Phương pháp và công nghệ dữ liệu Lớn

Machine learning for Big Data

Phùng Quốc Định
Centre for Pattern Recognition and Data Analytics
Deakin University, Australia
Email: dinh.phung@deakin.edu.au
(published under **Dinh Phung**)

Outline

- Mô hình lớn cho dữ liệu lớn
  - Why traditional methods might fail on big data?
  - Three approaches to scale up big models
    - Scale up Gradient Descent Methods
    - Scale up Probabilistic Inference
    - Scale up Model Evaluation (bootstrap)
- Ba cách tiếp cận để phát triển mô hình lớn
  - Data augmentation
  - Stochastic Variational Inference for Graphical Models
  - Stochastic Gradient Descent and Online Learning
- Công nghệ nào cho mô hình lớn?
  - Hadoop, Spark, TensorFlow

Tiếp cận Dữ liệu lớn

- Technical approach a big data problem

AI SYSTEMS AND APPLICATIONS

Deployment into applications and products

TECHNOLOGY for DATA PROCESSING

Pick a suitable processing technology or platform

CLOUD COMPUTING INFRASTRUCTURE

In some cases, one can decide to build their own

Mô hình lớn cho dữ liệu lớn

Scale up Machine Learning and Statistical Models

- Mô hình lớn cho dữ liệu lớn
  - Three approaches to scale up big models
    - Scale up Gradient Descent Methods
    - Scale up Probabilistic Inference
    - Scale up Model Evaluation (bootstrap)
- Ba cách tiếp cận để phát triển mô hình lớn
  - Data augmentation
  - Stochastic Variational Inference for Graphical Models
  - Stochastic Gradient Descent and Online Learning
- Công nghệ nào cho mô hình lớn?
  - Hadoop, Spark, TensorFlow

Scaling up ML and Stats models

Mô hình tính toán cho dữ liệu lớn

- Tại sao nhiều phương pháp truyền thống trong ML và Stats có thể không dùng được trong bài toán DLL?
  - Massive data challenge
  - Massive model size challenge
  - Theoretical challenge: classic ML models unable to cope

COLLABORATIVE FILTERING

Movie and product recommendation

billions model parameters

READMISSION PREDICTION

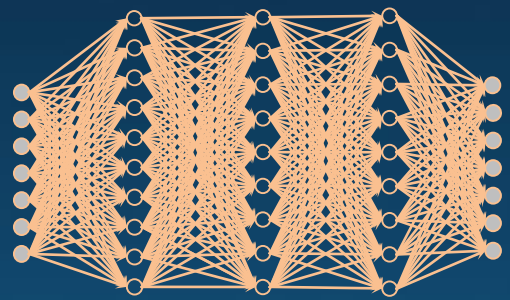
Analyzing Electronic Medical Records and Genomics data

billions model parameters

GOOGLE BRAIN

Deep learning for images

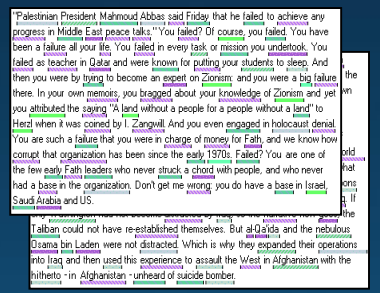
billions model parameters



BAYESIAN TOPIC MODELS

Document analysis and summarization

trillion model parameters

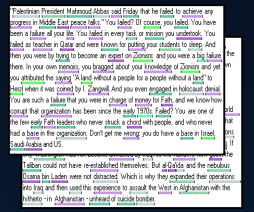


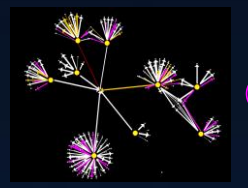
Scaling up ML and Stats models

[Xing and Qirong, Tuts KDD'15]

- Modern ML/Stats algorithms vs traditional program
  - Traditional programs are transaction-centric, require individual correctness at every step.
  - ML algorithms are optimization-centric and iterative convergent algorithms.
    - Error tolerance: robust against small variations/errors during intermediate steps.
    - Structural dependency: changing model parameterization and optimization strategy affects efficient parallelization and scalability.
    - Non-uniform convergence: optimal solutions might converge very differently at different number of steps.

Scale up ML/Stats models





∞

Feature extraction from data

$\mathcal{D} = \{x_i, y_i\}, i = 1, \ldots . n$

Data Manipulation

Analytics

Communication & Visualization

Refinement

Data Acquisition/Raw Data

Data Storage

Data Cleaning

Exploratory Data Analysis

Finding Dependency and Relationship

Confirmatory Data Analysis

Advanced Machine learning Algorithms
Statistical Models

Bài toán tối ưu

Tham số

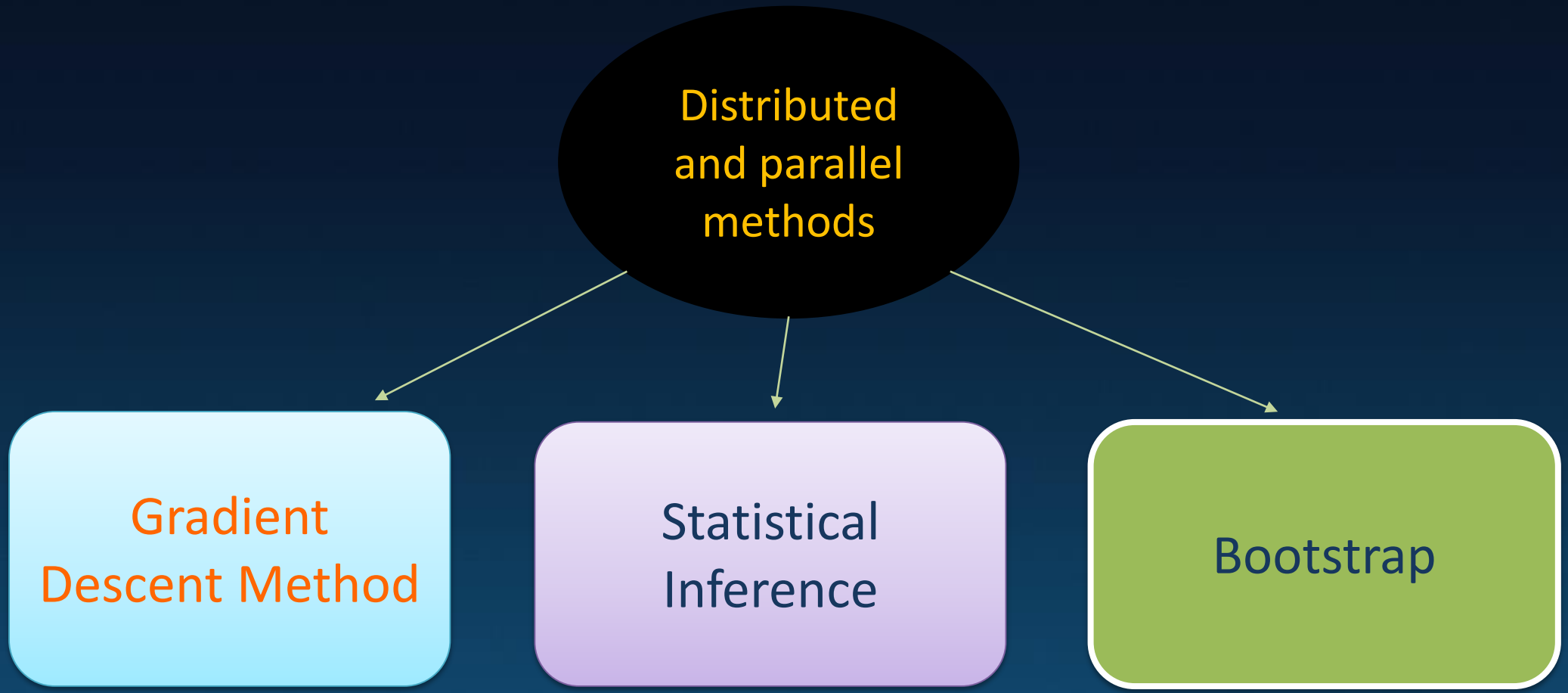
Dữ liệu vào

$$\max J(\theta) = \mathcal{L}(D;\theta) + \lambda R(\theta)$$

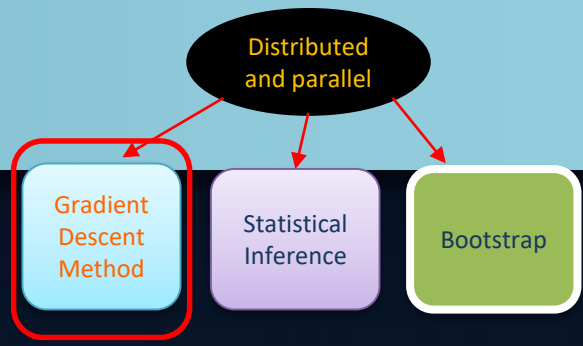
Mô hình

Scale up ML/Stats

- Three approaches to scale up ML/Stats models



Scale up Gradient Descent optimization



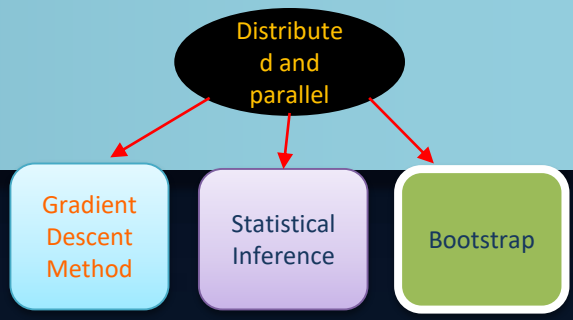
- Distributed/Parallel Gradient Methods
  - Many machine learning methods reduce to minimizing some objective function.
    - Linear regression: $J(\theta) = \frac{1}{2}\sum_i [\theta^{\mathrm{T}} x_i - y_i]^2$
    - Logistic regression: $J(\theta) = \sum_i \log(1 + e^{-y_i \theta^{\mathrm{T}} x_i})$
  - Almost all objectives are optimized with Gradient Descent (GD) update

$$\theta \leftarrow \theta - \eta \sum_{i=1}^{n} g(\theta, x_i, y_i)$$

Iterative-convergent

    - Linear regression: $g = (y_i - \theta^{\mathrm{T}} x_i) x_i$
    - Logistic regression: $g = y_i \left(1 + e^{-y_i \theta^{\mathrm{T}} x_i}\right)^{-1} x_i$

