w_{i'}, h_{j'}) \\ h_{j'} & \leftarrow h_{j'} - \eta \cdot |\Omega| \cdot \nabla_{h_{j'}} f_{i'j'}(w_{i'}, h_{j'}) \end{cases}$$

can be run simultaneously.

Outline

Original Formulation

In statistics and machine learning, we often solve: given data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$,

$$\min_{\mathbf{w}} P(\mathbf{w}) = \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(\langle \mathbf{w}, \mathbf{x}_j \rangle),$$

where each ℓ_j is convex.
Important examples:

- Generalized Linear Model
  - Linear Regression
  - Logistic Regression
  - Poisson Regression
- Support Vector Machines

Problem reformulation

$$\min_{\mathbf{w}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(\langle \mathbf{w}, \mathbf{x}_j \rangle)$$

Problem reformulation

$$\min_{\mathbf{w}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(\langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \min_{\mathbf{w},\mathbf{u}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j)$$

$$\text{subject to} \quad u_j = \langle \mathbf{w}, \mathbf{x}_j \rangle \quad \{j = 1 \dots n\}$$

Problem reformulation

$$\min_{\mathbf{w}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(\langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \min_{\mathbf{w},\mathbf{u}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j)$$

$$\text{subject to} \quad u_j = \langle \mathbf{w}, \mathbf{x}_j \rangle \quad \{j = 1 \dots n\}$$

$$\Leftrightarrow \min_{\mathbf{w},\mathbf{u}} \max_{\mathbf{h}} \quad \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

Problem reformulation

$$\min_{\mathbf{w}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(\langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \min_{\mathbf{w},\mathbf{u}} \quad \lambda \sum_{i=1}^{m} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j)$$

$$\text{subject to} \quad u_j = \langle \mathbf{w}, \mathbf{x}_j \rangle \quad \{j = 1 \dots n\}$$

$$\Leftrightarrow \min_{\mathbf{w},\mathbf{u}} \max_{\mathbf{h}} \quad \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w},\mathbf{u}} \quad \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

Problem reformulation

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w},\mathbf{u}} \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

Problem reformulation

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w},\mathbf{u}} \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \lambda \sum_{i=1}^{d} \phi_i(w_i) - \frac{1}{n} \sum_{j=1}^{n} h_j \langle \mathbf{w}, \mathbf{x}_j \rangle + \frac{1}{n} \sum_{j=1}^{n} \min_{u_j} (\ell_j(u_j) + h_j u_j)$$

Problem reformulation

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w},\mathbf{u}} \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \lambda \sum_{i=1}^{d} \phi_i(w_i) - \frac{1}{n} \sum_{j=1}^{n} h_j \langle \mathbf{w}, \mathbf{x}_j \rangle + \frac{1}{n} \sum_{j=1}^{n} \min_{u_j} (\ell_j(u_j) + h_j u_j)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \lambda \sum_{i=1}^{m} \phi_j(w_i) - \left\langle \mathbf{w}, \frac{1}{n} \sum_{j=1}^{n} h_j \mathbf{x}_j \right\rangle + \frac{1}{n} \sum_{j=1}^{n} \ell_j^*(-h_j)$$

Problem reformulation

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w},\mathbf{u}} \; \lambda \sum_{i=1}^{d} \phi_i(w_i) + \frac{1}{n} \sum_{j=1}^{n} \ell_j(u_j) + \frac{1}{n} \sum_{j=1}^{n} h_j(u_j - \langle \mathbf{w}, \mathbf{x}_j \rangle)$$

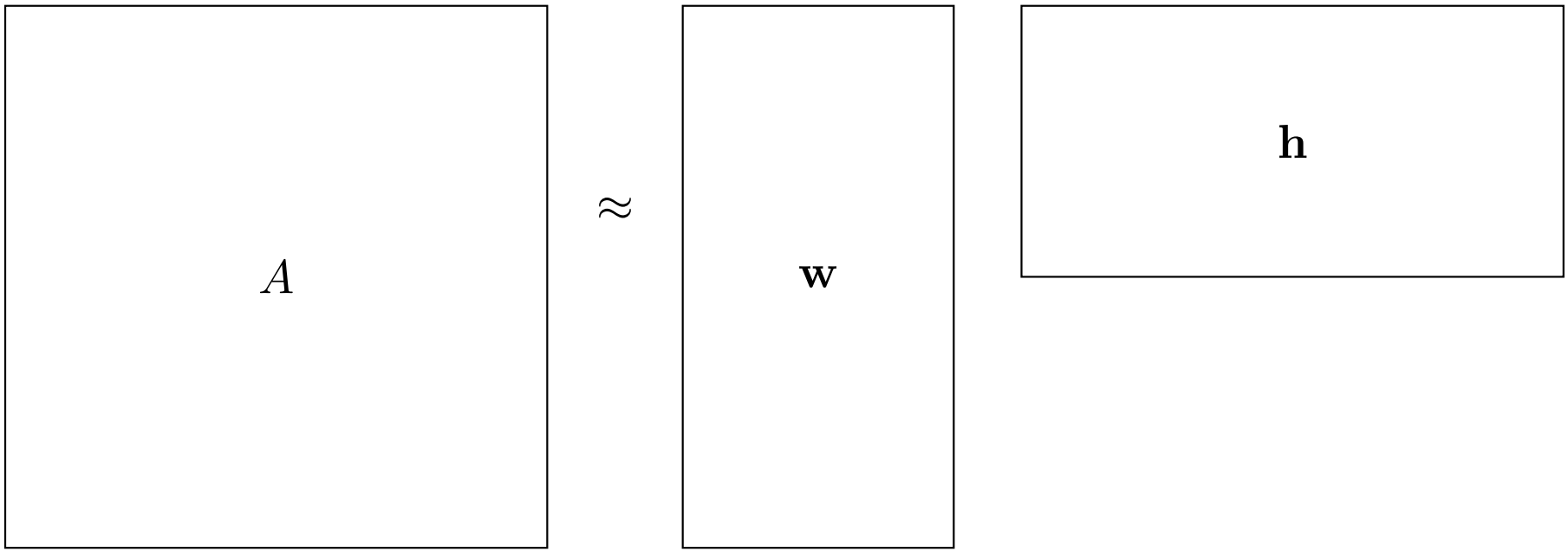
$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \; \lambda \sum_{i=1}^{d} \phi_i(w_i) - \frac{1}{n} \sum_{j=1}^{n} h_j \langle \mathbf{w}, \mathbf{x}_j \rangle + \frac{1}{n} \sum_{j=1}^{n} \min_{u_j} (\ell_j(u_j) + h_j u_j)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \; \lambda \sum_{i=1}^{m} \phi_j(w_i) - \left\langle \mathbf{w}, \frac{1}{n} \sum_{j=1}^{n} h_j \mathbf{x}_j \right\rangle + \frac{1}{n} \sum_{j=1}^{n} \ell_j^*(-h_j)$$

$$\Leftrightarrow \max_{\mathbf{h}} \min_{\mathbf{w}} \; f(\mathbf{w}, \mathbf{h}) := \sum_{(i,j) \in \Omega} \underbrace{\left(\frac{\lambda}{\left|\bar{\Omega}_i\right|} \phi_i(w_i) - \frac{1}{n} h_j w_i x_{ij} + \frac{1}{n \left|\Omega_j\right|} \ell_j^*(-h_j) \right)}_{f_{ij}(w_i, h_j)}$$

Outline

Matrix completion



Matrix completion

$$\min_{\substack{\mathbf{w} \in \mathbb{R}^{m \times k} \\ \mathbf{h} \in \mathbb{R}^{n \times k}}} f(\mathbf{w}, \mathbf{h}),$$

$$f(\mathbf{w}, \mathbf{h}) = \frac{1}{2} \sum_{(i,j) \in \Omega} \left\{ \left(A_{ij} - w_i^\top h_j \right)^2 + \lambda \left(\|w_i\|^2 + \|h_j\|^2 \right) \right\}.$$

Other Examples

- Matrix Factorization Models
  - Singular Value Decomposition
  - Nonnegative Matrix Factorization
- Item Response Theory Model

$$J(w_1, w_2, \ldots, w_m, h_1, \ldots, h_n) := \sum_{i=1}^{m} \sum_{j=1}^{n} -y_{ij} \cdot (w_i - h_j) + \log\left(1 + \exp\left(w_i - h_j\right)\right).$$

- Multinomial Logistic Regression (Gopal and Yang, ICML 2012)

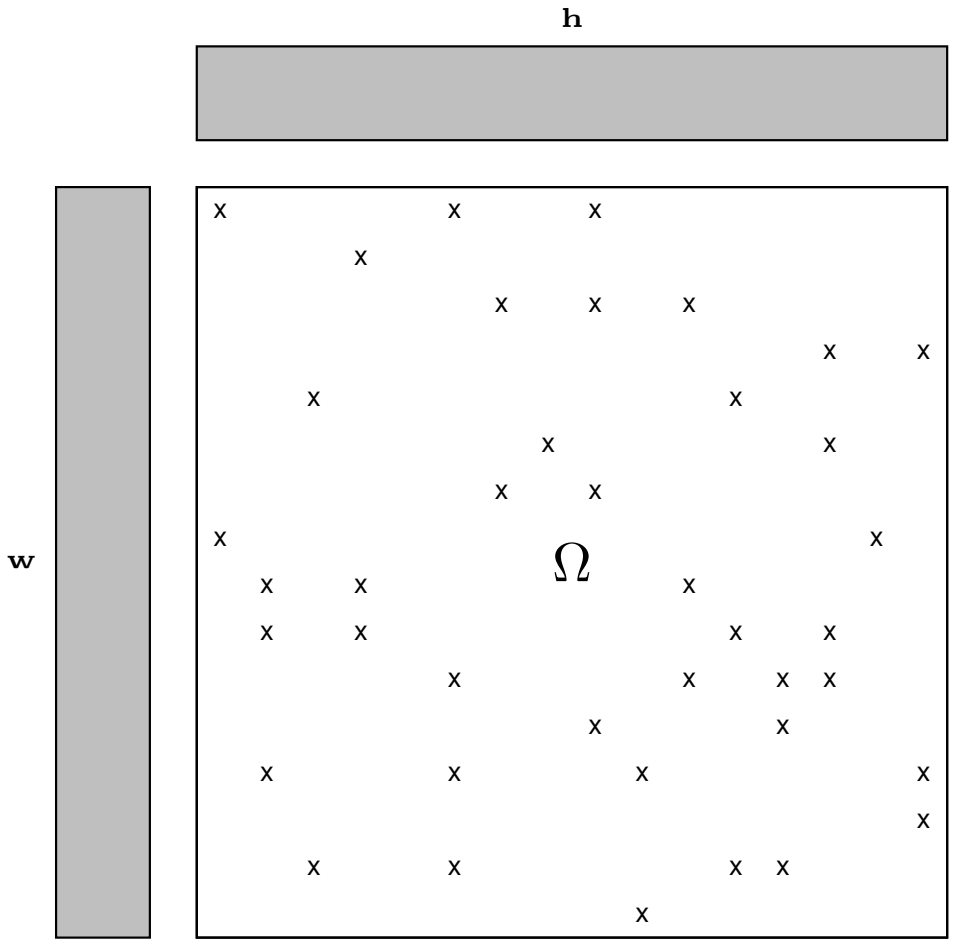
Outline

Outline

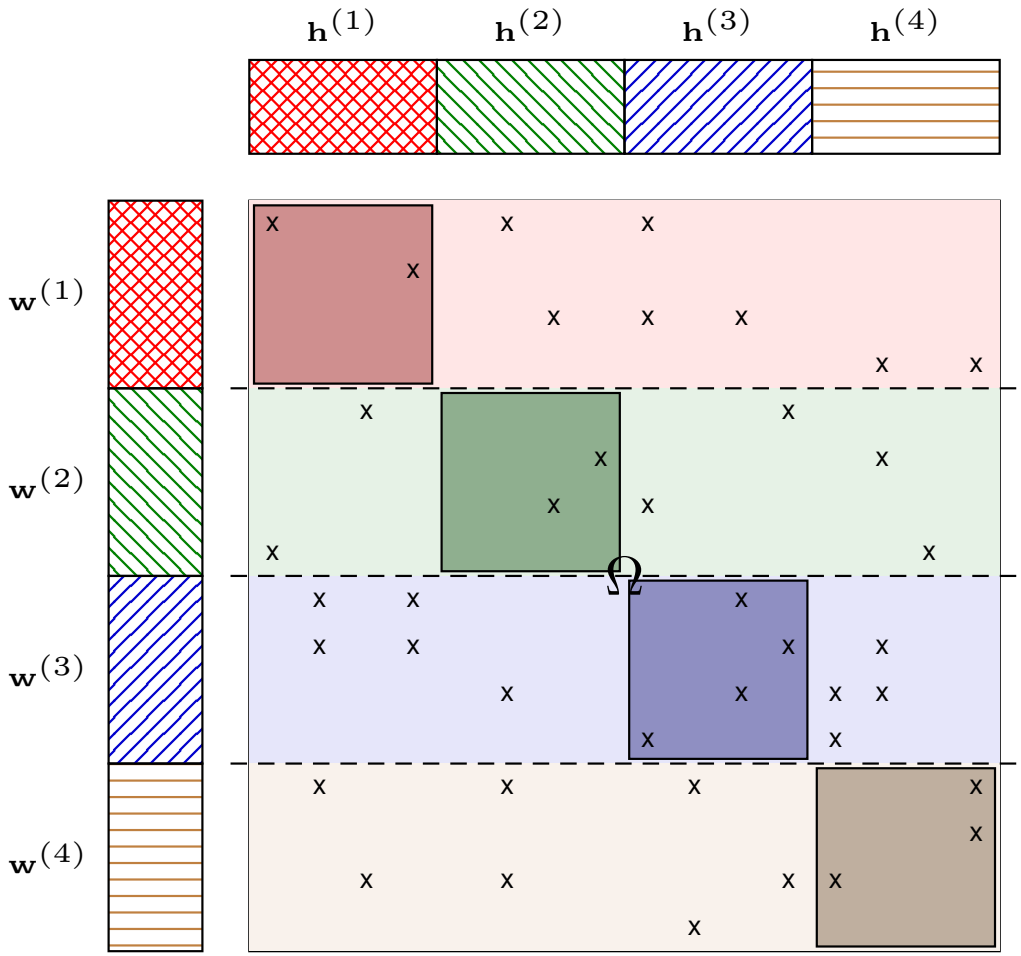
Distributed SGD/SSO

- Gemulla et al. (KDD 2011) proposed Distributed Stochastic Gradient Descent (DSGD) algorithm
- We extend it for saddle-point problems as Distributed Stochastic Saddle-point Optimization (DSSO) algorithm

