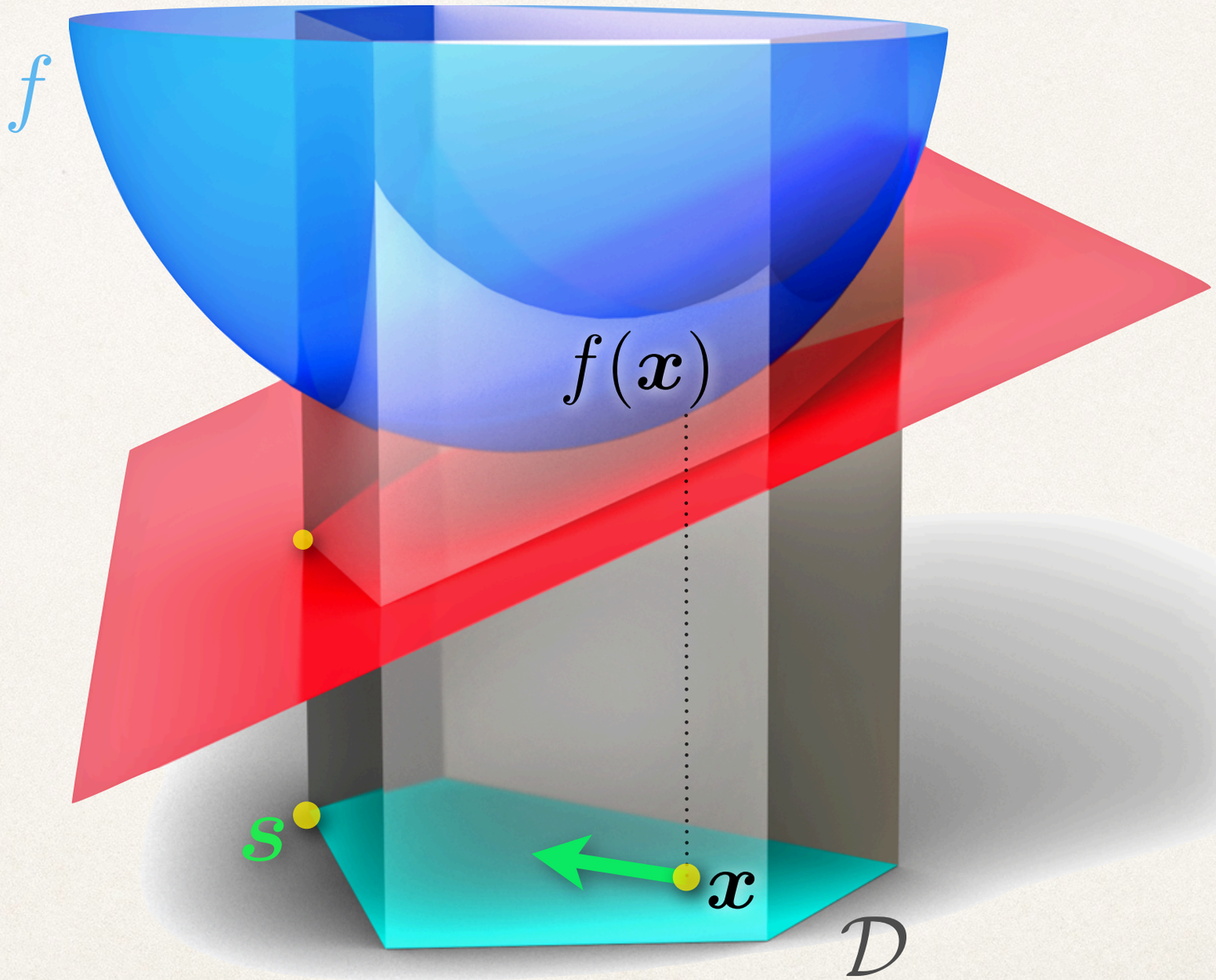
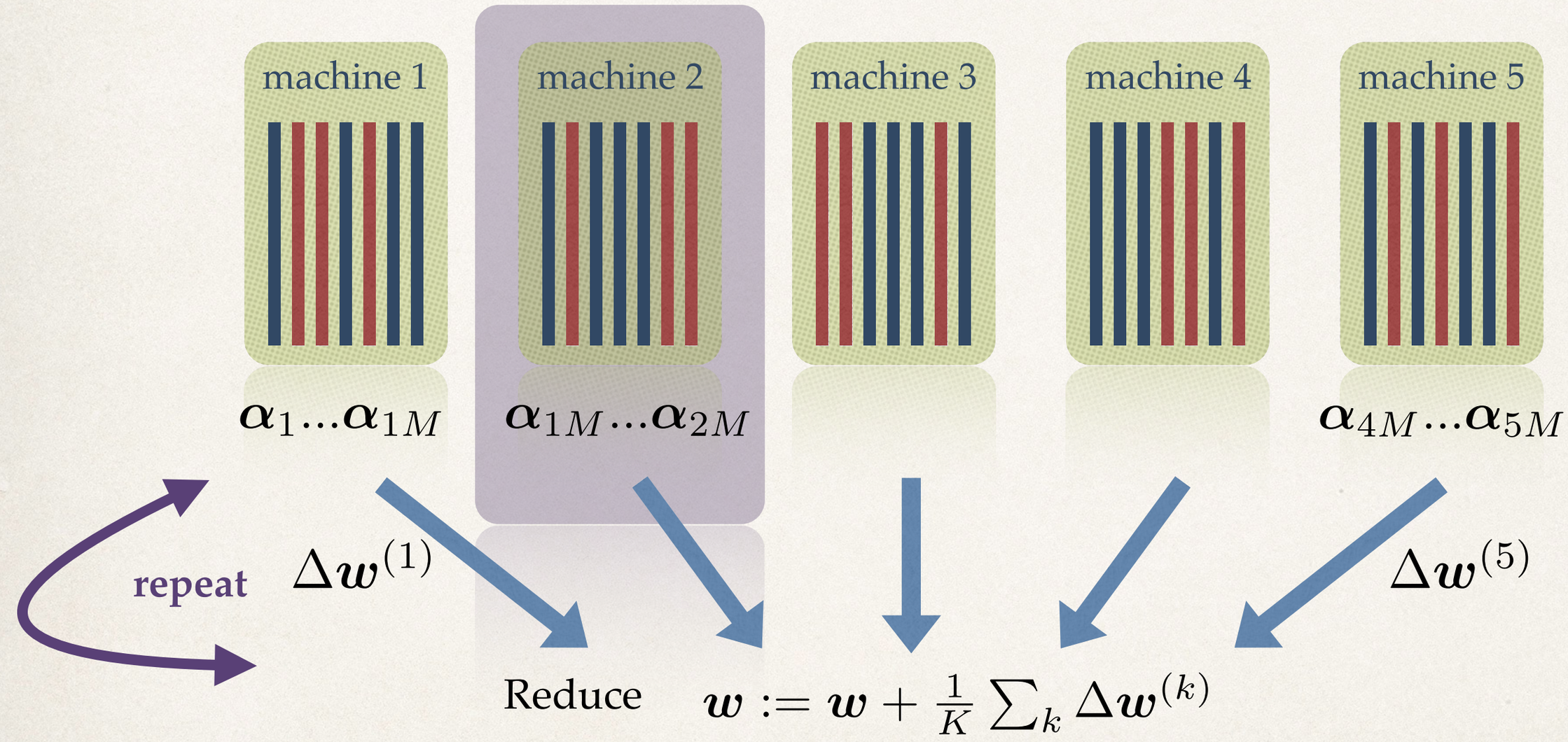
Frank-Wolfe Algorithm