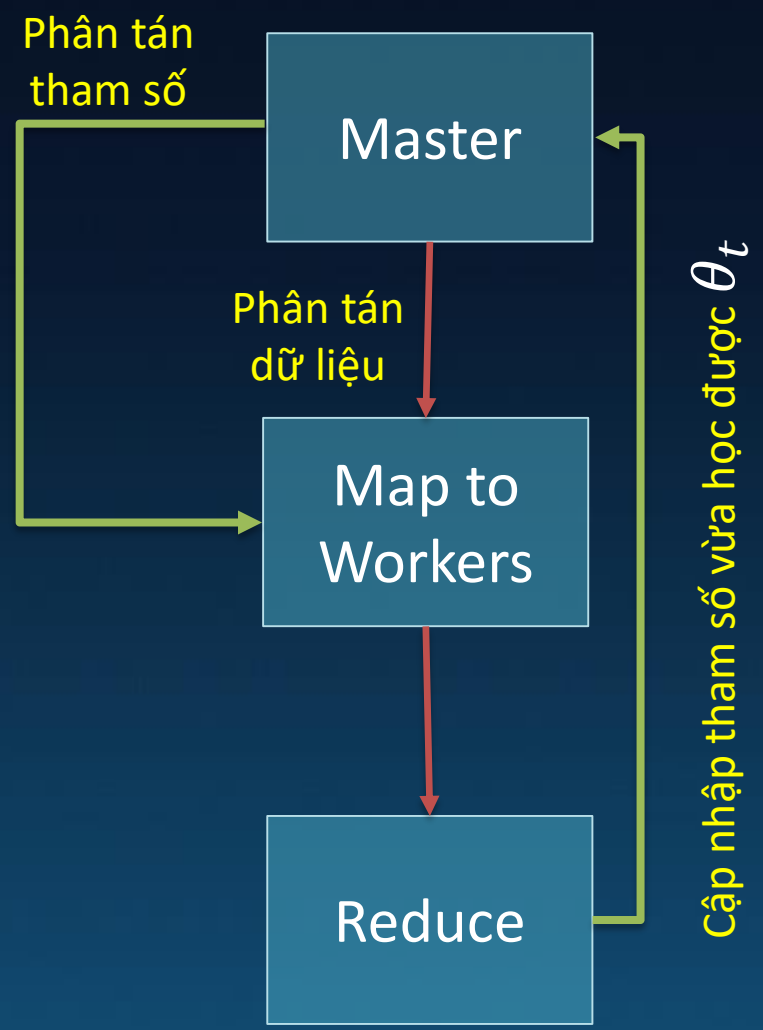
Scale up Gradient Descent optimization



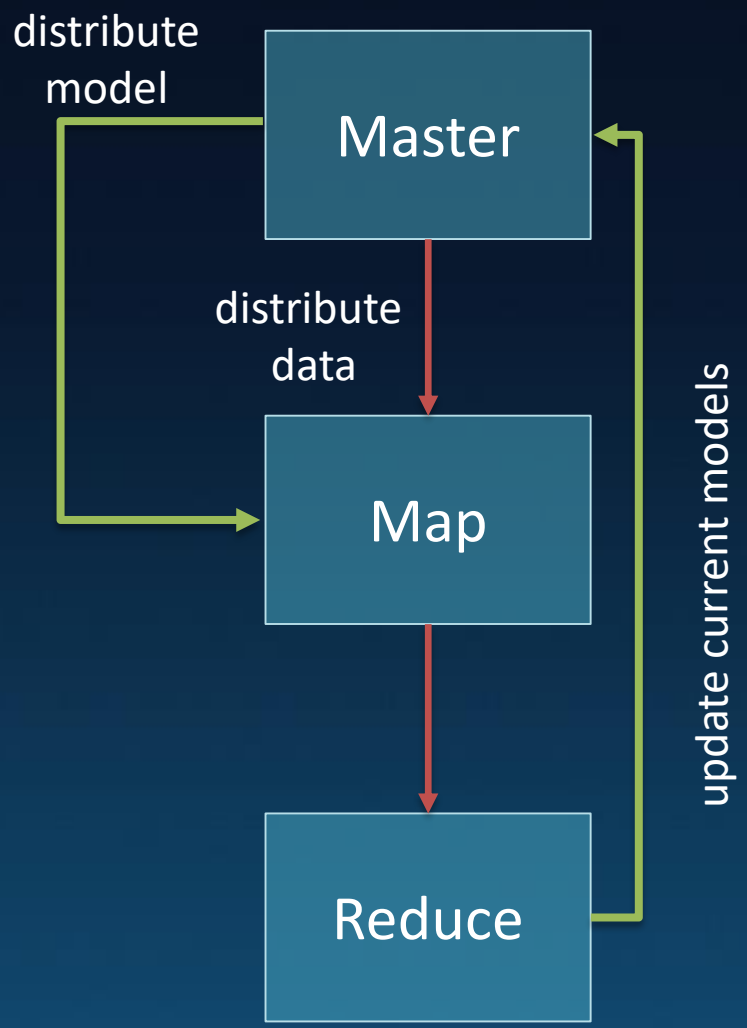
- Gradient Descent-based ML models
  - Regression (linear, ridge, Lasso, …)
  - Kernel methods (SVM, SVDD, ..)
  - Ensemble models (boosted gradient tree, …)
  - Almost all deep learning methods
    - Deep neural nets
    - Neural embedding: word2vec, node2vec, etc
    - Deep autoencorder: denoising autoencorder
    - Deep generative models
      - Variational Autoencoder (VAE)
      - Generative Adversarial Networks (GAN)

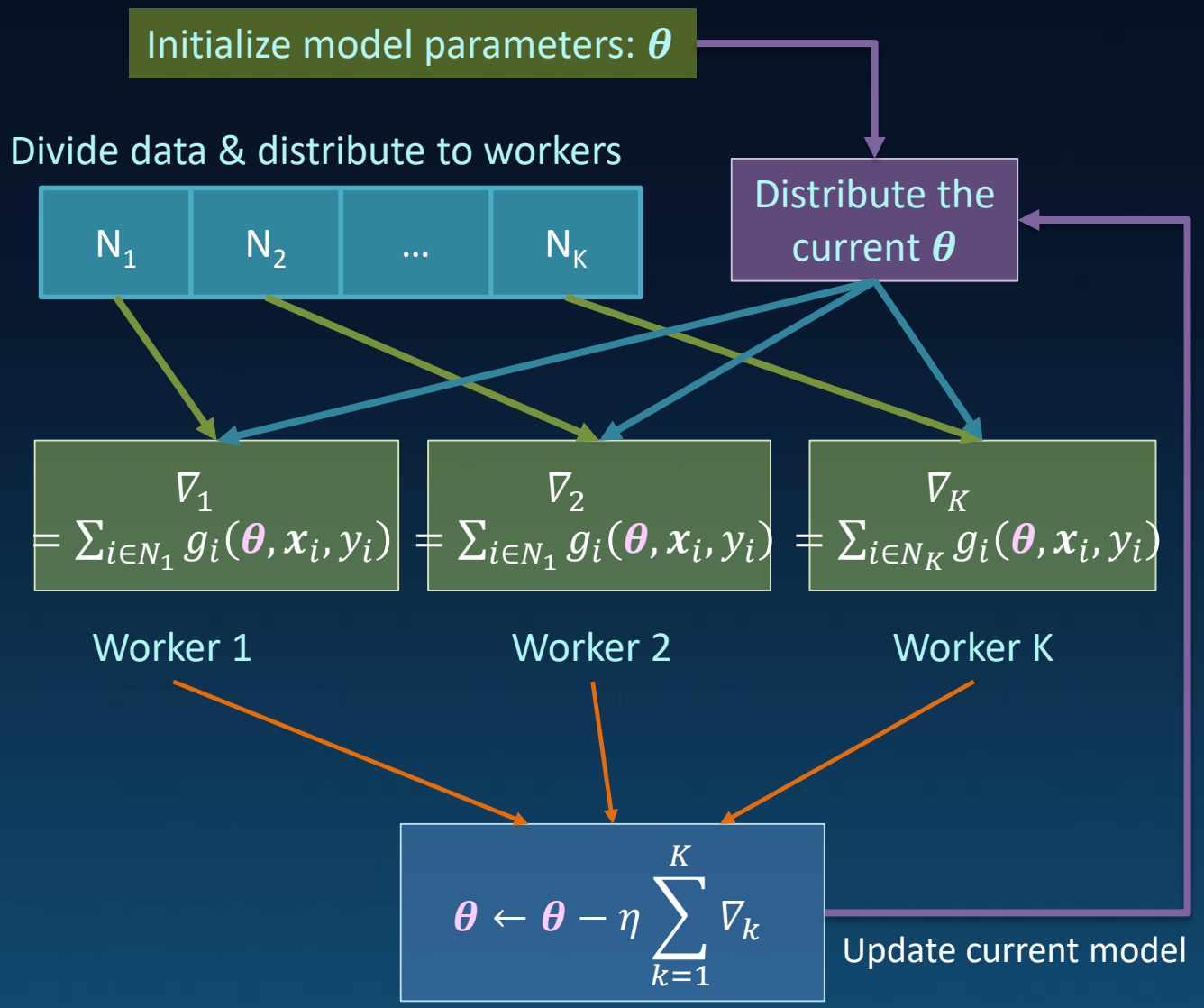
Scale up Gradient Descent optimization

- Distributed Gradient Methods
  - While not converged
    - Master: distribute model parameters to workers.
    - Master: evoke map-reduce jobs
      - Many mappers compute gradient w.r.t chunks of datasets given.
      - Single reducer to combine full gradient.
  - EndWhile