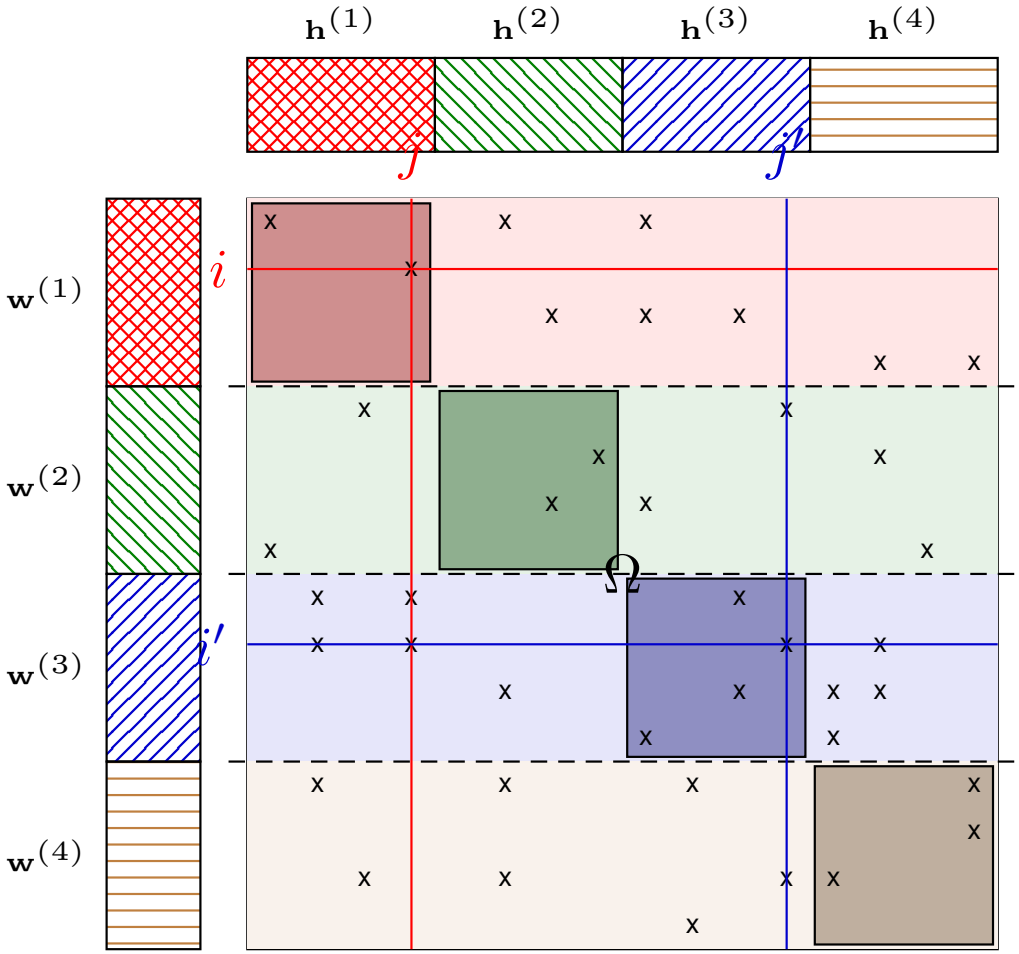
Distributed SGD/SSO



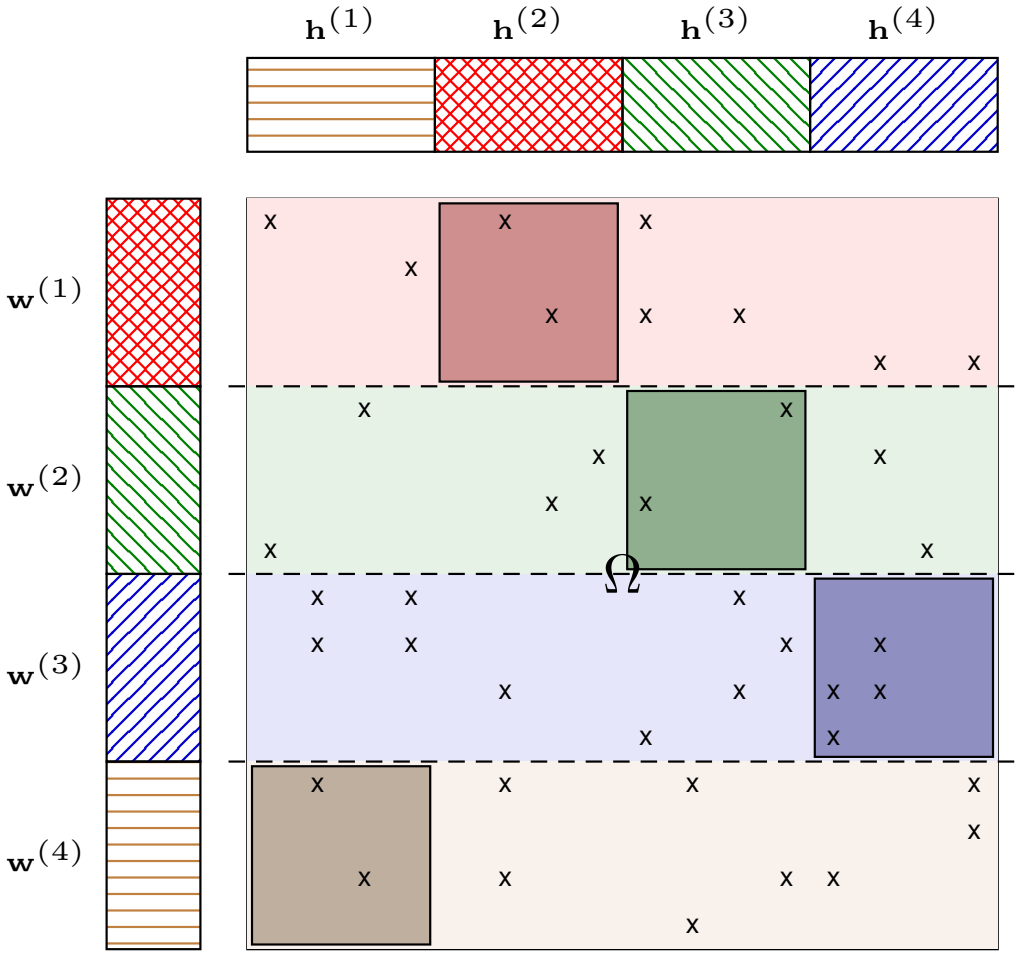
Distributed SGD/SSO



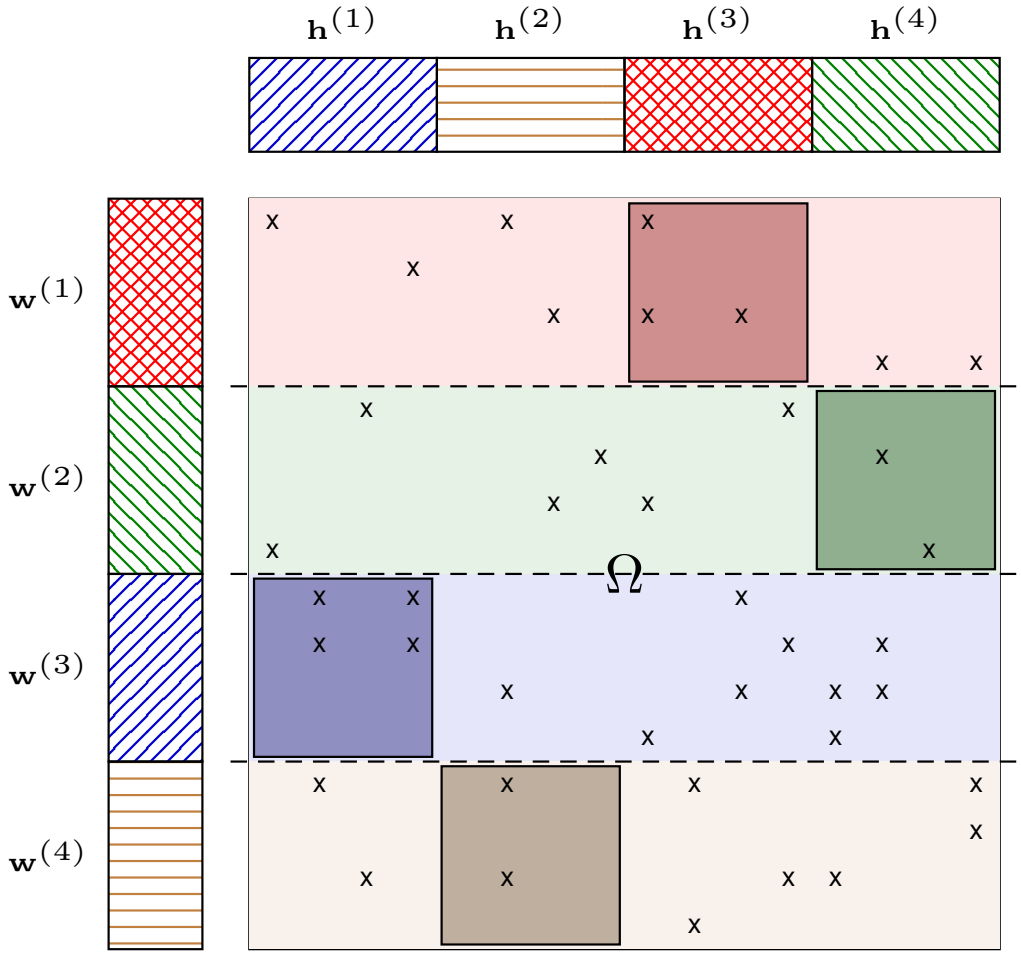
Distributed SGD/SSO



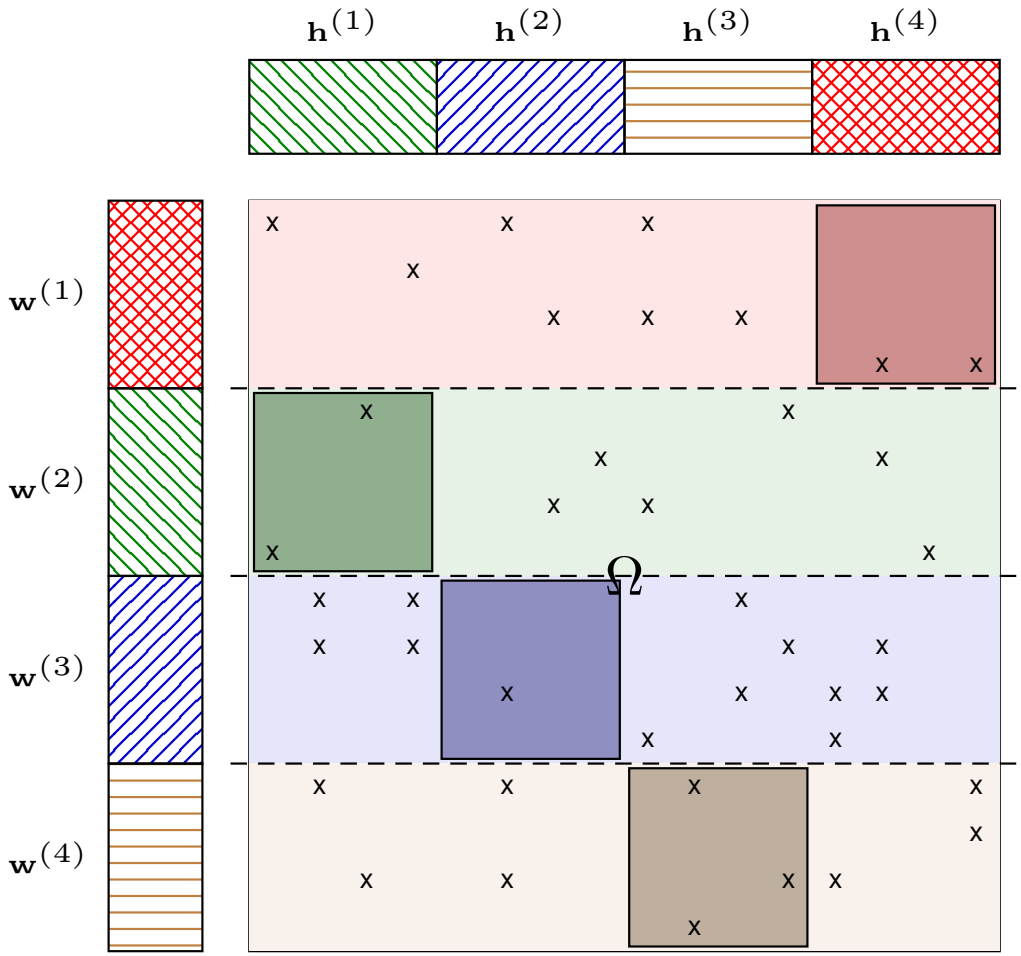
Distributed SGD/SSO



Distributed SGD/SSO



Distributed SGD/SSO



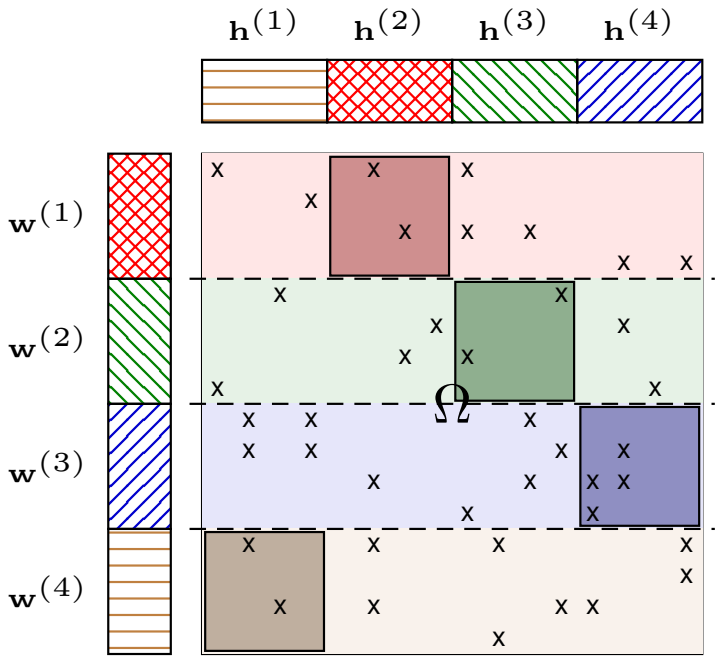
Convergence Guarantee

- Gemulla et al., 2011 showed that DSGD converges to local optimum of the *original* function with probability one, using ODE method of Kushner and Yin, 2003.
- The proof can be adapted for DSSO.

Scaling

Suppose p is the number of processors.