[ICML 2014 Tutorial]

CoCoA

Communication Efficient Distributed Coordinate Ascent



State of the art distributed solver for

SVM, Logistic Regression, Lasso / Sparse Least Squares, Ridge Regression(?)

[NIPS 2014 - CoCoA]

Dissolve *struct*

A Library for Distributed Structured Prediction

built on Spark

Structured SVM solver

Block Coordinate Frank-Wolfe

Distributed Optimization

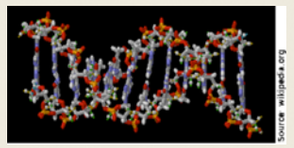
CoCoA

Open Source
Approximate Inference *allowed!*
drop-in replacement for SVM^{struct}

Applications:

Text
- Parsing
- POS tagging, chunking
- sentence alignment
- named entity recognition

Biology
Protein structure & function prediction

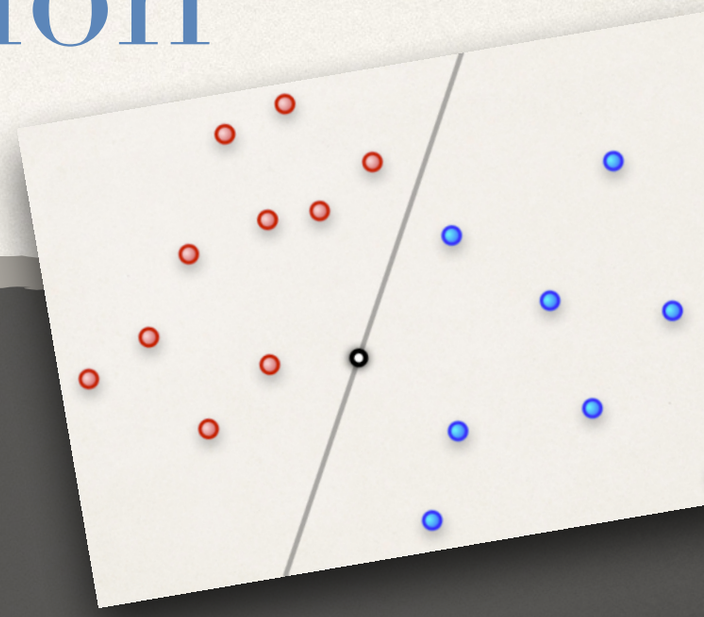


Vision
Horse Segmentation, OCR

more?
- Scene understanding
- object localization & recog.