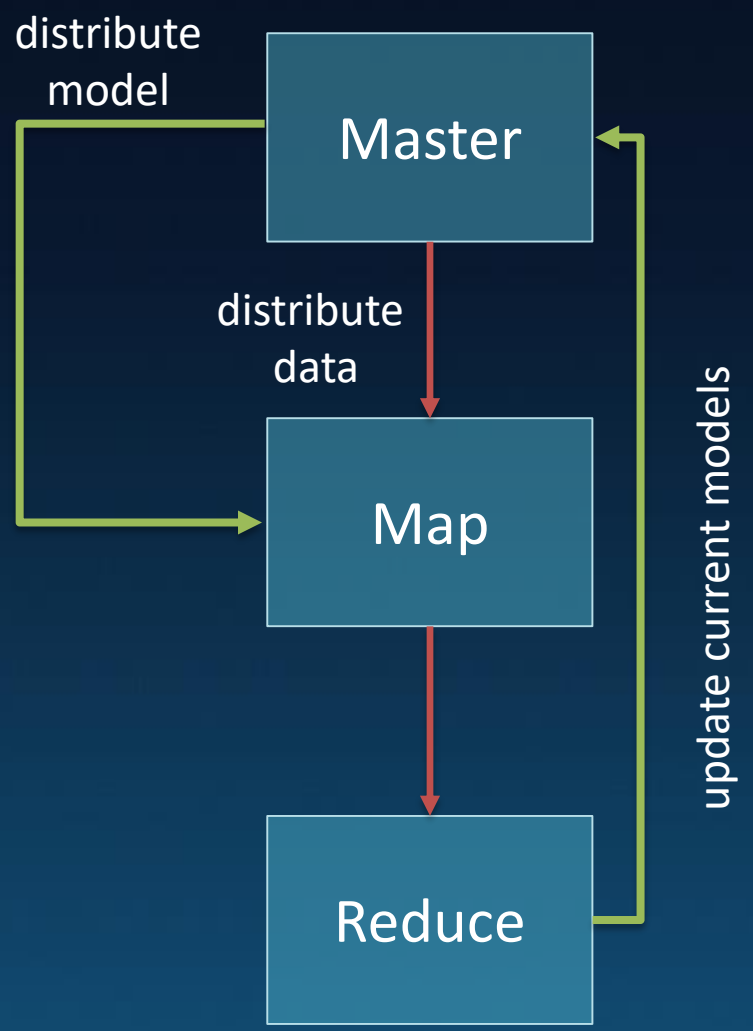
Scale up Gradient Descent optimization





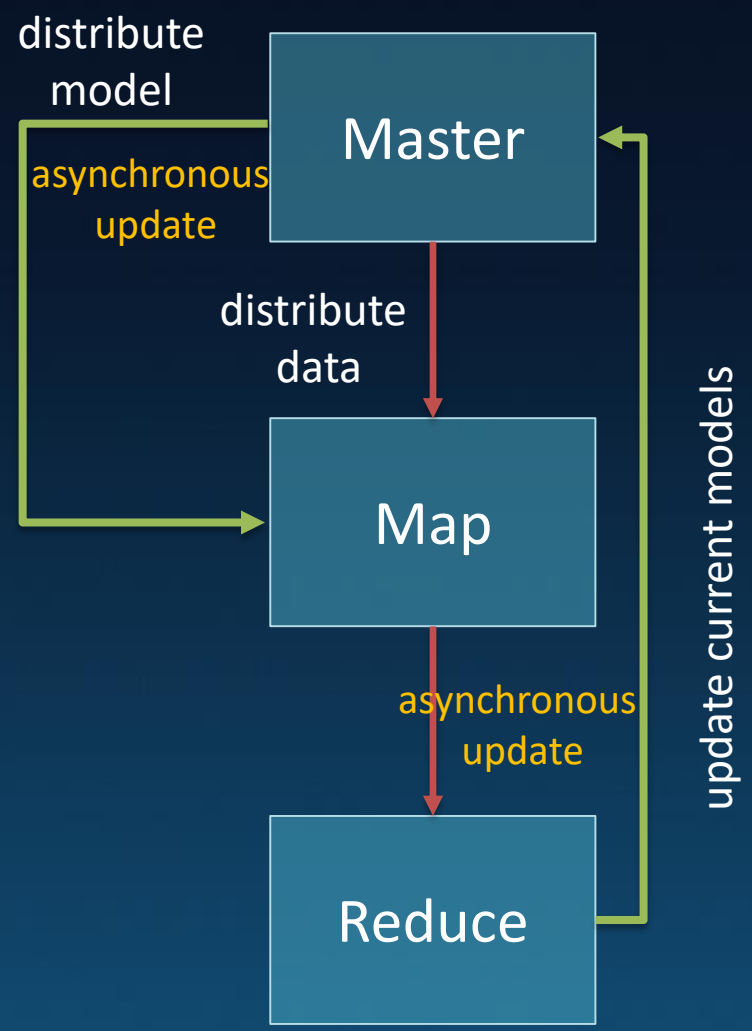
Scale up Gradient Descent optimization

- Distributed Gradient Methods
- Ưu điểm:
  - Consistency tuyệt đối
  - Hội tụ đến lời giải tốt ưu (nếu lặp đủ)
  - Tốc độ hội tụ lý thuyết nhanh (tuyến tính)
- Nhược điểm:
  - Đòi hỏi phải có sự đồng bộ tại phía máy chủ chứa tham số, điều này khiến cho máy chủ đó phải đợi tất cả các workers tính toán xong.



Scale up Gradient Descent optimization

- Vậy làm thế nào cân bằng giữa: tính chính xác của lời giải và giảm sự đồng bộ?
- Vì sao? Giảm yêu cầu đồng bộ sẽ làm quá trình tính toán nhanh hơn.
- Chiến lược đơn giản nhất:
  - sau khi mỗi worker tính toán gradient xong thì gửi ngay lập tức về server.
  - server sẽ cập nhật tham số ngay lập tức và trả lại tham số cho worker đó.
- Chiến lược này có thể cho kết quả tốt trong một số trường hợp nhưng <u>không đảm bảo về mặt lý thuyết</u>.



Scale up Gradient Descent optimization

HOGWILD!: A Lock-Free Approach to Parallelizing SGD

[Recht et. al. NIPS's 11]

- Có rất nhiều bài toán tối ưu trong KHDL có thể biểu diễn lại dưới dạng như sau:

$$\min_x f(x) \triangleq \sum_{e \in E} f_e(x_e) \text{ where } f: \mathbb{R}^n \to \mathbb{R} \text{ and } E \subseteq 2^{\{1,\dots,n\}}.$$

  - $|E|$ is very large and $|e|$ is small for every e

- Ví dụ như những mô hình và bài toán dưới đây:
  - Sparse SVM
  - Matrix Completion
  - Graph Cuts

Scale up Gradient Descent optimization

Ví dụ: Sparse SVM

- Given a sparse training set
  - $D = \{(\mathbf{z}_1, y_1), \ldots, (\mathbf{z}_N, y_N)\}$ where data item $\mathbf{z}_i$ is sparse.
- We need to fit this data set using SVM
  - $\min_x \left(\sum_{e \in E} \max(0, 1 - y_e x^{\mathrm{T}} \mathbf{z}_e) + \frac{\lambda}{2} \|x\|^2 \right)$ where $E = D$
- Let α_e denote the non-zero components in z_e and let d_j denote the number of training examples which are non-zero in component $j (j = 1, 2, \ldots, d)$

$$\min_x \left(\sum_{e \in E} \left(\max(0, 1 - y_e {x_{\alpha_e}}^{\mathrm{T}} \mathbf{z}_{\boldsymbol{\alpha_e}}) + \frac{\lambda}{2} \sum_{j \in \alpha_e} \frac{x_j^2}{d_j} \right) \right)$$

Scale up Gradient Descent optimization

HOGWILD!: A Lock-Free Approach to Parallelizing SGD

Algorithm

for *t=1,2,…* **do**
  sample e uniformly at random from E
  read current state x_e *and evaluate* $G_e(x_e)$
  for $v \in e$ *do* $x_v = x_v - \gamma G_{ev}(x_e)$
endfor
where $G_e(x_e) = |E| \nabla f_e(x_e)$

$$\min_x f(x) \triangleq \sum_{e \in E} f_e(x_e)$$

- If the lag between when the gradient is computed and when it is used is less than a finite τ, the HOGWILD! Algorithm is proven to converge to the real optimal solution.
- This algorithm is ideal for **paralleling in a single machine**, but less ideal for **distributing on many machines**.