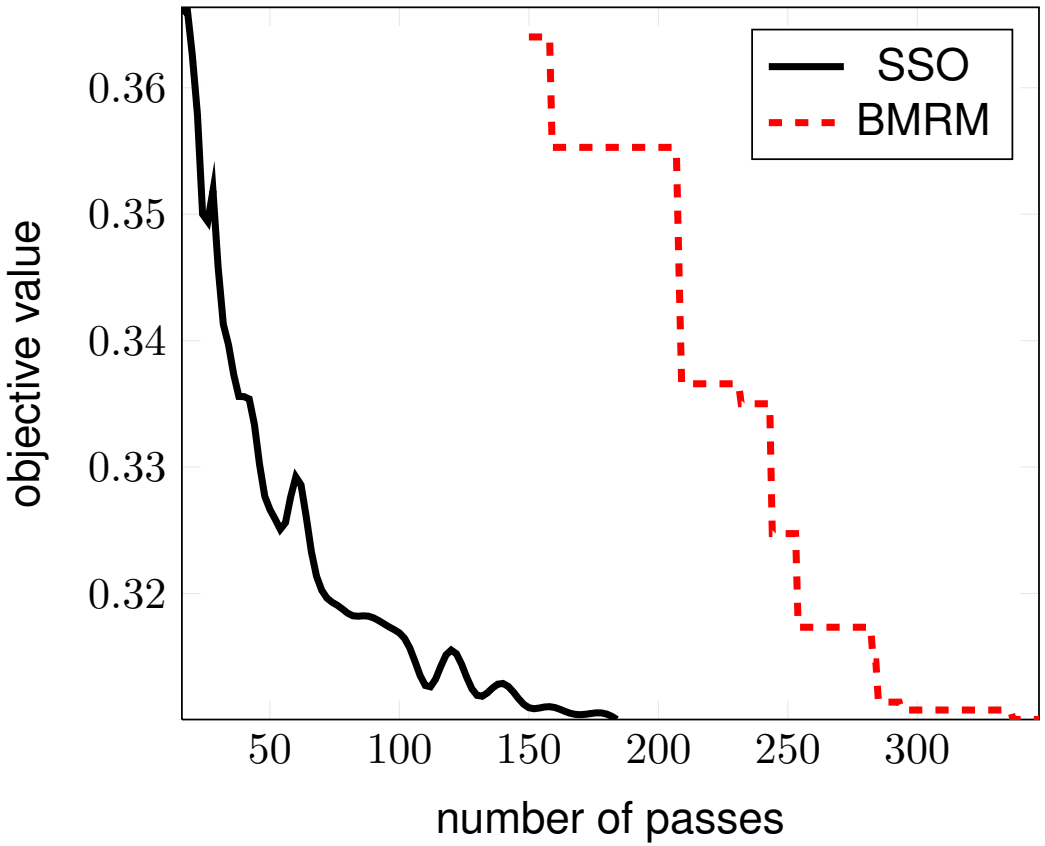
- Space complexity for storing $\mathbf{w}$, $\mathbf{h}$, Ω: $O\left(\frac{1}{p}\right)$
- Amount of communication for each iteration: $O(p)$



Experiments: Logistic Regression (Serial)

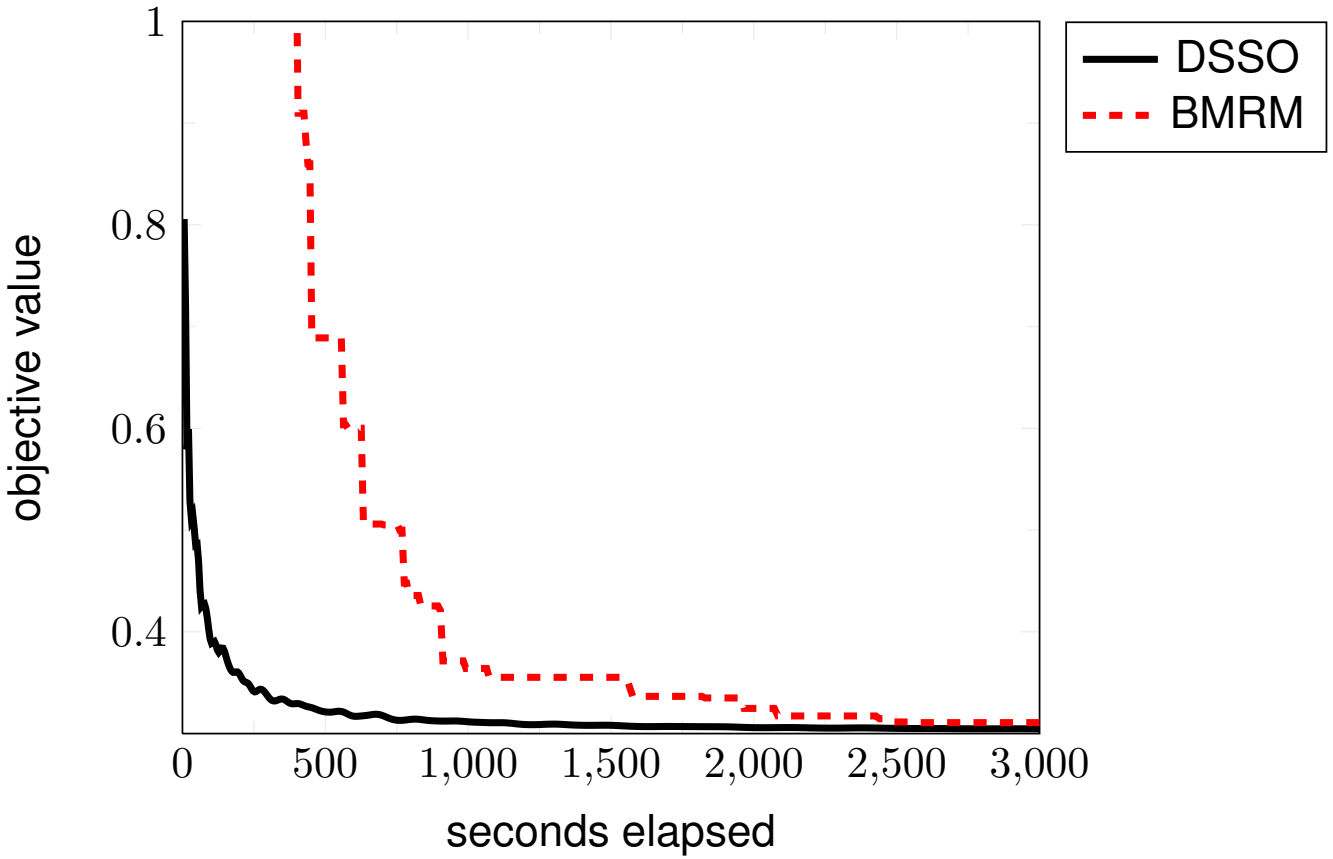
kdda: 8.4 M data points and 20 M features

$\lambda = 0.00001$



Experiments: Logistic Regression (Parallel)

kdda: 8.4 M data points and 20 M features
$\lambda = 0.00001$, machines=4, cores=8



Outline

Some observations

The good

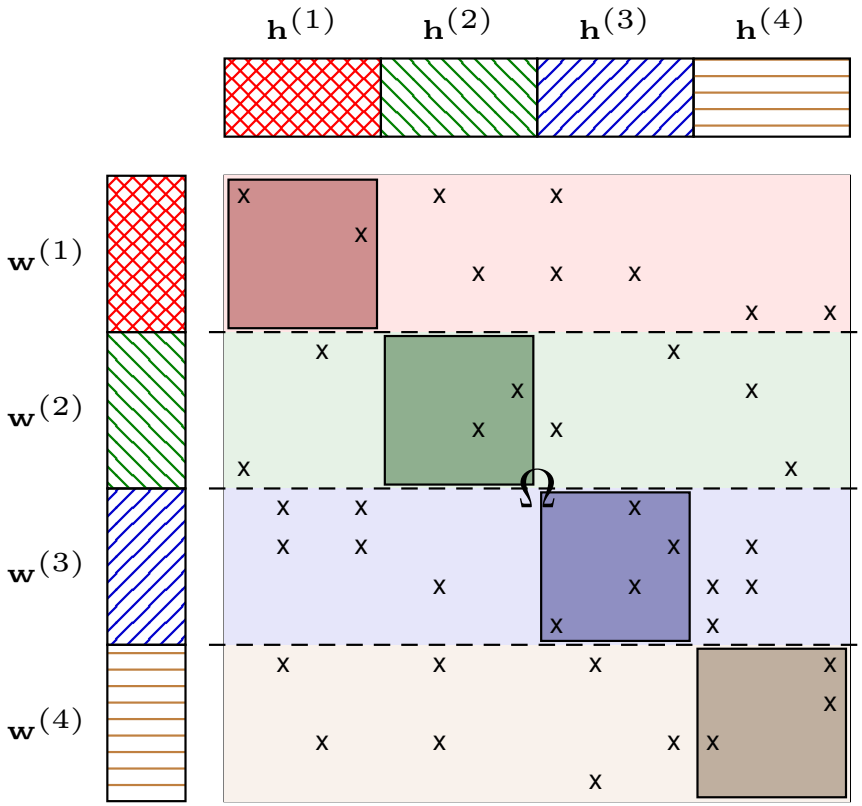
- Updates are decoupled and easy to parallelize
- Easy to implement using map-reduce

The bad

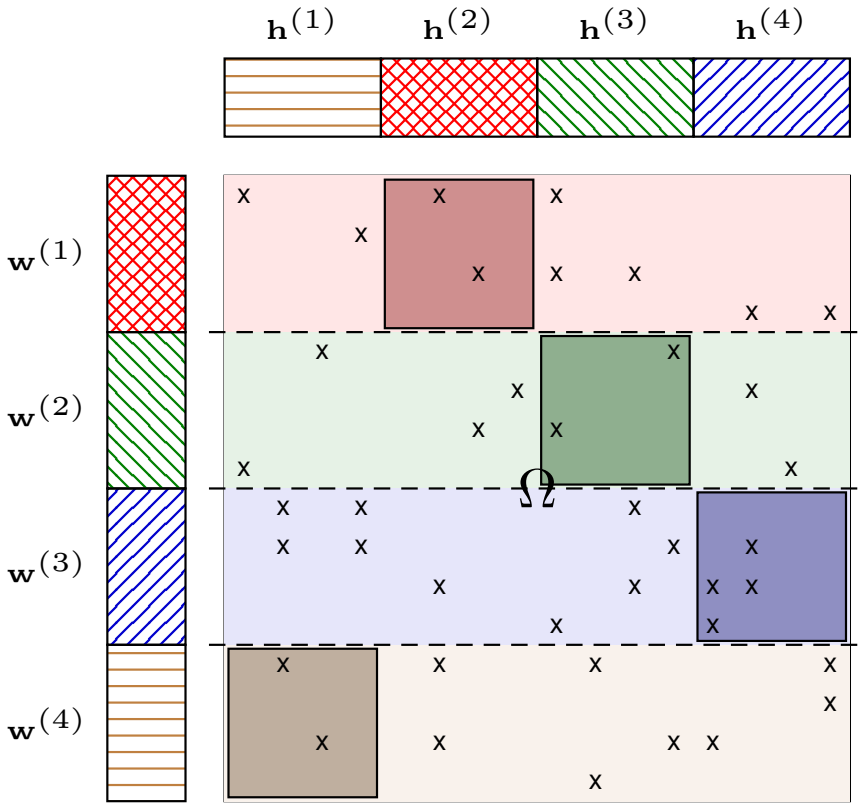
- Communication and computation are interleaved
  - When network is active then CPU is idle
  - When CPU is active then network is idle

Question: Can we keep CPU and network simultaneously busy?