Your Application?

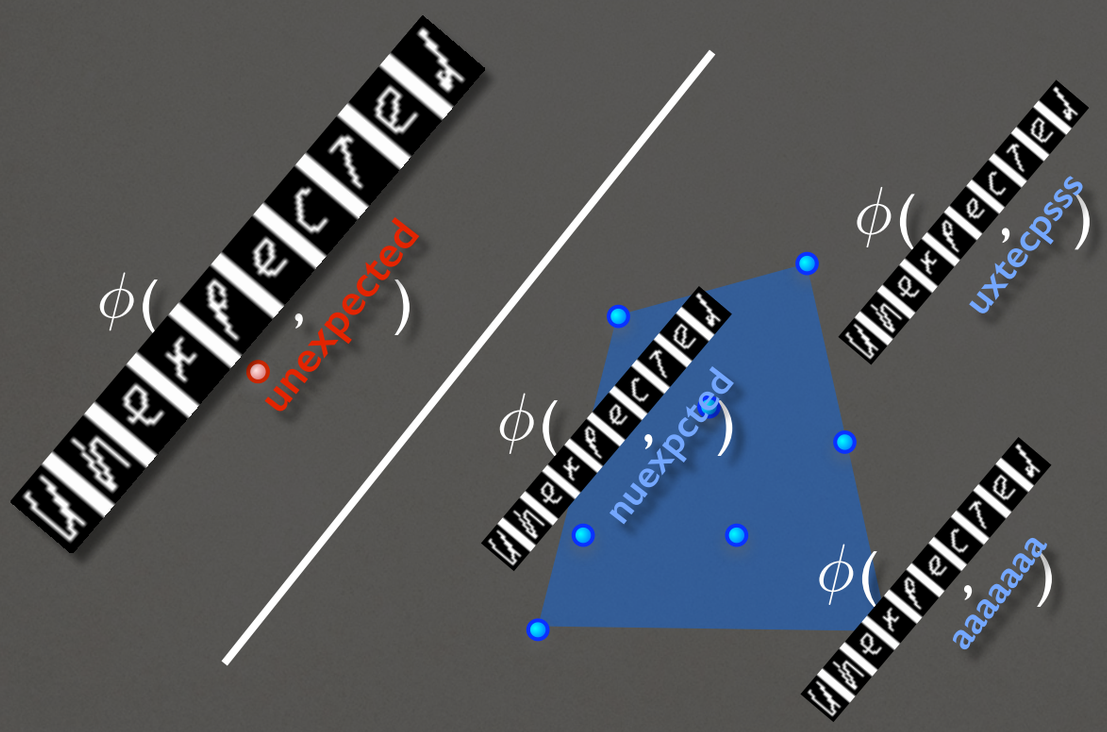
Structured Prediction



joint feature map $\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^d$

large margin separation

$$\langle \boldsymbol{w}, \phi(\boldsymbol{x}_i, \boldsymbol{y}_i) - \phi(\boldsymbol{x}_i, \boldsymbol{y}) \rangle \geq L(\boldsymbol{y}, \boldsymbol{y}_i) - \xi_i \quad \forall \boldsymbol{y}$$



optimization problem:

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \quad \frac{\lambda}{2} \|\boldsymbol{w}\|^2 + \frac{1}{n} \sum_{i=1}^{n} \max_{\boldsymbol{y} \in \mathcal{Y}} \left\{ L(\boldsymbol{y}_i, \boldsymbol{y}) - \langle \boldsymbol{w}, \underbrace{\boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \boldsymbol{\phi}(\boldsymbol{x}_i, \boldsymbol{y})}_{(i,\boldsymbol{y})\text{-th column of } A} \rangle \right\}$$

decoding oracle

A distributed, easy to use solver library for large scale structured SVMs

Apache Spark

Take advantage of lightning-fast in-memory computation, offered by Apache Spark

Primal-Dual

Library is based on the recent communication efficient distributed CoCoA framework, internally using the primal-dual BCFW Solver.

SVM struct compatible

Interface is same as in the widely used SVM struct

Approximate Inference

Compatible with approximate oracles, and automatic caching for efficiency