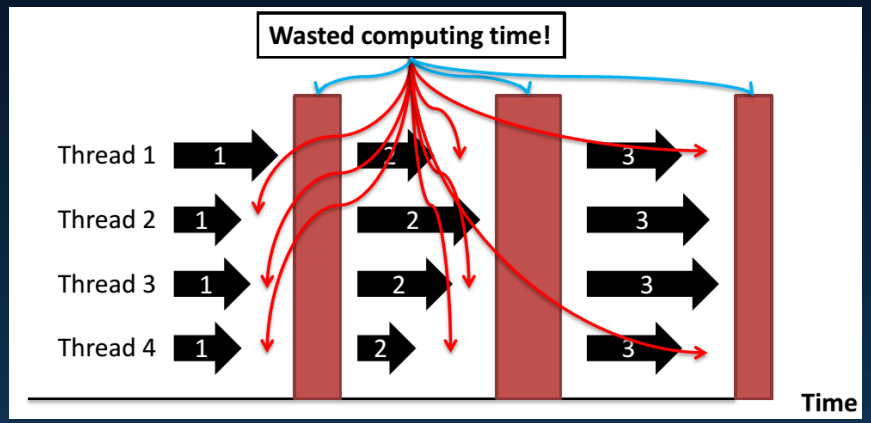
Đồng bộ hay không đồng bộ?

Synchronous vs Asynchronous

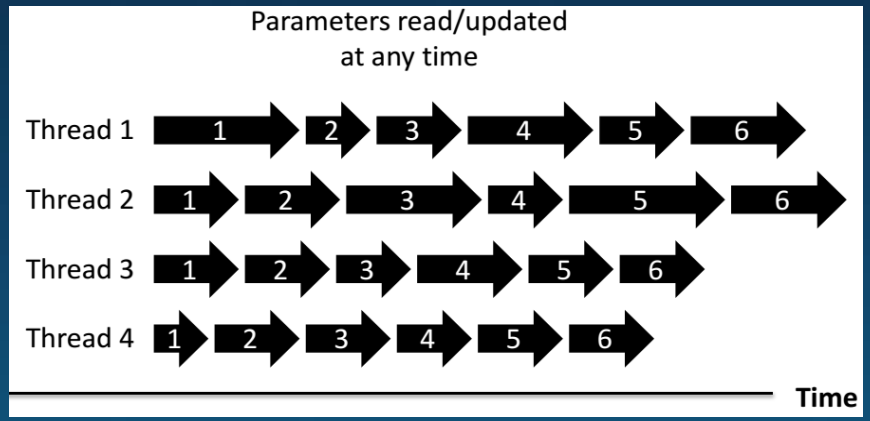
[Xing and Qirong, KDD Tuts'15]

- The fastest thread must wait for the slowest one.
- Model parameter is guaranteed to converge to optimal solution.

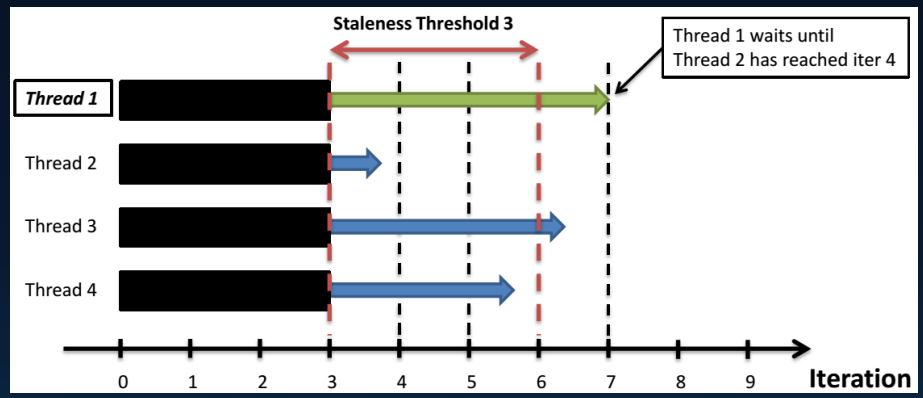
Bulk Synchronous Parallel (BSP)



Asynchronous



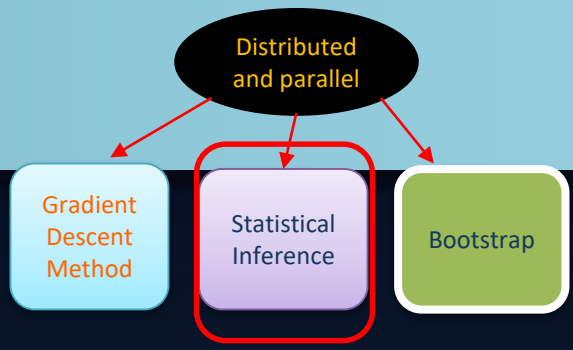
Stale Synchronous Parallel [Ho et. al. 2013, Dai et al 2015]



- Key idea: the time gap between fastest and slowest thread is bounded by τ.
- Theoretical guarantee is given in terms of τ.

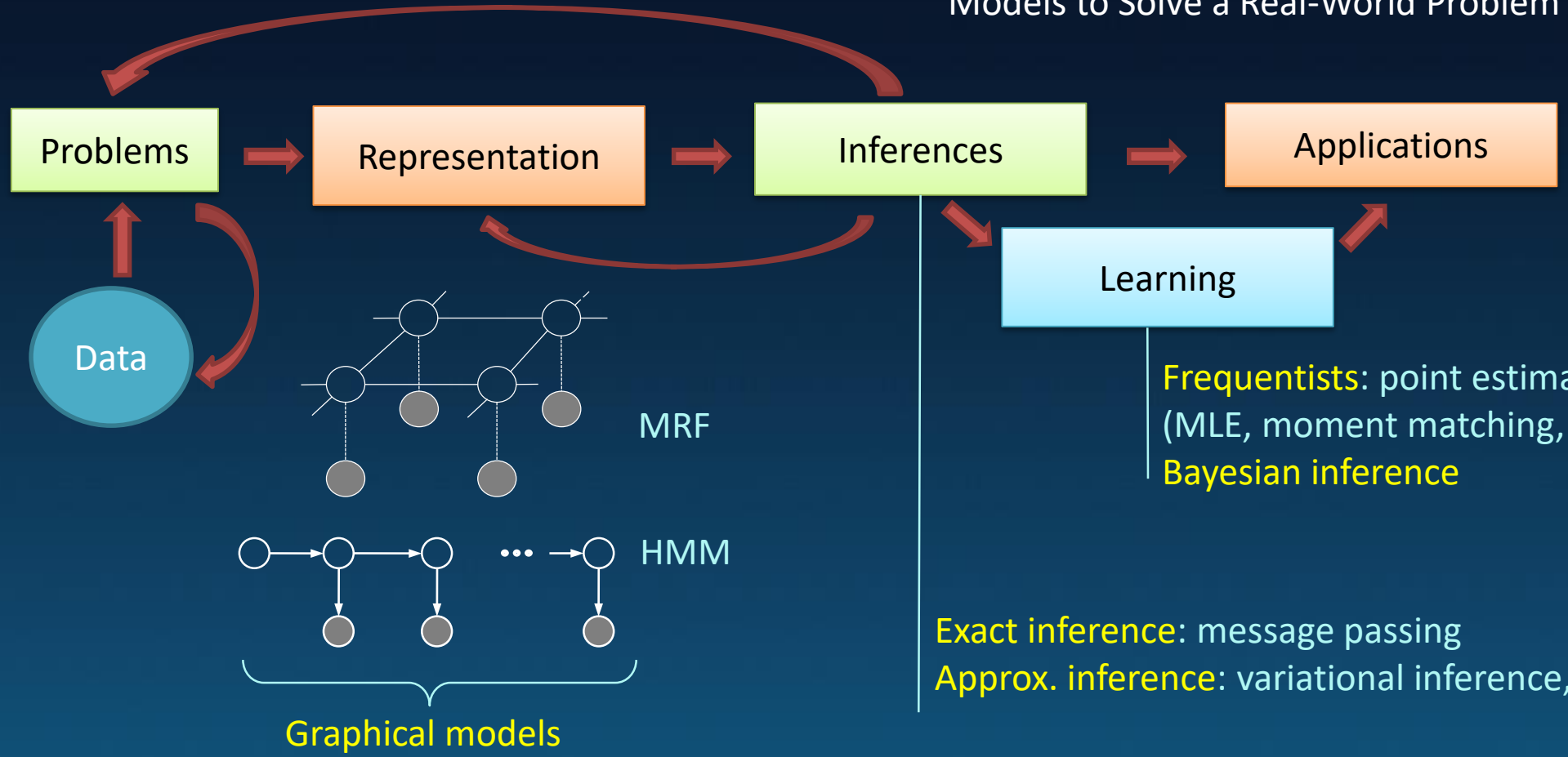
- Model parameter is updated/read at any time whenever a thread finishes its workload.
- Model parameter might be drifted from optimal solution.