Frequent synchronization harms scaling



Frequent synchronization harms scaling



Frequent synchronization harms scaling

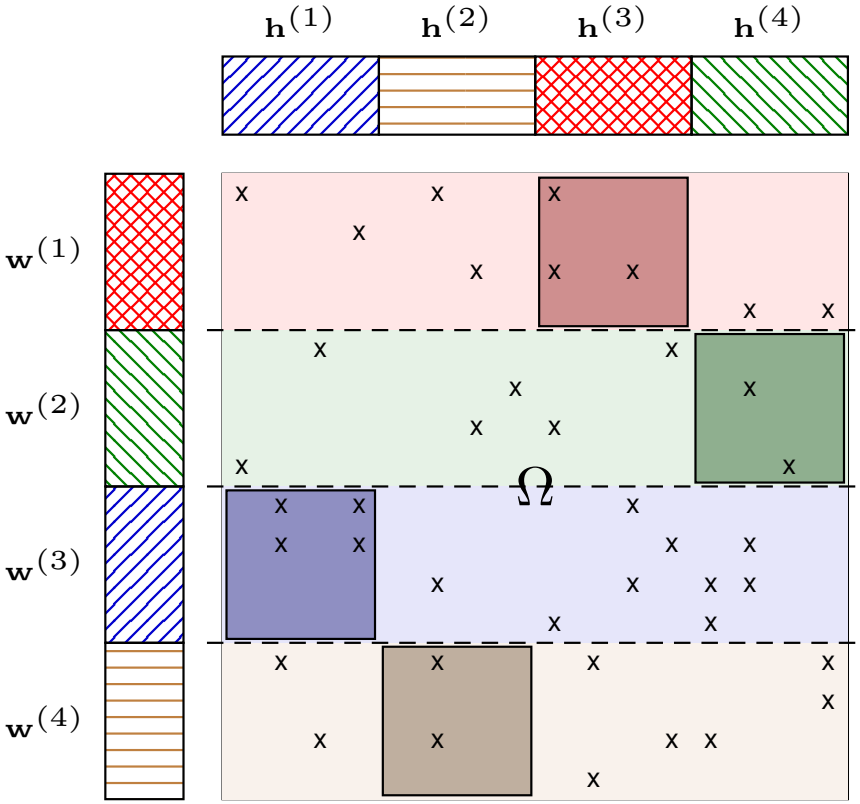


Illustration of NOMAD communication

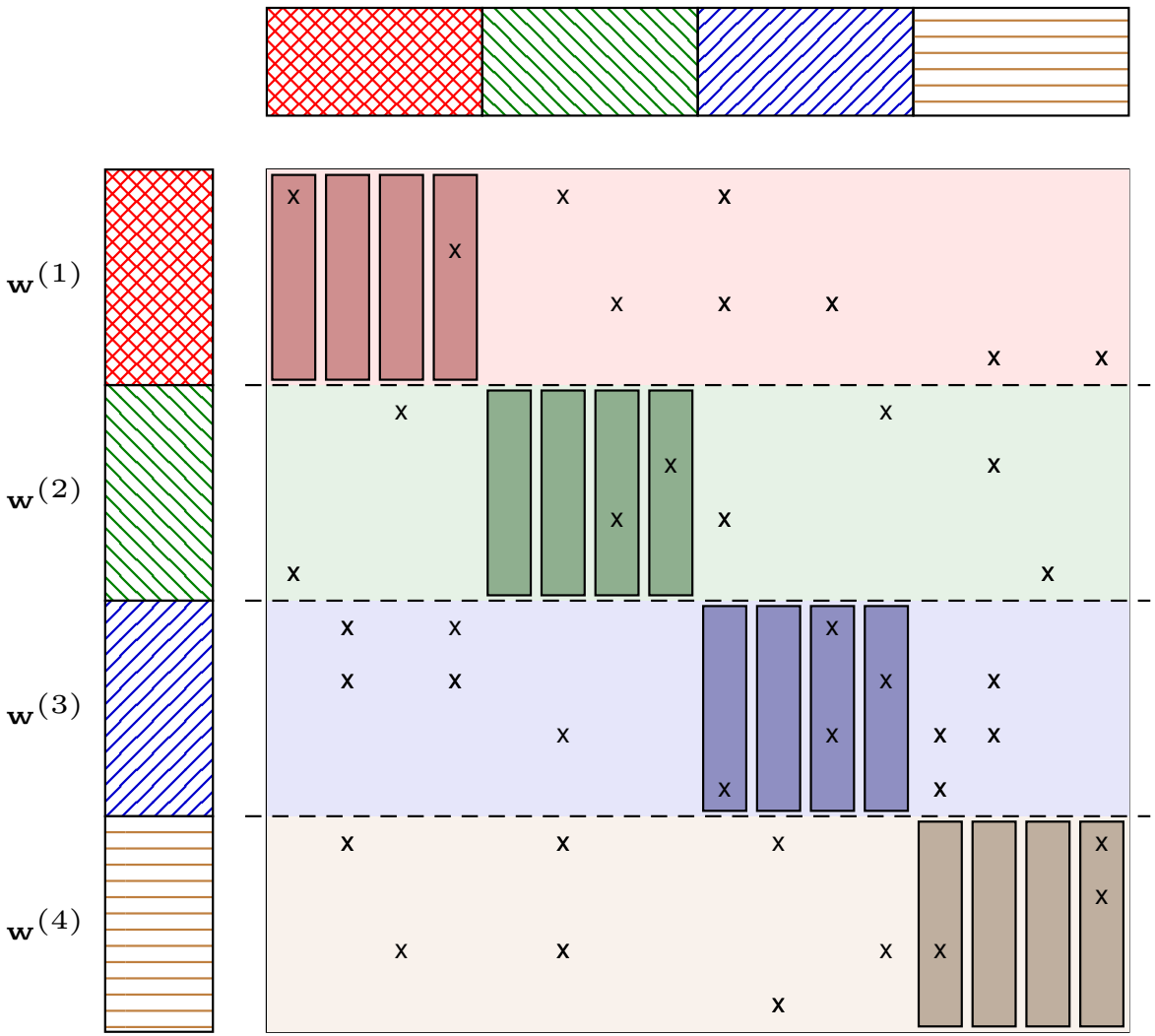


Illustration of NOMAD communication

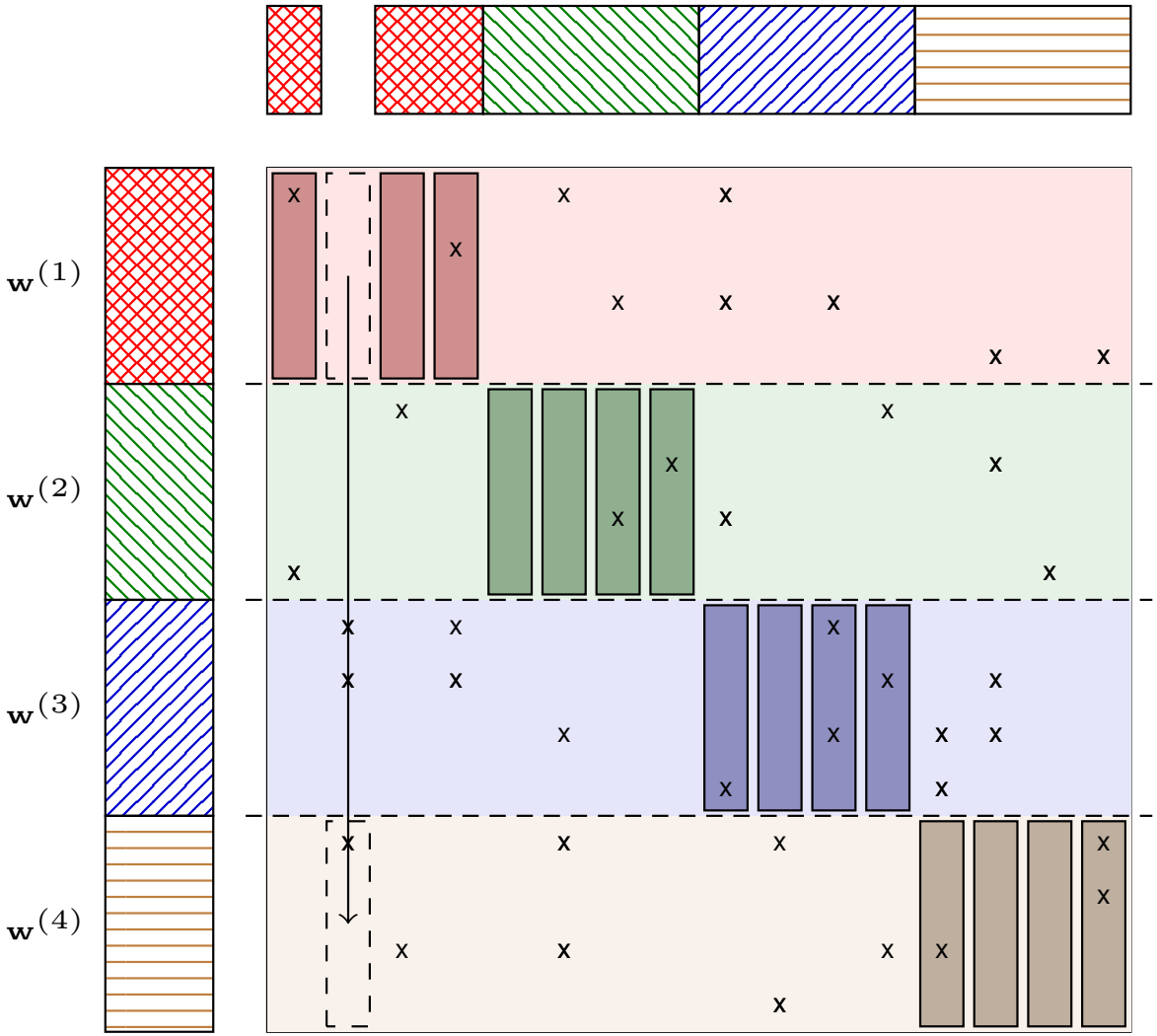


Illustration of NOMAD communication