Scale up Statistical Inference



- Distributed Statistical Inference

A Typical Life Cycle for Applying Graphical Models to Solve a Real-World Problem



Scale up Statistical Inference

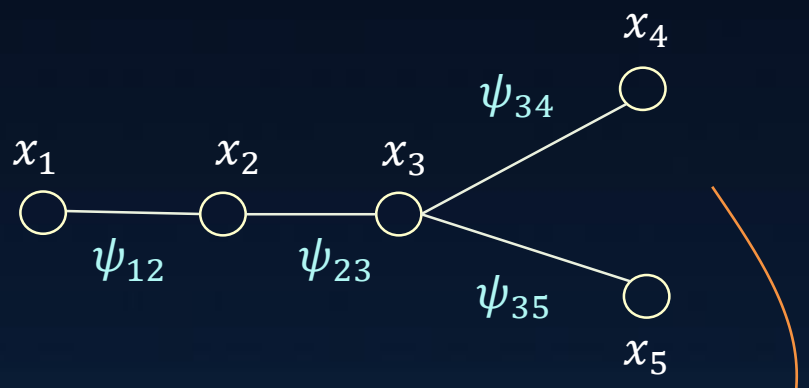
- Graphical models
  - Express a joint distribution over massive number of random variables.
  - Conditional distributions respect graph structure $G = (V, E)$ with N vertices
  - $P(x_1, \dots, x_n) = \frac{1}{Z} \prod_c \psi_c(x_c)$
- Inference = Belief Propagation

$$p(\boldsymbol{x}) = \frac{1}{Z}\psi_{12}\psi_{23}\psi_{34}\psi_{35} \quad Z = \sum_{x_1,x_2,x_3,x_4,x_5} \psi_{12}\psi_{23}\psi_{34}\psi_{35}$$

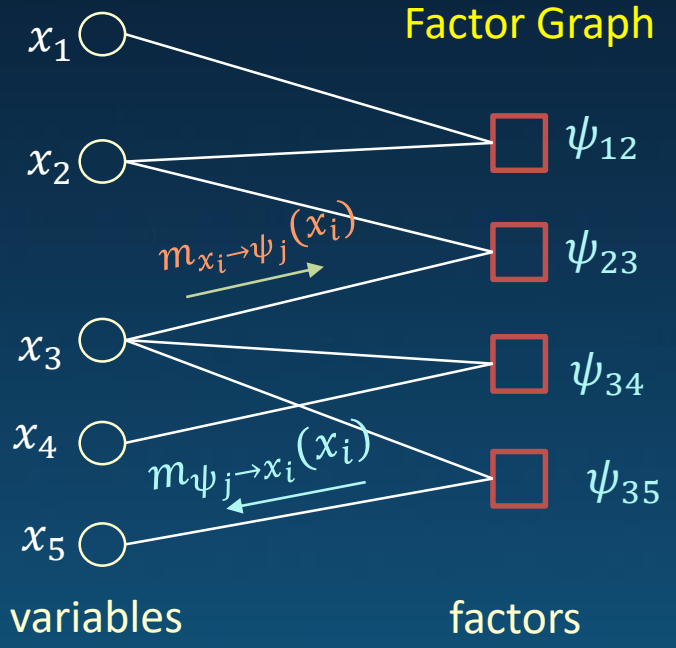
$$m_{x_i \to \psi_j}(x_i) \propto \prod_{k \in \mathcal{N}_i \setminus j} m_{\psi_k \to x_i}(x_i)$$ variable to factor

$$m_{\psi_j \to x_i}(x_i) \propto \sum_{\text{all except } x_i} \psi_j(x_j) \prod_{k \in \mathcal{N}_j \setminus i} m_{x_k \to \psi_j}(x_k)$$

factor to variable



$$\psi_{ij} = \exp\{\theta_i(x_i)\theta_j(x_j) + \lambda_{ij}(x_i, x_j)\}$$

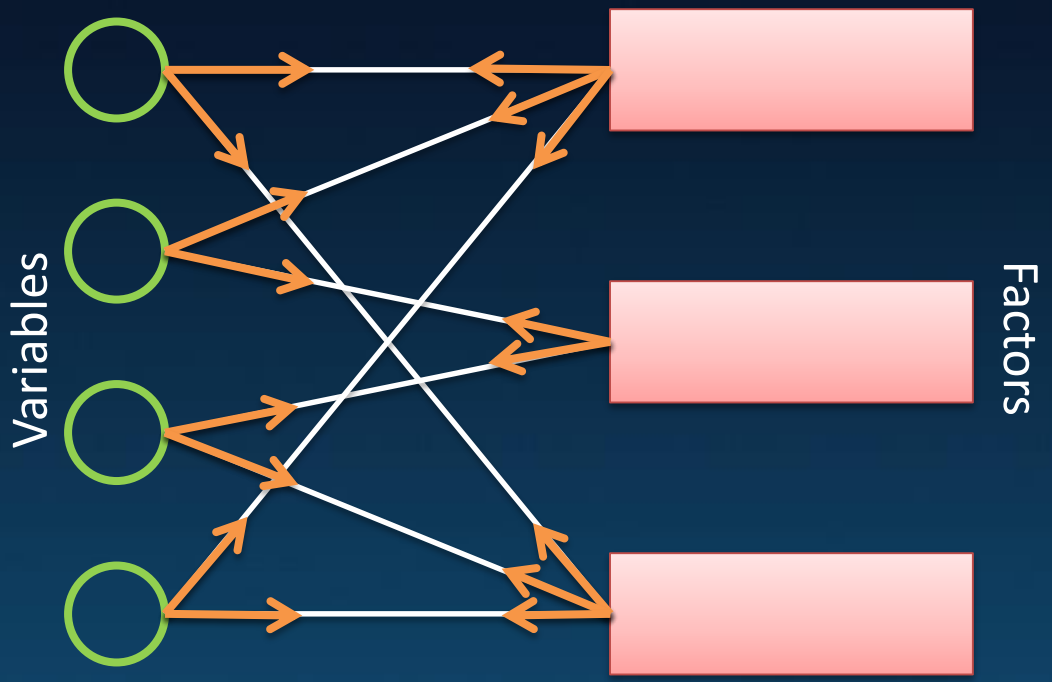


Belief Propagation

[Slides from Joseph Gonzalez et al. 09]

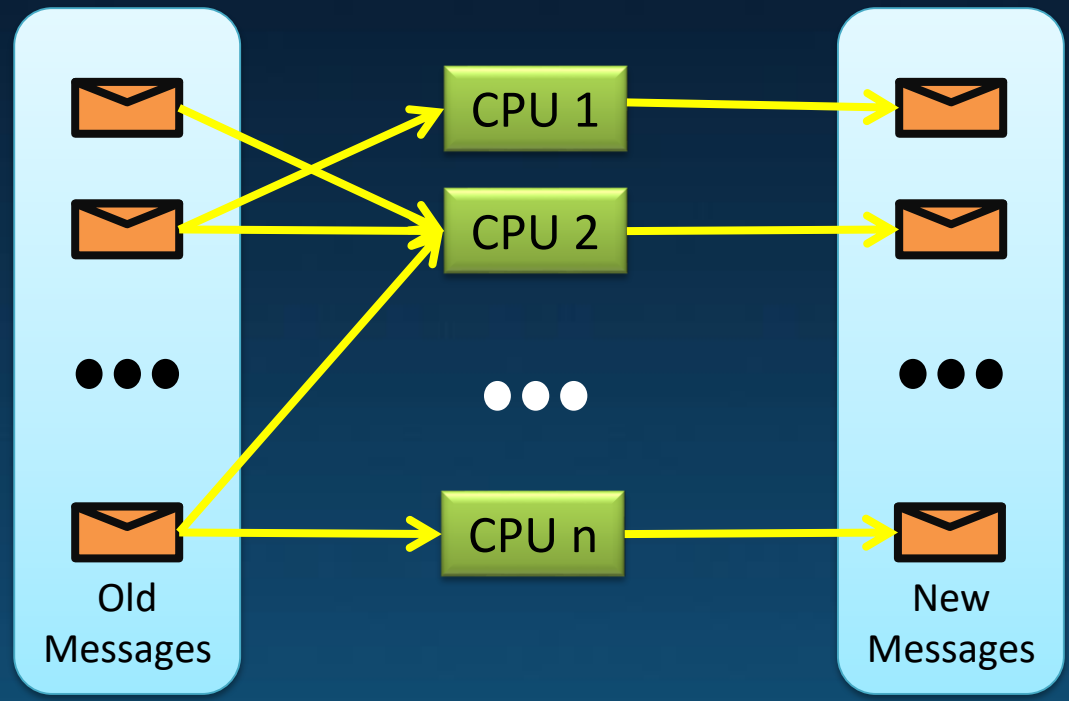
Belief Propagation (BP)

- Iterative message passing



Parallel Synchronous BP

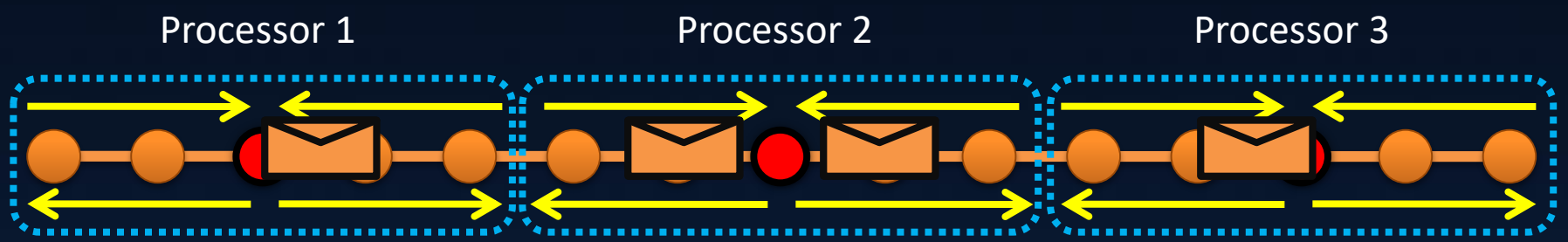
- Given all messages from previous iteration, all messages in current iteration can be computed simultaneously and independently



Joseph Gonzalez et al., Parallel Splash Belief Propagation, JMLR 2009

ChainSplash Parallel BP

[Slides from Joseph Gonzalez et al. 09]



- ChainSplash parallel BP is a parallel **approximate** inference algorithm.
- Given a chain graph, the procedure is:
  1. Divide the chain evenly among processors
  2. Loop in parallel on all processors until converged:
     a. On each processor, run sequential Forward – Backward
     b. On each processor, exchange messages with its 2 neighbours

Theorem: Using *p* processors this algorithm achieves a τ_ϵ approximation in time:

$$O\left(\frac{n}{p} + \tau_\epsilon\right)$$

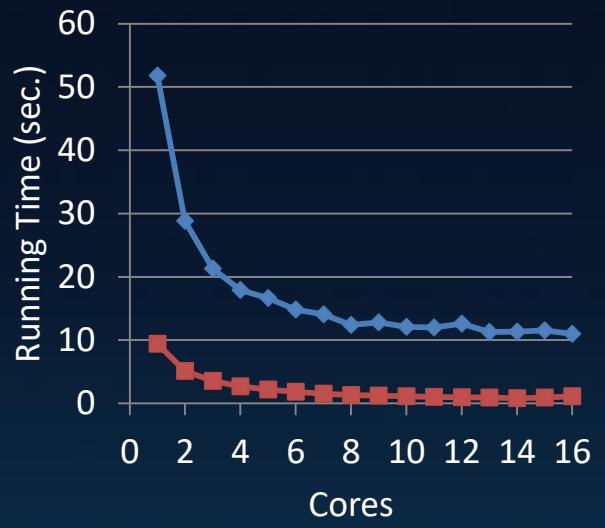
The Splash Belief Propagation algorithm

Splash Belief Propagation

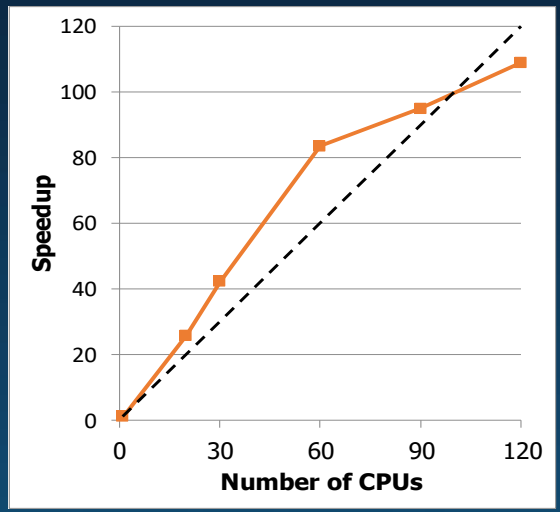
=

Splash operation + dynamic Splash scheduling

Synchronous BP (blue) vs Splash BP (red) on shared memory system with a small graph



Splash BP (orange) speedup performance on distributed system with a large graph

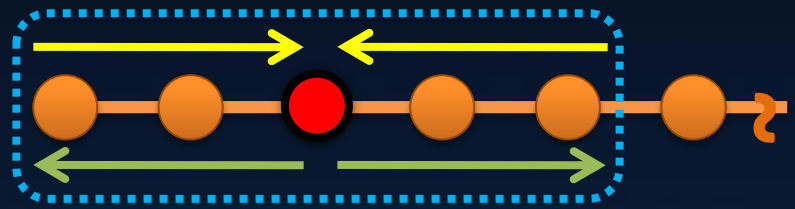


[Nguồn: Joseph Gonzalez et al., 2009]

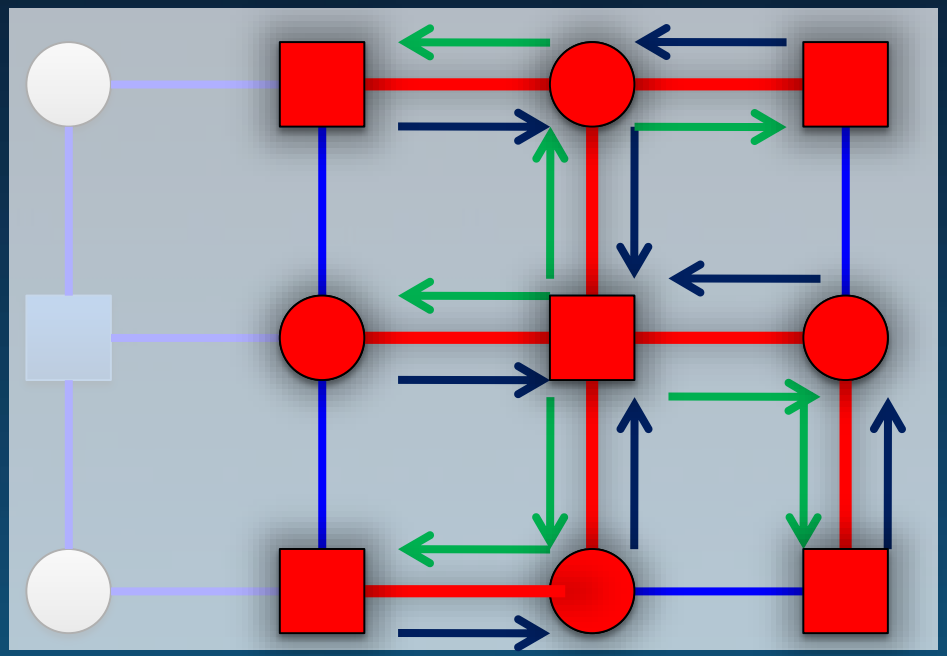
The Splash Operation

[Slides from Joseph Gonzalez et al. 09]

- Generalize the optimal chain algorithm:



to arbitrary cyclic graphs:



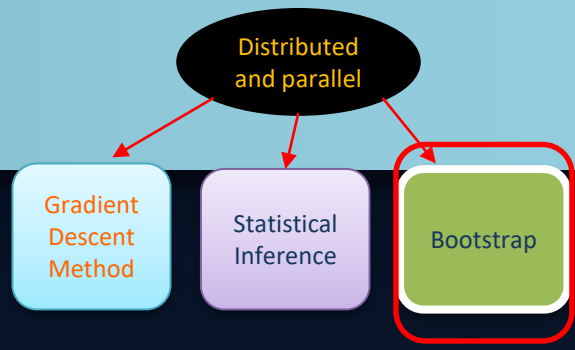
- Grow a BFS Spanning tree with fixed size.
- Forward Pass computing all messages at each vertex.
- Backward Pass computing all messages at each vertex.

Scale up Statistical Inference

- Một số cách tiếp cận khác
  - MCMC inference for graphical models
    - Naïve approach: take multiple chains in parallel, take average/consensus between chains.
  - Embarrassingly Parallel (but correct) MCMC [Neiswanger et al, 14]
  - Parallel Gibbs sampling
    - Approximated distributed LDA [Newman et al, 2009]
    - Asynchronous version [Ahmed et al. 2012, Dai et al. 2015]
    - …. see Eric Xing's KDD 2015 tutorials for more references.
  - Variational inference for graphical models
    - Stochastic Variational Inference (SVI) [Hoftman et al., JMLR13, Huynh et al, UAI16]
    - Deterministic Wasserstein-based Optimization [Nguyen, Annals of Stats, 13]

Scale up Model Evaluation

Bootstrap for massive data [Kleiner, Jordan 12]



- It is an important theoretical aspect to assess the quality of inference.

Data $X_1, X_2, \dots, X_n \sim f_\theta$ ⟶ Estimate $\hat{\theta}$ ⟶ Quality $|\theta - \hat{\theta}|$?

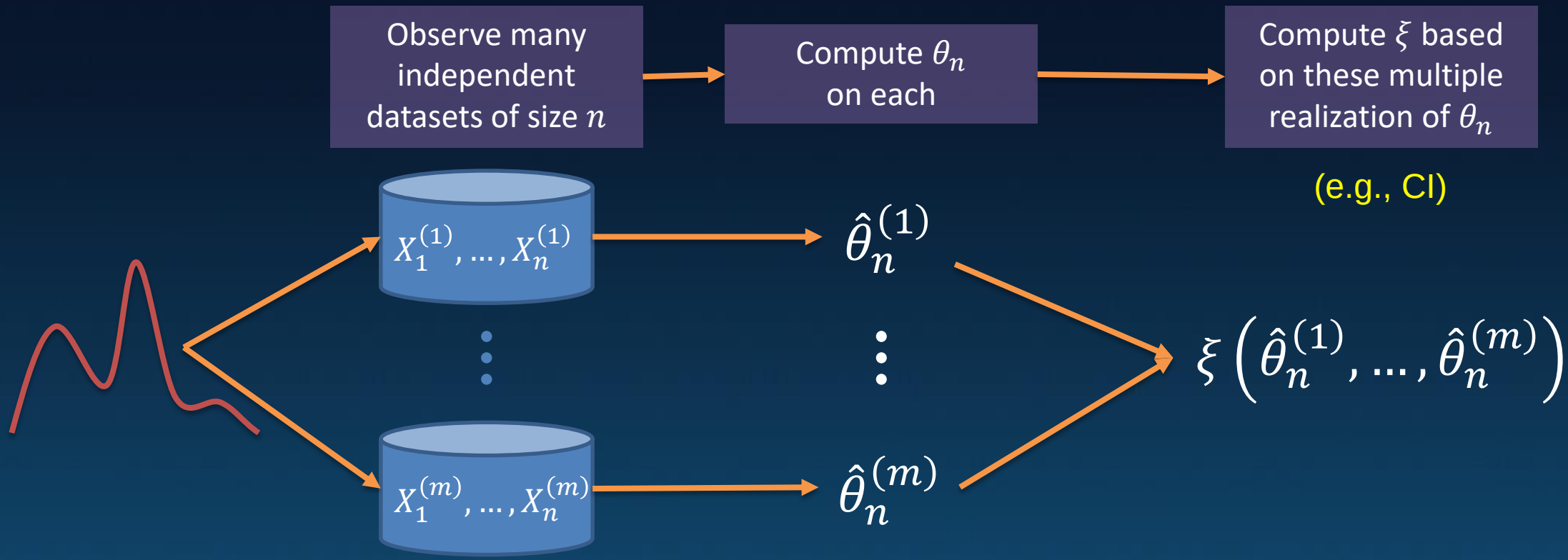
- Bootstrap: generic framework for computing confidence interval
  - it's useful in ML/Stats to assess the quality of the estimates (e.g., model parameters) in terms of confidence interval, error bars.
  - Model selection, smoothing, bias correction, …
  - Provide more useful information than a simple point estimate.
- Can it be used on big data?