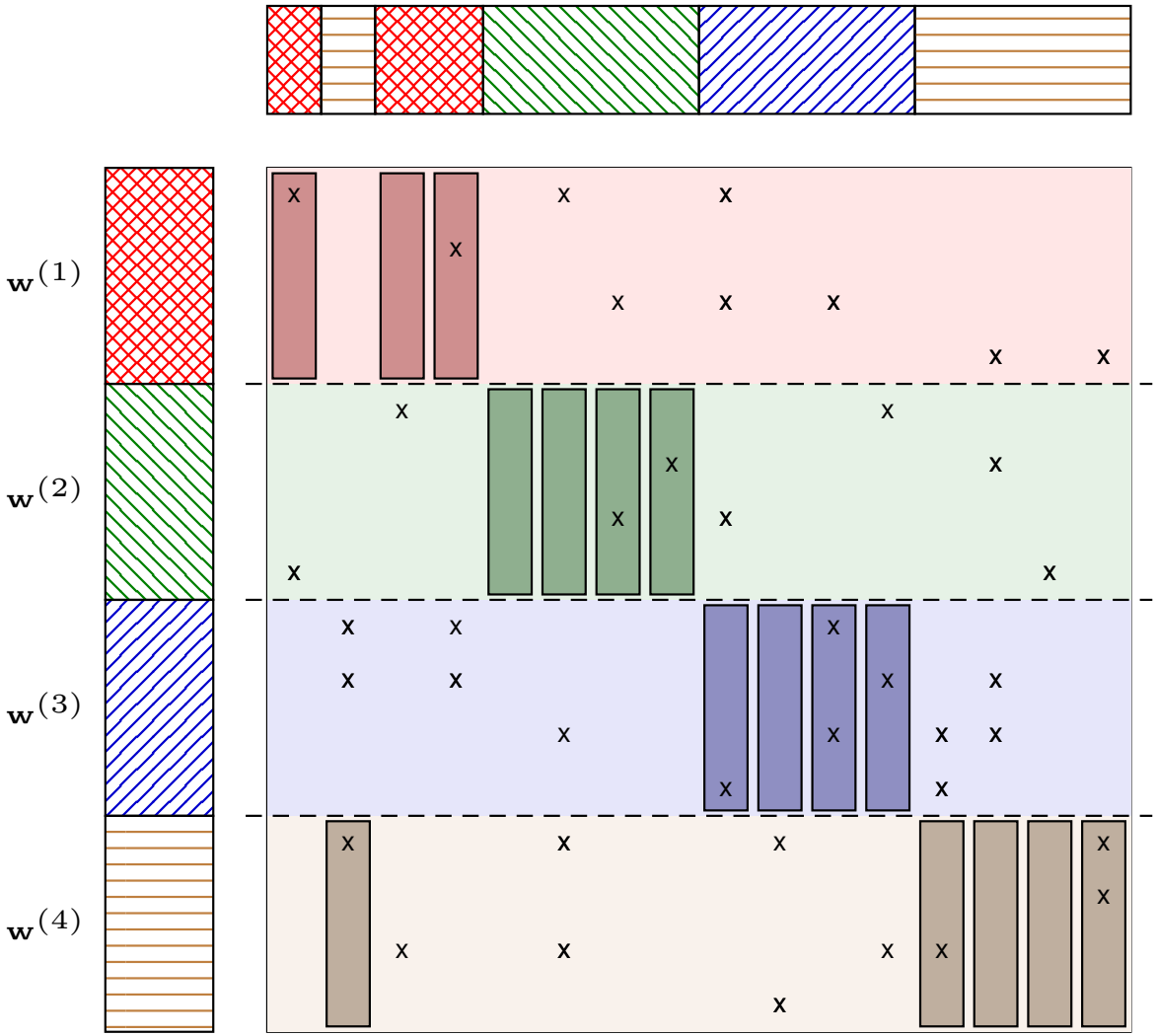


Illustration of NOMAD communication

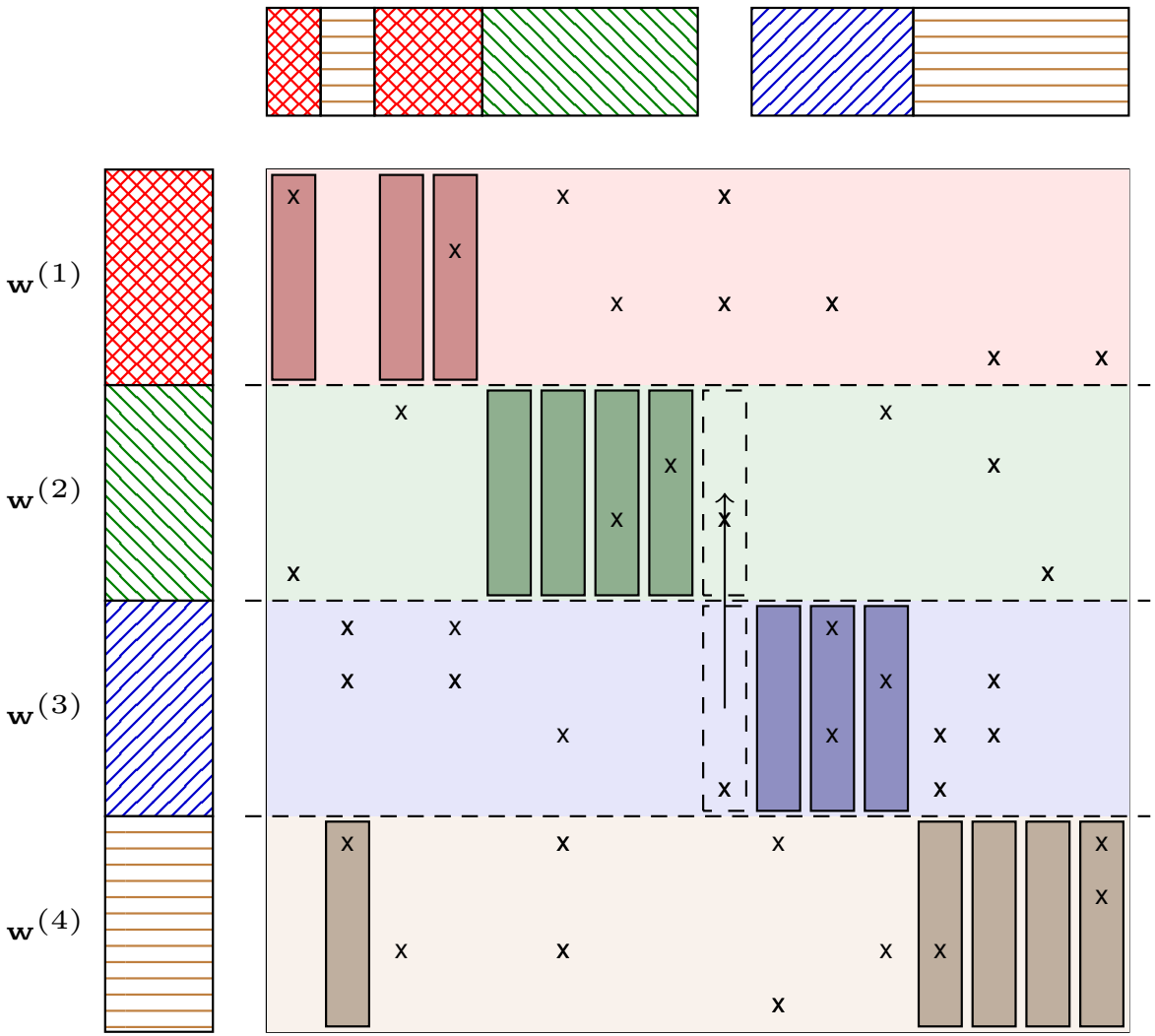


Illustration of NOMAD communication

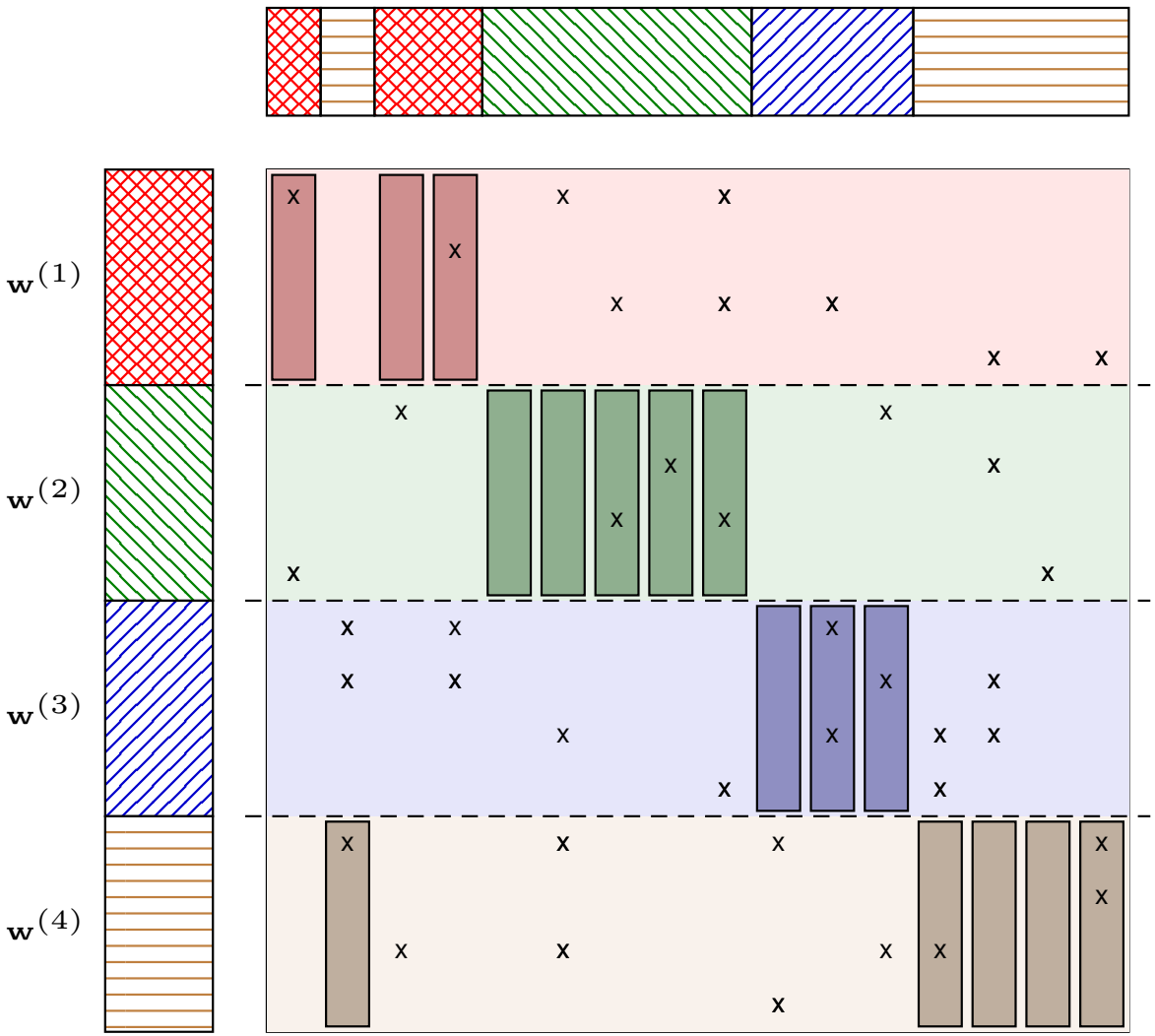


Illustration of NOMAD communication

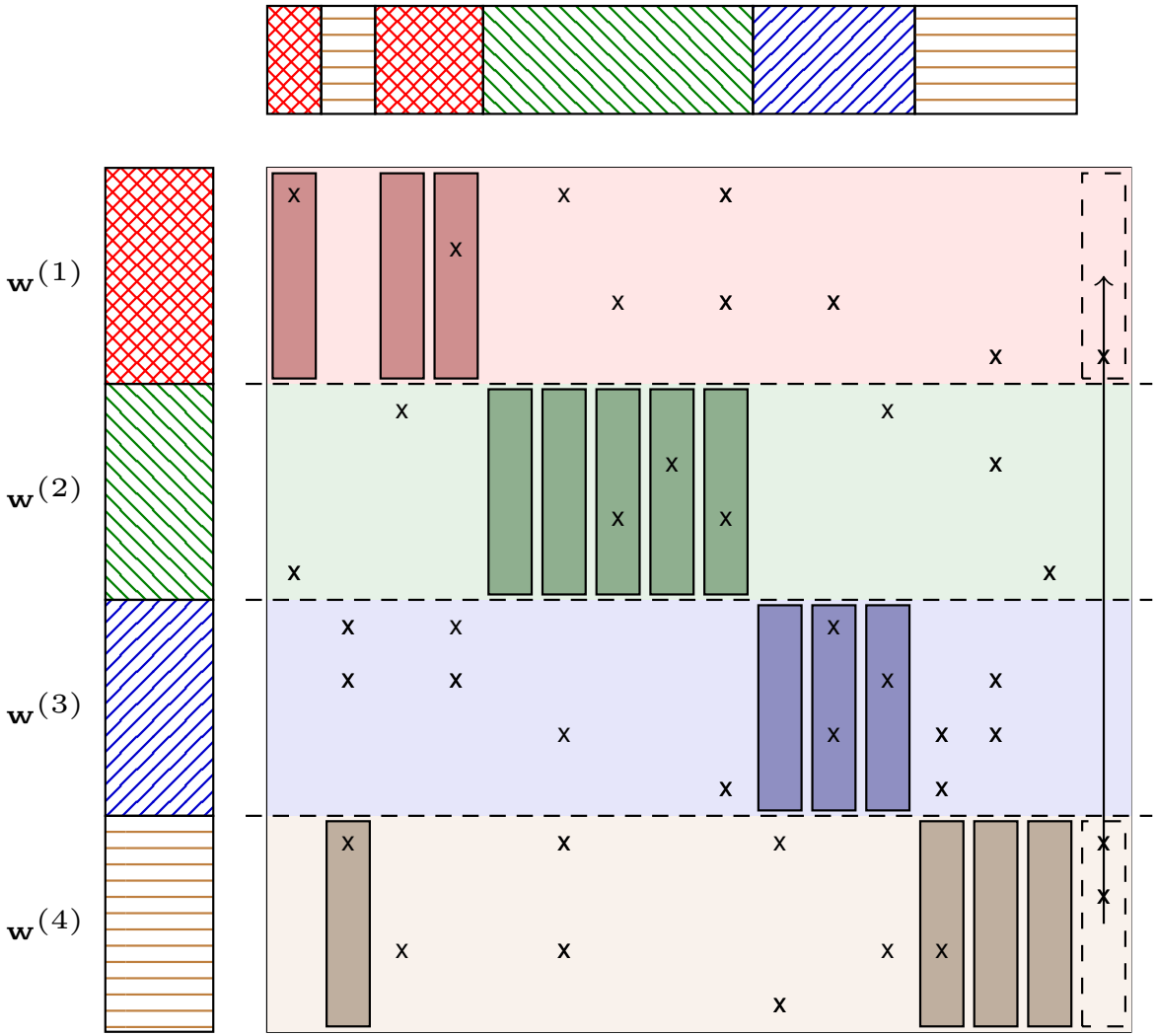


Illustration of NOMAD communication

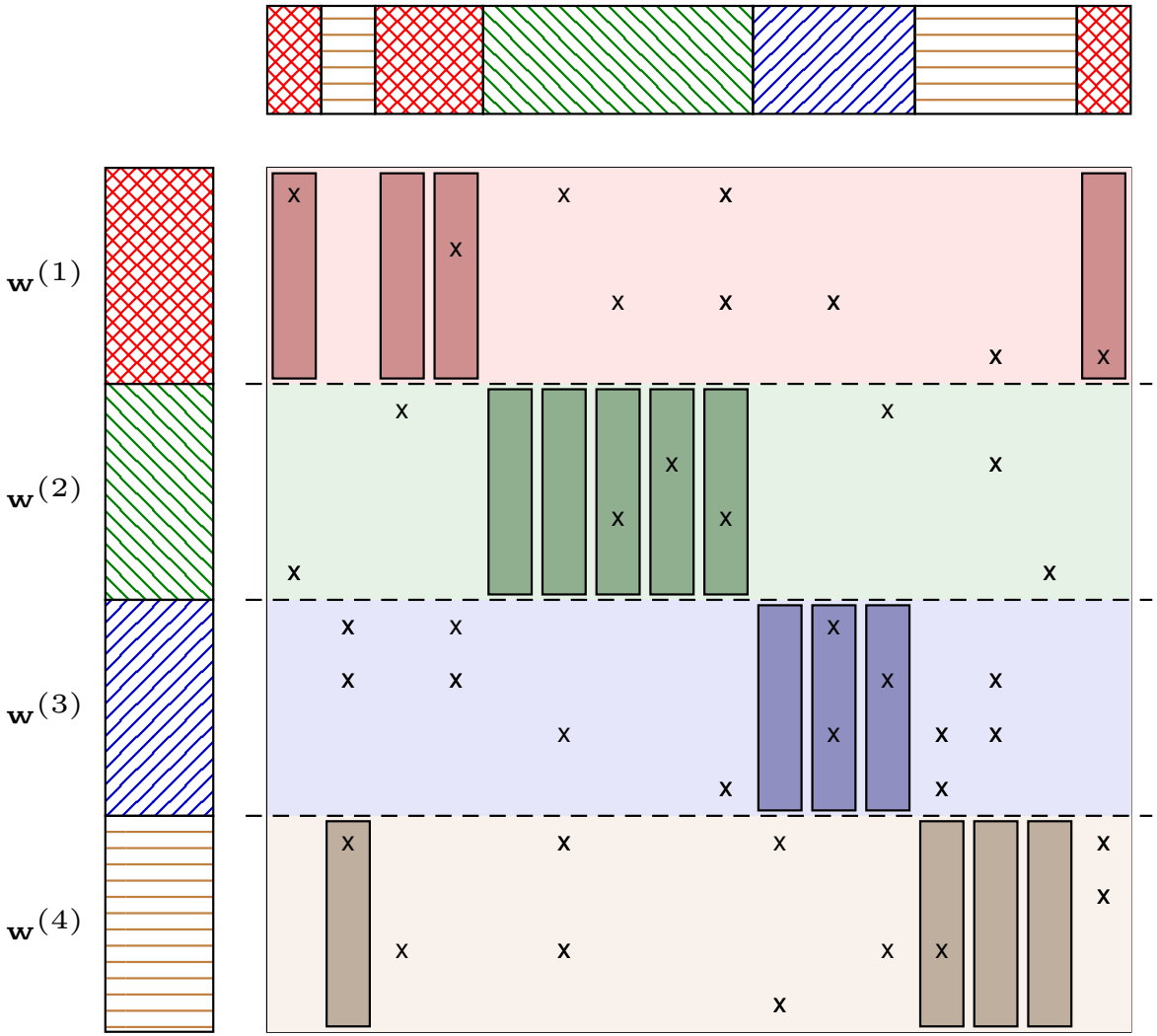


Illustration of NOMAD communication

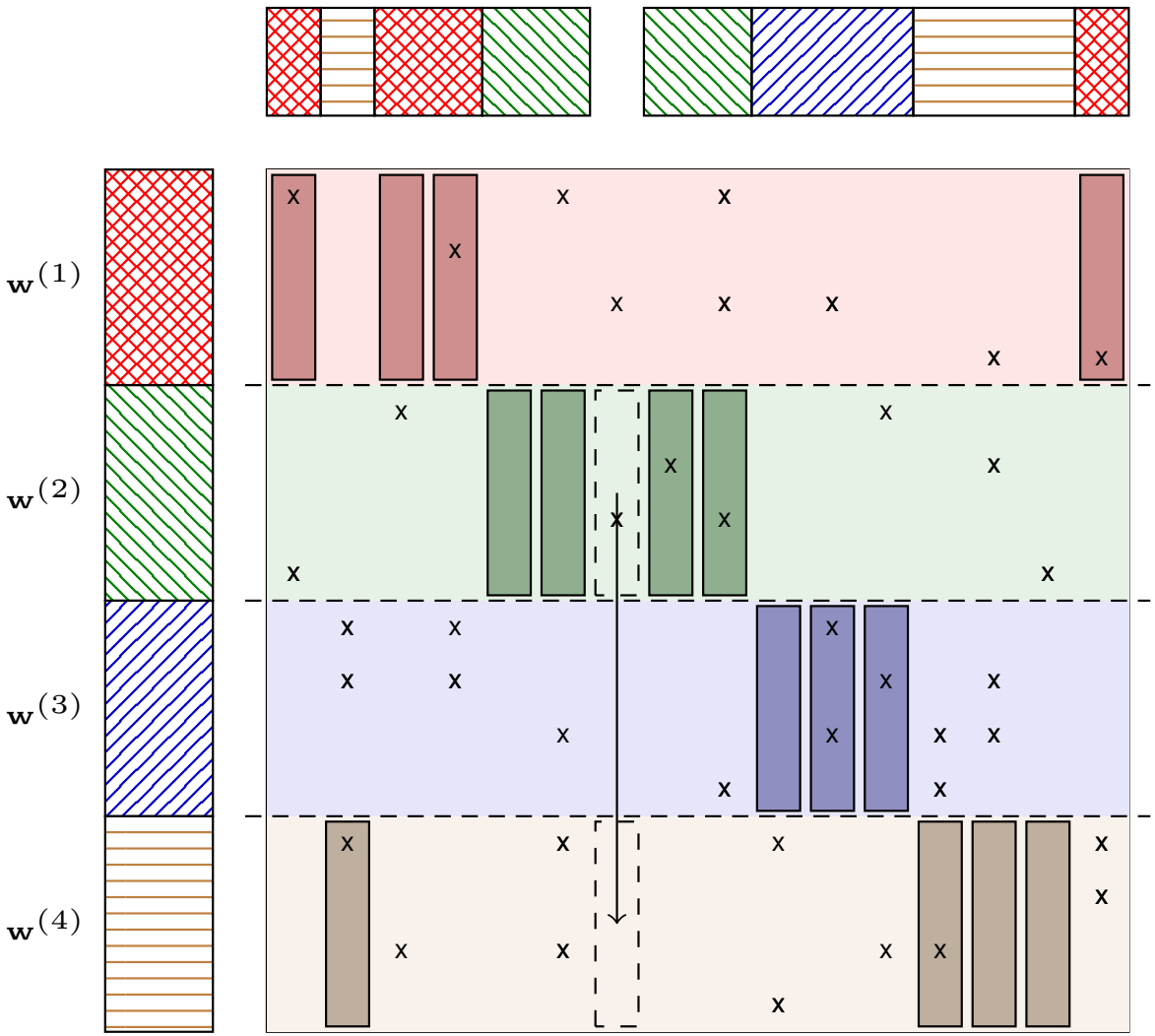


Illustration of NOMAD communication

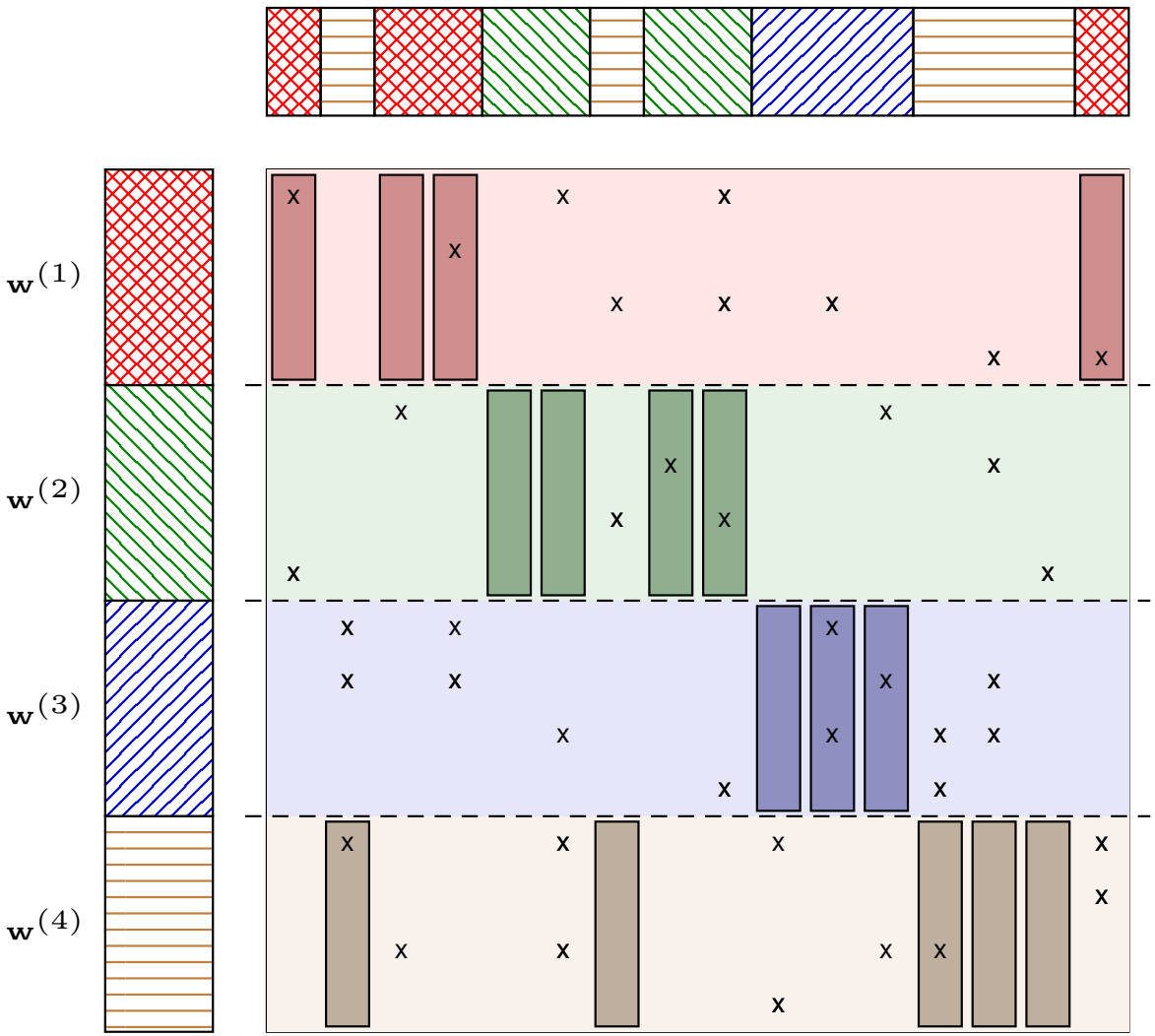


Illustration of NOMAD communication

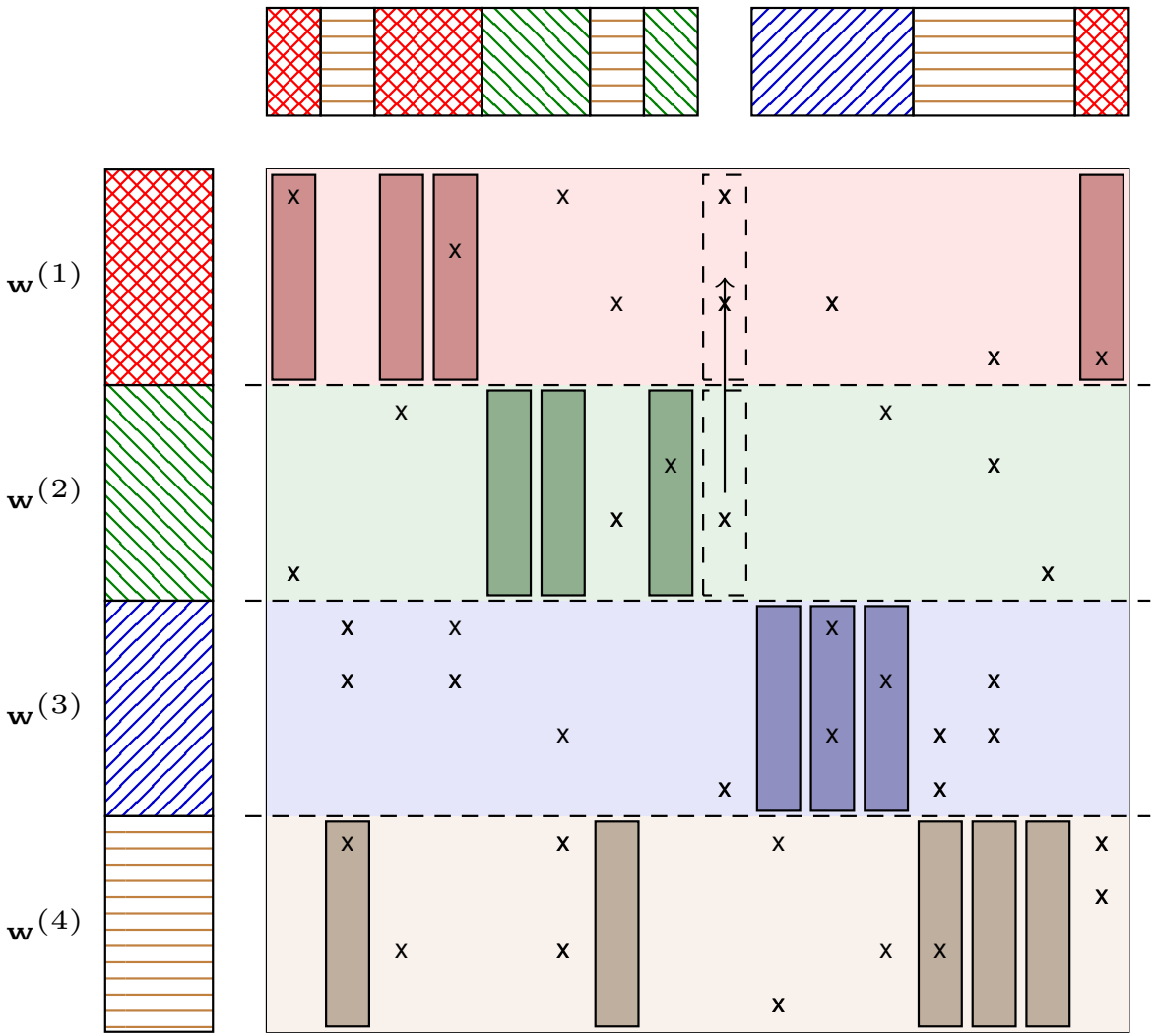
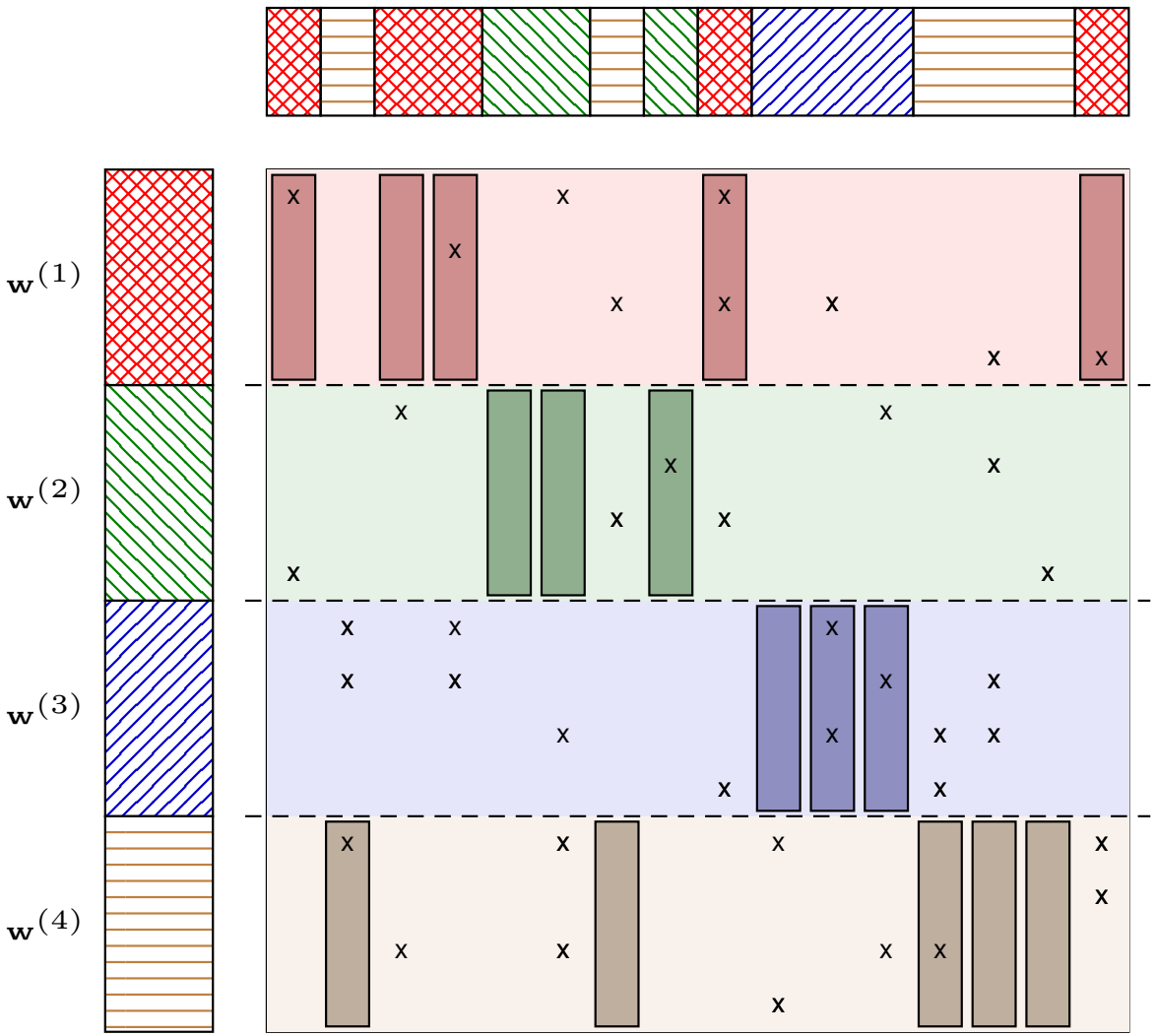
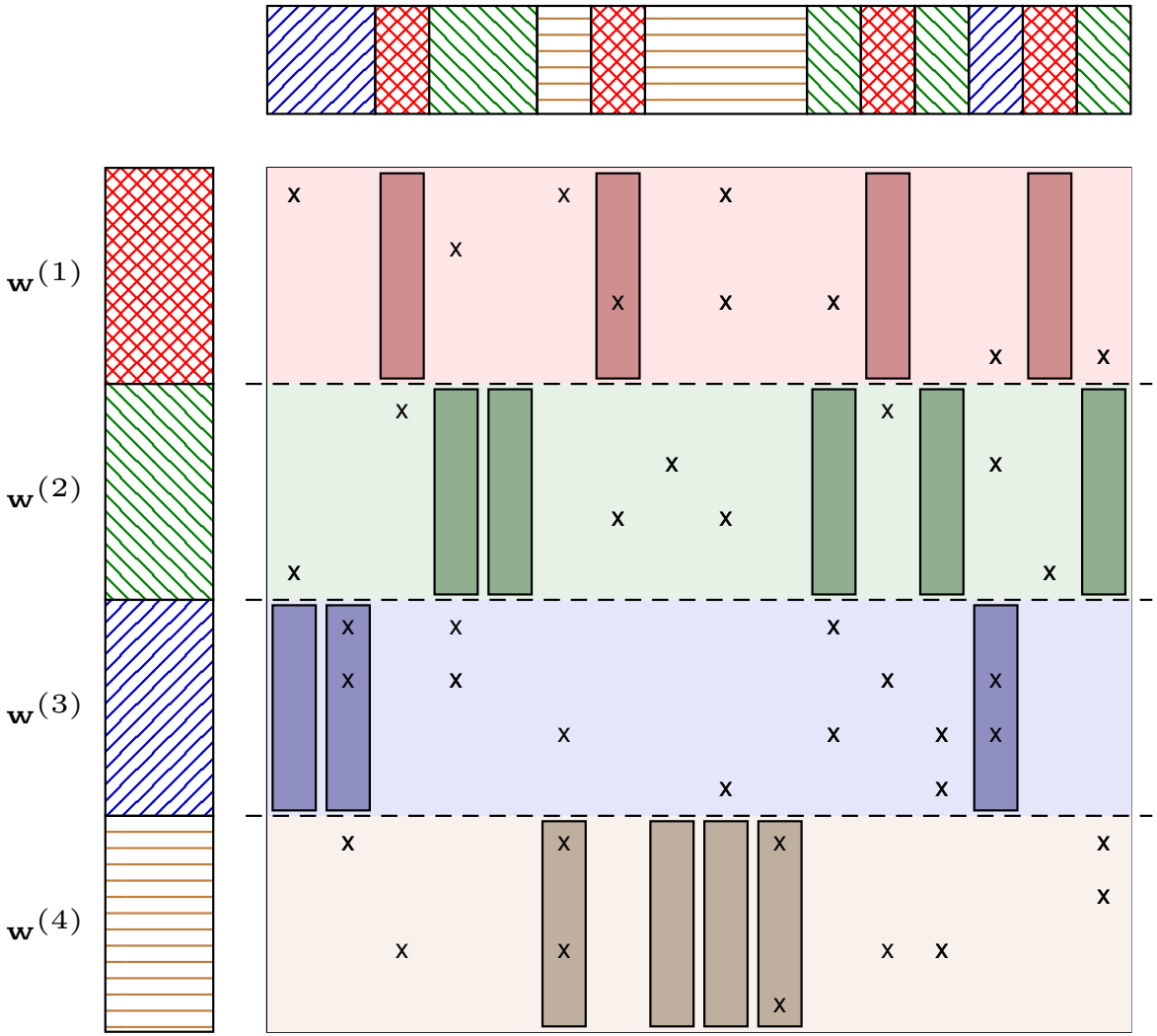