Scale up Model Evaluation

Bootstrap for massive data

[Kleiner, Jordan 12]

- Trường hợp lý tưởng:



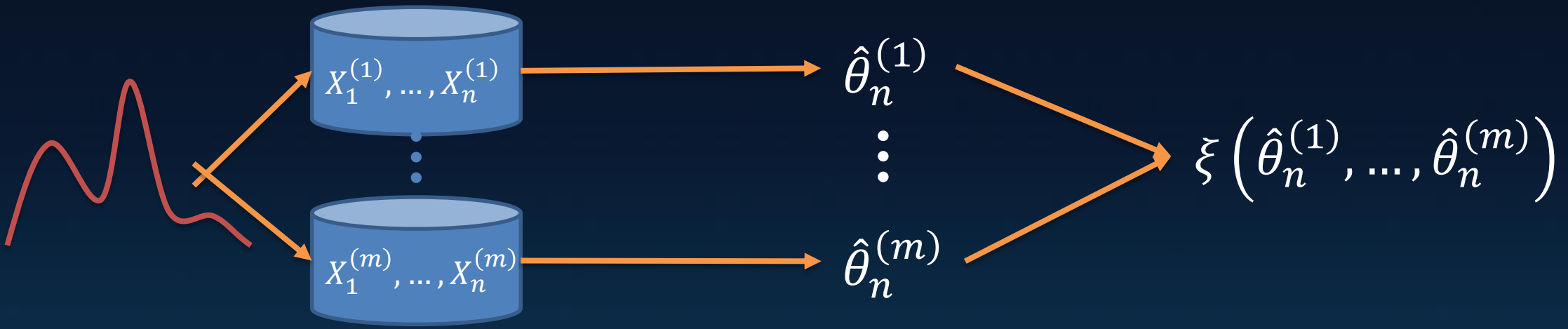
Bu, we only observe one dataset of size n!

Scale up Model Evaluation

Bootstrap for massive data

[Kleiner, Jordan 12]

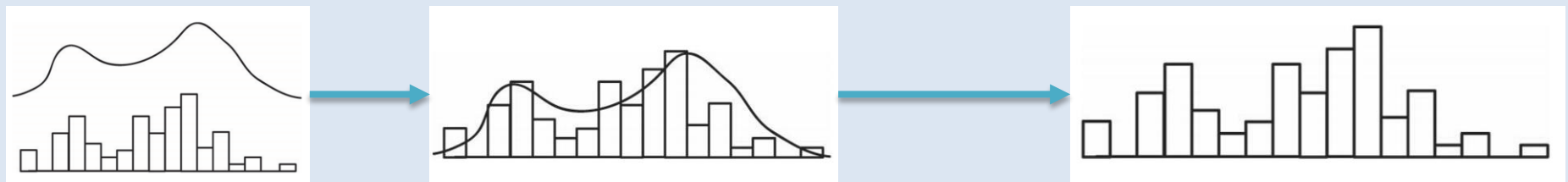
- **Bootstrap (Efron, 1979)**
  - Use the observed data to simulate datasets of size n



Repeatedly ***resample*** n points with replacement from the original datasets of size n

→ Compute θ_n^* on each ***resample***

→ Compute ξ based on these multiple realization of θ_n^* as our estimate of ξ for θ_n



Scale up Model Evaluation

Bootstrap for massive data

[Slide from Jordan, ICML14]

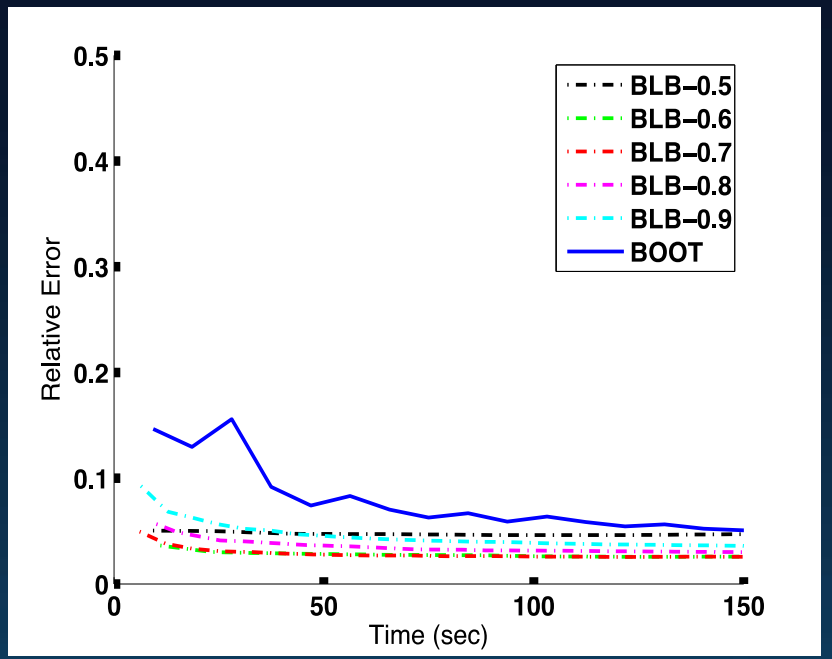
- Computational bottleneck with bootstrap
  - Seemingly a wonderful match to modern parallel and distributed computing platforms
  - But the expected number of distinct points in a bootstrap resample is $\sim 0.632n$
    - E.g., if original dataset has size 1TB, then expect resample to have size $\sim 632GB$
  - Can't feasibly send resampled datasets of this size to distributed servers.
  - Even if one could, can't compute the estimate locally on datasets this large.

Scale up Model Evaluation

Bootstrap for massive data

[Kleiner, Jordan 12]

- The Bag of Little Bootstrap (BLB)
  - The subsample contains only b points, and so the resulting empirical distribution has its support on b points
  - But we can (and should!) resample it with replacement n times, not b times
  - Doing this repeatedly for a given subsample gives bootstrap confidence intervals on the right scale---no analytical rescaling is necessary!
  - Now do this (in parallel) for multiple subsamples and combine the results (e.g., by averaging)



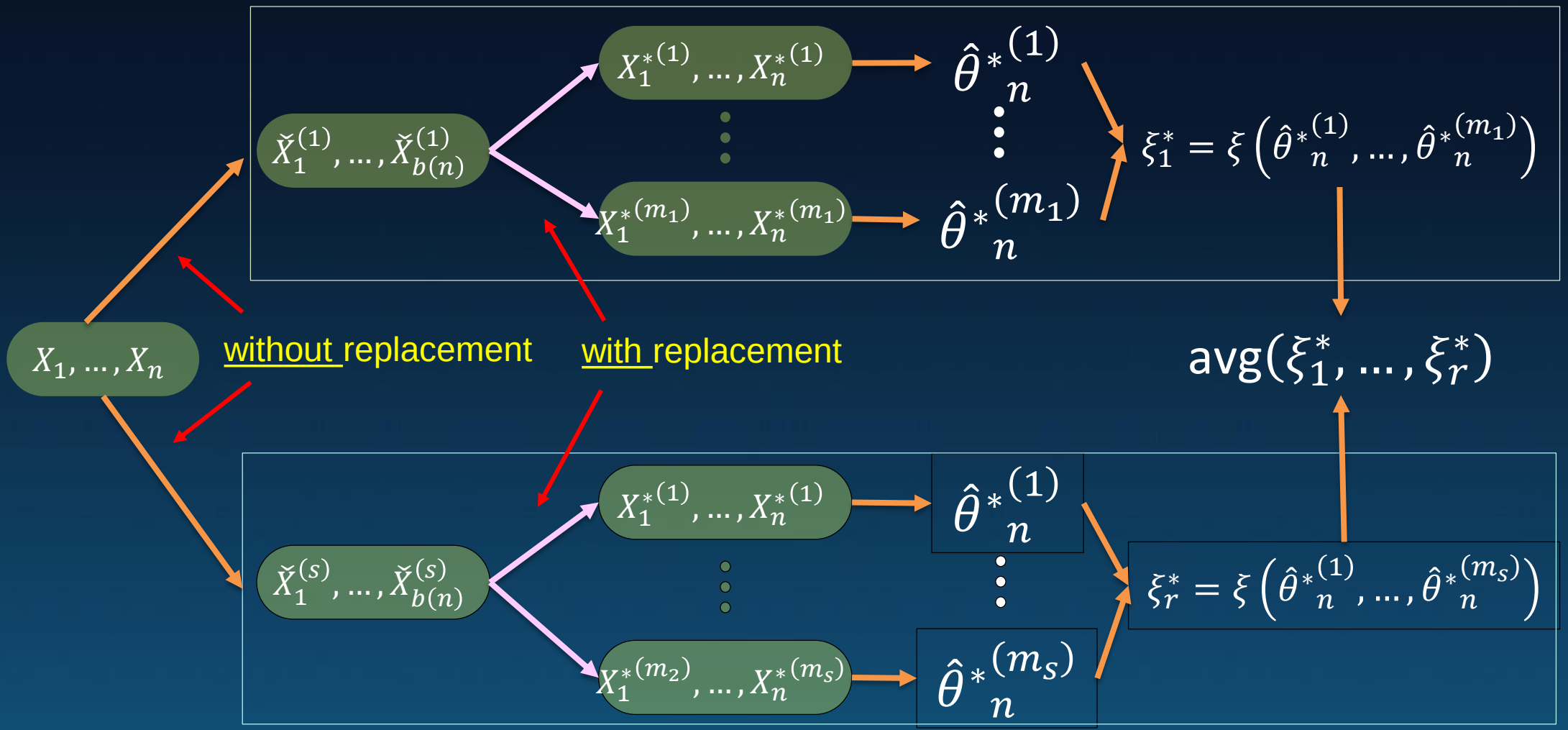
[Slides from Jordan, ICML14]

Scale up Model Evaluation

Bootstrap for massive data

[Kleiner, Jordan 12]

- The Bag of Little Bootstrap (BLB)



Scale up Model Evaluation

Bootstrap for massive data

[Kleiner, Jordan 12]

- **BLB is asymptotically consistent and higher-order correct** (like the bootstrap), under essentially the same conditions that have been used in prior analysis of the bootstrap.