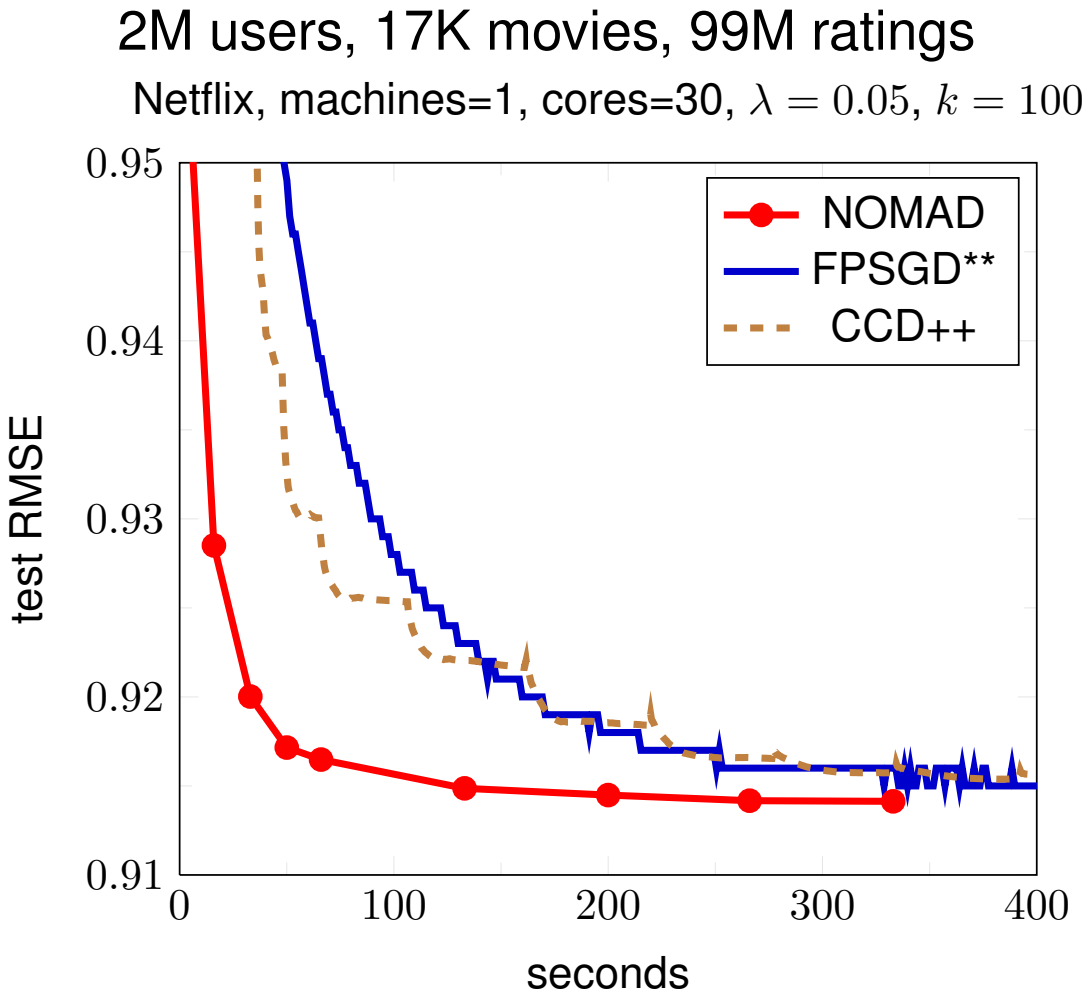
Illustration of NOMAD communication



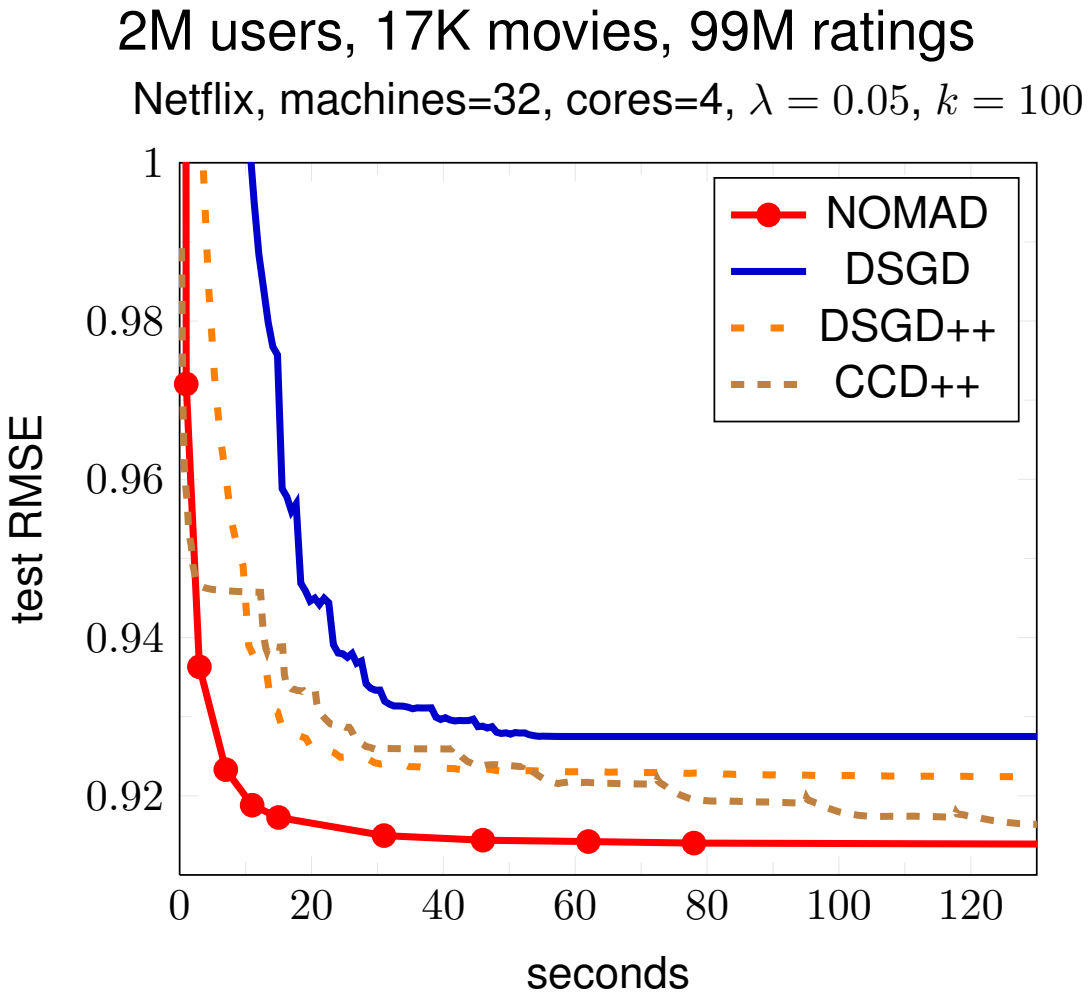
Eventually . . .



Experiments: Scaling on a single machine



Experiments: Scaling across multiple machines



Outline

Problem setting

Users and Items

- $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$: set of users
- $\mathcal{Y} = \{y_1, y_2, \ldots, y_m\}$: set of items
- $\Omega \subset \mathcal{X} \times \mathcal{Y}$: set of transactions

Learning to Rank

- $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ function which induces a ranking

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

- Objective function:

$$L(f) := \sum_{(x,y) \in \Omega} \text{rank}_f(x, y)$$

Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

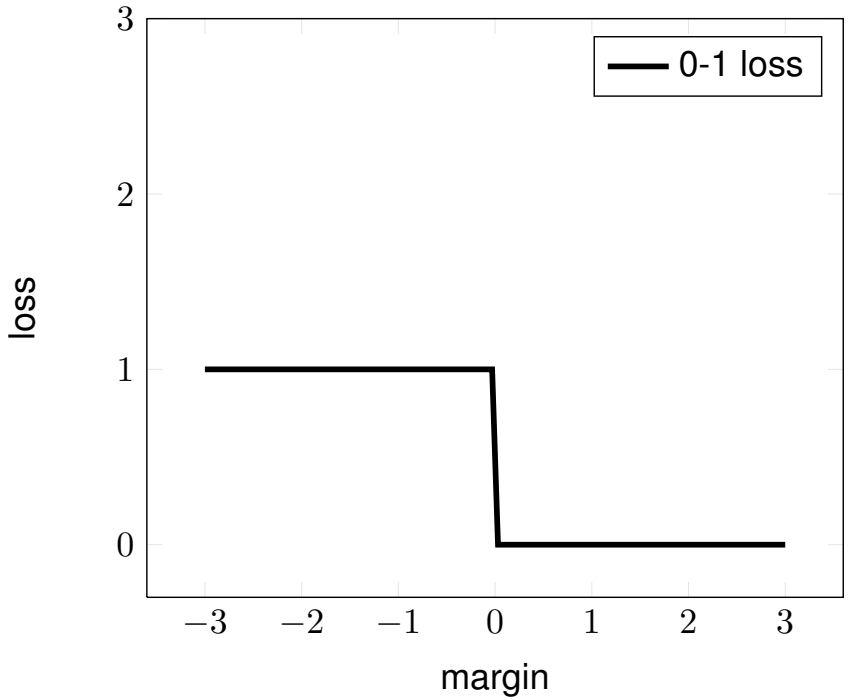
$$L(f) := \sum_{(x,y) \in \Omega} \text{rank}_f(x, y)$$

Rewriting the rank

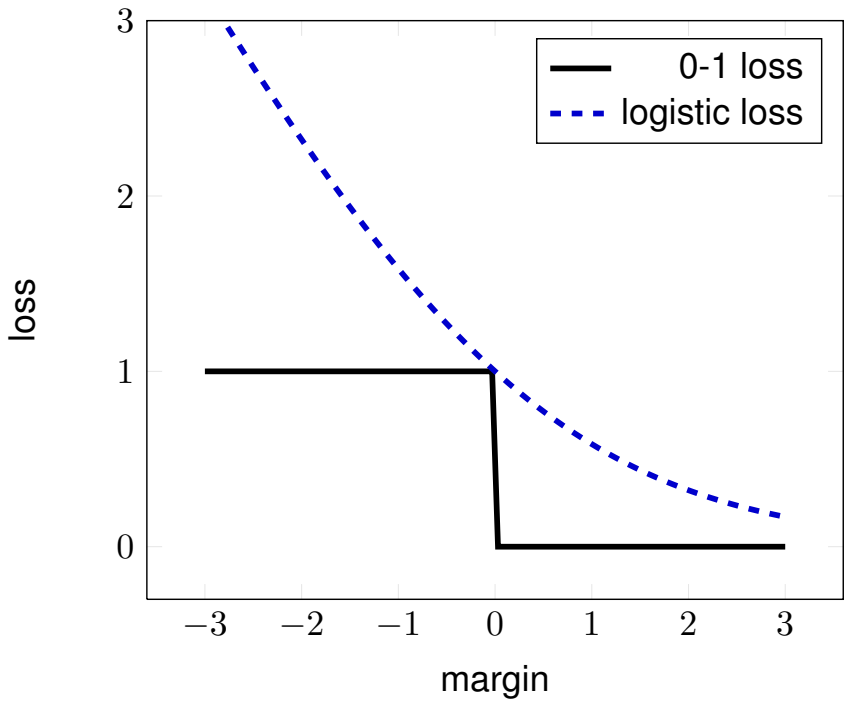
$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$L(f) := \sum_{(x,y) \in \Omega} \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

Zero-One loss



Zero-One loss



Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$L(f) := \sum_{(x,y) \in \Omega} \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

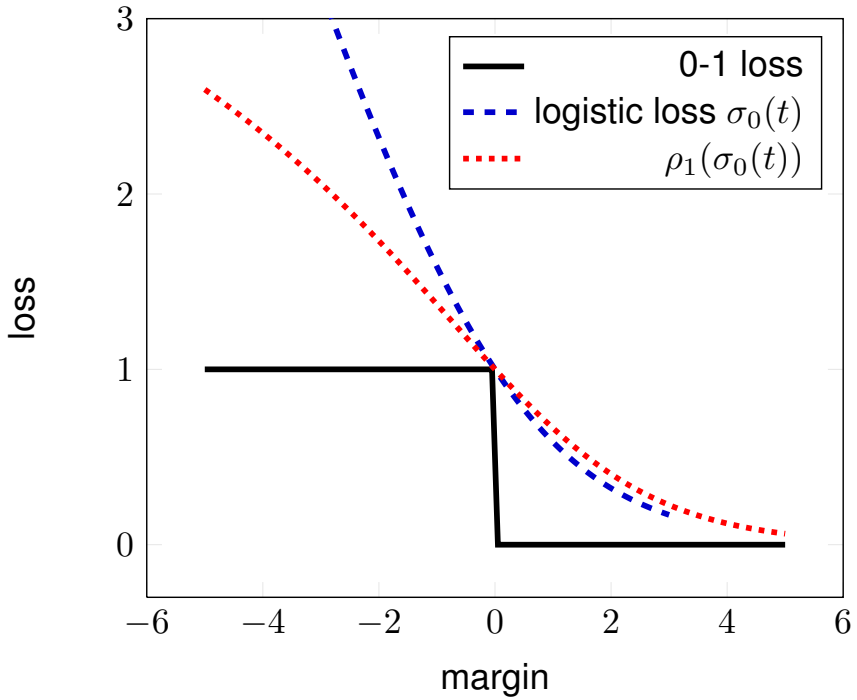
Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$\overline{L}(f) := \sum_{(x,y) \in \Omega} \sum_{y' \in \mathcal{Y}, y' \neq y} \sigma_0\left(f(x, y) - f(x, y')\right)$$

where $\sigma_0(t) = \log_2\left(1 + 2^{-t}\right)$.

Bending the loss



Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$\overline{L}(f) := \sum_{(x,y) \in \Omega} \sum_{y' \in \mathcal{Y}, y' \neq y} \sigma_0\left(f(x, y) - f(x, y')\right)$$

where $\sigma_0(t) = \log_2\left(1 + 2^{-t}\right)$.

Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$\overline{L}(f) := \sum_{(x,y) \in \Omega} \rho_1 \left(\sum_{y' \in \mathcal{Y}, y' \neq y} \sigma_0\left(f(x, y) - f(x, y')\right) \right)$$

where $\sigma_0(t) = \log_2\left(1 + 2^{-t}\right)$ and $\rho_1(t) = \log_2(t + 1)$

Rewriting the rank

$$\text{rank}_f(x, y) = \sum_{y' \in \mathcal{Y}, y' \neq y} I\left(f(x, y) - f(x, y') < 0\right)$$

$$\overline{L}(f) := \sum_{(x,y) \in \Omega} \rho_1 \left(\sum_{y' \in \mathcal{Y}, y' \neq y} \sigma_0\left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle\right) \right)$$

$$\sigma_0(t) = \log_2\left(1 + 2^{-t}\right), \rho_1(t) = \log_2(t + 1)$$

Stochastic gradients

$$\overline{L}(f) := \sum_{(x,y)\in\Omega} \rho_1 \left(\sum_{y'\in\mathcal{Y}, y'\neq y} \sigma_0 \left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle \right) \right)$$

$$\nabla \overline{L}_{x,y}(f) := |\Omega| \, \nabla \rho_1 \left(\sum_{y'\in\mathcal{Y}, y'\neq y} \sigma_0 \left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle \right) \right)$$

Bounding ρ_1 (see Gopal and Yang, 2012)

$$\rho_1(t) = \log_2(t+1) \leq -\log_2 \xi + \frac{\xi \cdot (t+1) - 1}{\log 2}.$$

$$\overline{L}(f) := \sum_{(x,y)\in\Omega} \rho_1 \left(\sum_{y'\in\mathcal{Y}, y'\neq y} \sigma_0 \left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle \right) \right)$$

Bounding ρ_1 (see Gopal and Yang, 2012)

$$\rho_1(t) = \log_2(t+1) \leq -\log_2 \xi + \frac{\xi \cdot (t+1) - 1}{\log 2}.$$

$$\overline{L}(f) := \sum_{(x,y)\in\Omega} -\log_2 \xi_{xy} + \frac{\xi_{xy}\left(\sum_{y'\neq y} \sigma_0(\langle U_x, V_y\rangle - \langle U_x, V_{y'}\rangle) + 1\right) - 1}{\log 2}$$

Alternate minimization

(U, V)-step

$$\nabla_{U,V} \overline{L}(U, V, \xi) := \frac{1}{\log 2} \sum_{(x,y)\in\Omega} \xi_{xy} \left(\sum_{y' \neq y} \nabla_{U,V} \sigma_0(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle) \right)$$

- Sample (x, y) uniformly from Ω
- Sample y' uniformly from $\mathcal{Y} \setminus \{y\}$
- Estimate the gradient by

$$\frac{|\Omega| \cdot (|\mathcal{Y}| - 1) \cdot \xi_{xy}}{\log 2} \cdot \nabla_{U,V} \sigma_0(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle)$$

Alternate minimization

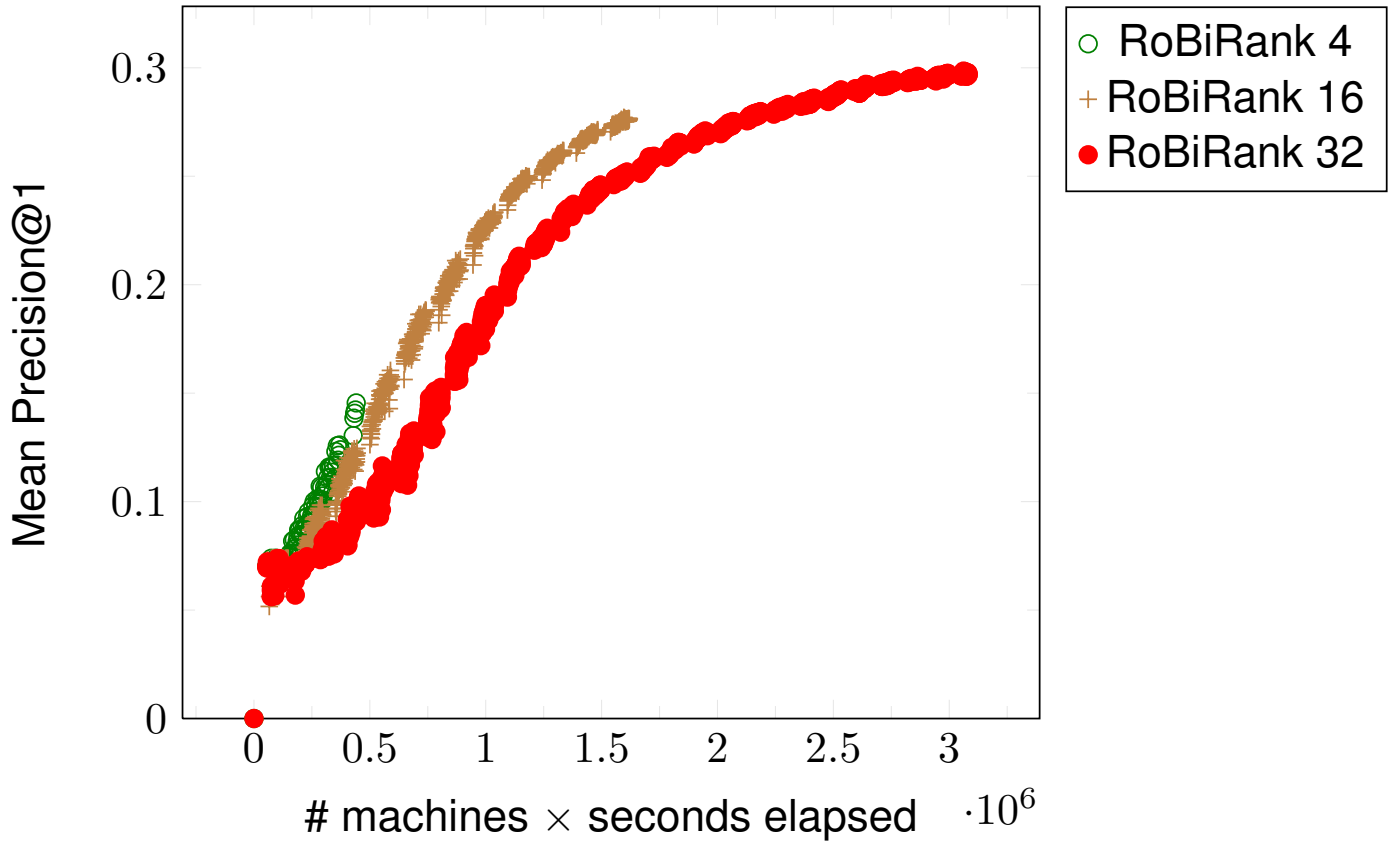
ξ_{xy}-step

If U and V are fixed then

$$\xi_{xy} = \frac{1}{\sum_{y' \neq y} \sigma_0(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle) + 1}$$

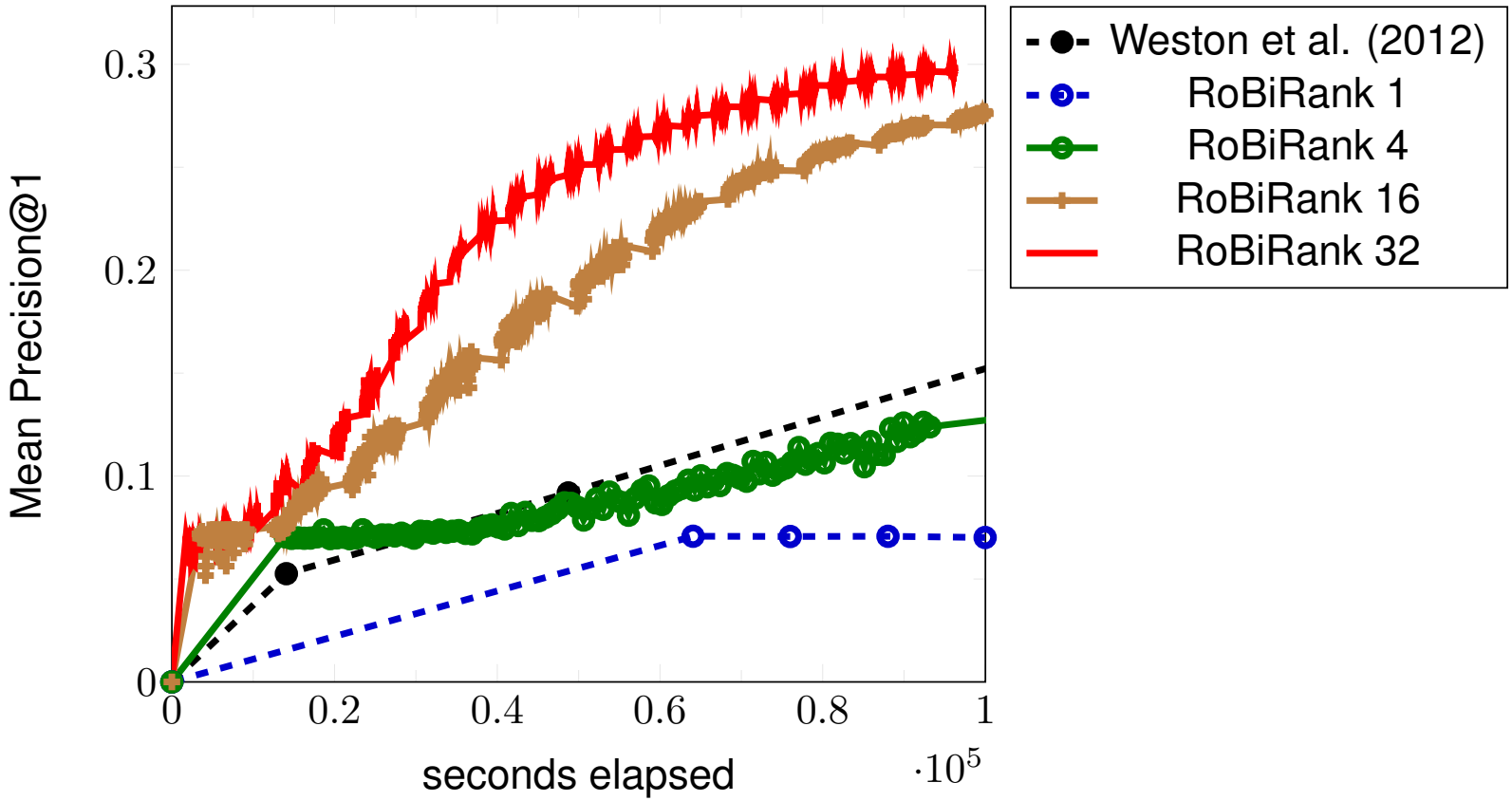
Experiments: Million song dataset

1,129,318 users, 386,133 songs, and 49,824,519 records



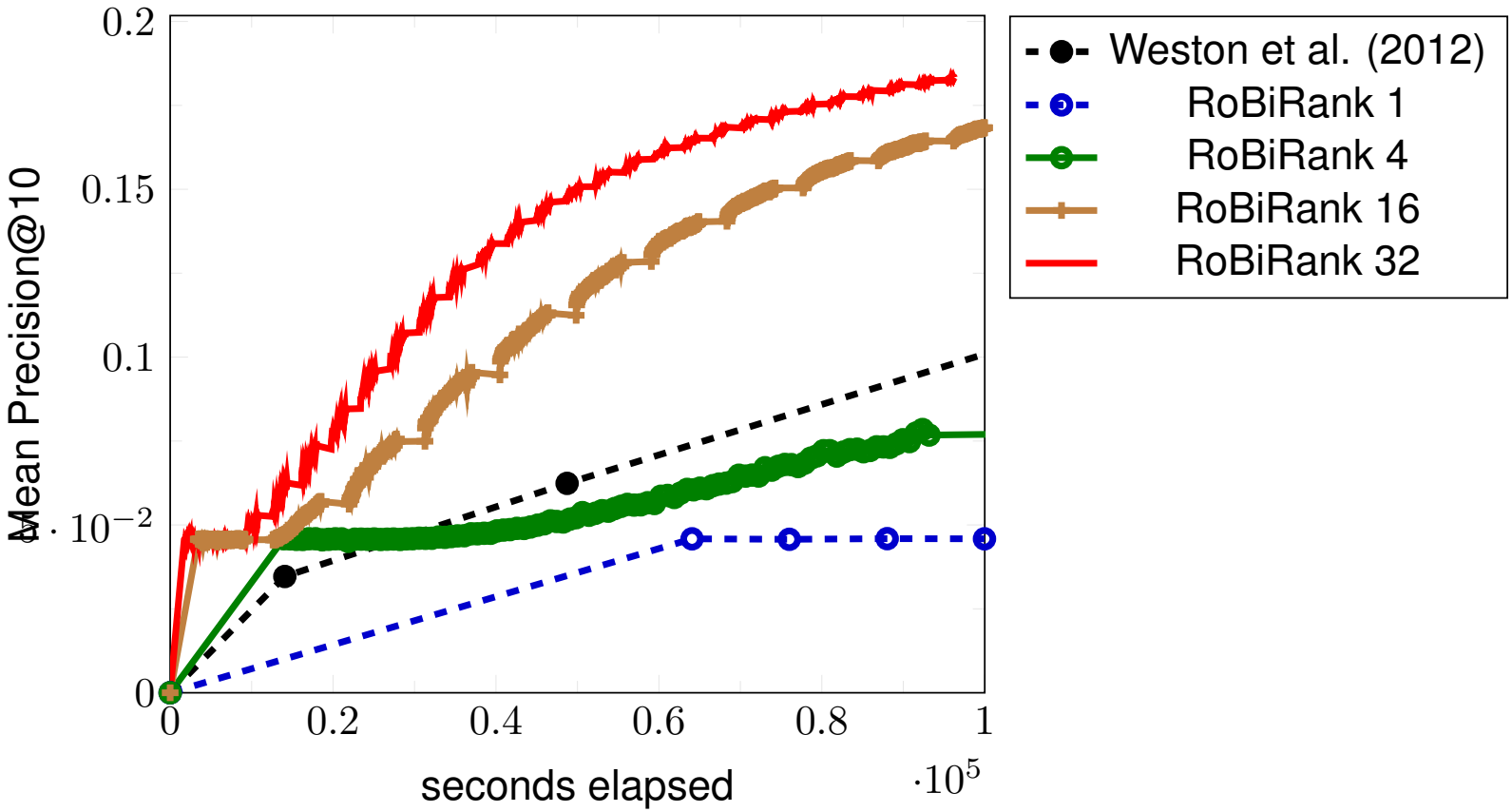
Experiments: Million song dataset

1,129,318 users, 386,133 songs, and 49,824,519 records



Experiments: Million song dataset

1,129,318 users, 386,133 songs, and 49,824,519 records



Summary

Conclusion

- Double separability can be found in many statistical models
- It can be exploited to parallelize stochastic optimization

Future Goals

- Finding more applications
- Further generalizations (separability of finite order?)
- Support other inference methods

Collaborators





Q & A

Thanks!