Theorem (asymptotic consistency): Under standard assumptions (particularly that θ is Hadamard differentiable and ξ is continuous), the output of BLB converges to the population value of ξ as n, b approach ∞, then:

$$\left| m_1^{-1} \sum_{j=1}^{m_1} \xi\left(Q_n\left(\mathbb{P}_{n,b}^{(j)}\right)\right) - \xi(Q_n(P)) \right| = O_P\left(\frac{\sqrt{\mathrm{Var}\left(\hat{p}_k^{(j)} - p_k \middle| \mathbb{P}_n\right)}}{\sqrt{nm_1}}\right) + O_P\left(\frac{1}{n}\right) + O\left(\frac{1}{b\sqrt{n}}\right)$$

Therefore, taking $m_1 = \Omega\left(n\mathrm{Var}\left(\hat{p}_k^{(j)} - p_k \mid \mathbb{P}_n\right)\right)$ and $b = \Omega(\sqrt{n})$ yields

$$\left| m_1^{-1} \sum_{j=1}^{m_1} \xi\left(Q_n\left(\mathbb{P}_{n,b}^{(j)}\right)\right) - \xi(Q_n(P)) \right| = O_P\left(\frac{1}{n}\right),$$

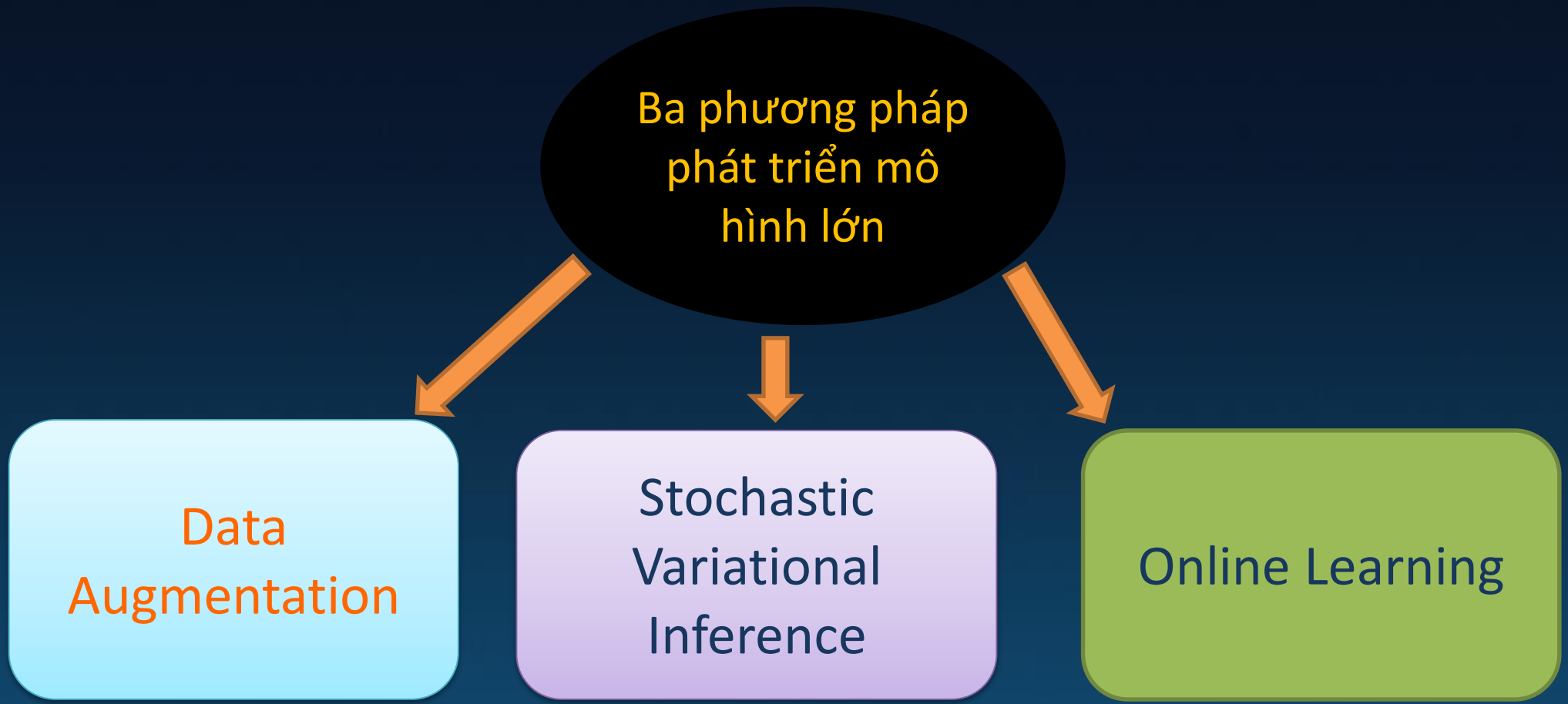
in which case BLB enjoys the same level of higher-order correctness as the bootstrap.

Ba cách tiếp cận để phát triển mô hình lớn

Three approaches to develop new scalable models

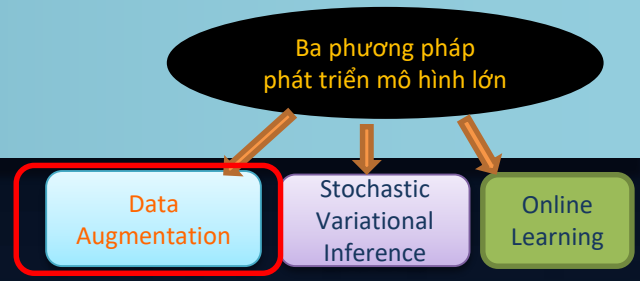
- Mô hình lớn cho dữ liệu lớn
  - Three approaches to scale up big models
    - Scale up Gradient Descent Methods
    - Scale up Probabilistic Inference
    - Scale up Model Evaluation (bootstrap)
- Ba cách tiếp cận để phát triển mô hình lớn
  - Data augmentation
  - Stochastic Variational Inference for Graphical Models
  - Stochastic Gradient Descent and Online Learning
- Công nghệ nào cho mô hình lớn?
  - Hadoop, Spark, TensorFlow

Three Approaches to Develop Scalable Models



Data Augmentation

Phương pháp bổ sung tham số độc lập



- Các bước chính:
  1) Formulate OP as sampling problem
  2) Aim to sample the joint distribution via Gibbs sampling
  3) Introduce auxiliary variables to turn intractable conditional distributions into tractable forms

- A useful result [**Polson JMLR'13**]: from Hinge-loss to distribution
  - $\exp(-|ab|) = \int_0^\infty \frac{a}{\sqrt{2\pi\lambda}} \exp\left\{-\frac{1}{2}(a^2\lambda + b^2\lambda^{-1})\right\} d\lambda$
  - Hence, $\exp\{-2\max(1 - y_i\boldsymbol{\theta}^T\boldsymbol{x}_i)\} = \int_0^\infty \frac{1}{\sqrt{2\pi\lambda_i}} \exp\left\{-\frac{(1+\lambda_i - y_i\boldsymbol{\theta}^T\boldsymbol{x}_i)^2}{2\lambda_i}\right\} d\lambda_i$
  - Therefore, $p(y_i, \lambda_i \mid \boldsymbol{\theta}, \boldsymbol{x}_i) = \frac{1}{\sqrt{2\pi\lambda_i}} \exp\left\{-\frac{(1+\lambda_i - y_i\boldsymbol{\theta}^T\boldsymbol{x}_i)^2}{2\lambda_i}\right\}$ We now can sample!

Data Augmentation

Giải bài toán SVM với hàng trăm triệu điểm dữ liệu

[Tu et al, ICPR16]

- Cách tiếp cận

Chuyển bài toán tối ưu sang suy diễn Bayes (MAP)

SVM giải bài toán tối ưu

$$\min_{\boldsymbol{\theta}} \left(\frac{\alpha}{2} \|\boldsymbol{\theta}\|_2^2 + \frac{1}{N} \sum_{n=1}^{N} \max(0, 1 - y_n \boldsymbol{\theta}^T \boldsymbol{x}_n) \right)$$

The conditional distribution for $\boldsymbol{\theta}$ is

$$p(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{y}, \alpha) \propto p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \alpha)$$

where

- $p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{\theta}) \propto \frac{1}{N} \sum_n \exp\left(-\max(0, 1 - y_n \boldsymbol{\theta}^T \boldsymbol{x}_n)\right)$
- $p(\boldsymbol{\theta} \mid \alpha) = \mathcal{N}(0, \alpha^{-1})$

Đại lượng khó xử lý (intractable)

Dùng kết quả của Polson để thêm biến phụ độc lập (auxiliary variables)

Introduce auxiliary variables $\lambda_{1:N}$ such that

$$\int_{\lambda_n} p(y_n, \lambda_n \mid \boldsymbol{x}_n, \boldsymbol{\theta}) = p(y_n \mid \boldsymbol{x}_n, \boldsymbol{\theta})$$

- Sample $\boldsymbol{\theta}$:

$$p(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\tau}, \alpha) \propto \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Sample λ_n:

$$p(\lambda_n \mid \boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\tau}_{-n}, \lambda) \sim \mathcal{IG}(|1 - y_n \boldsymbol{\theta}^T \boldsymbol{x}_n|^{-1}, 1)$$

Bây giờ ta đã có thể tính song song

Data Augmentation

Giải bài toán SVM với hàng trăm triệu điểm dữ liệu

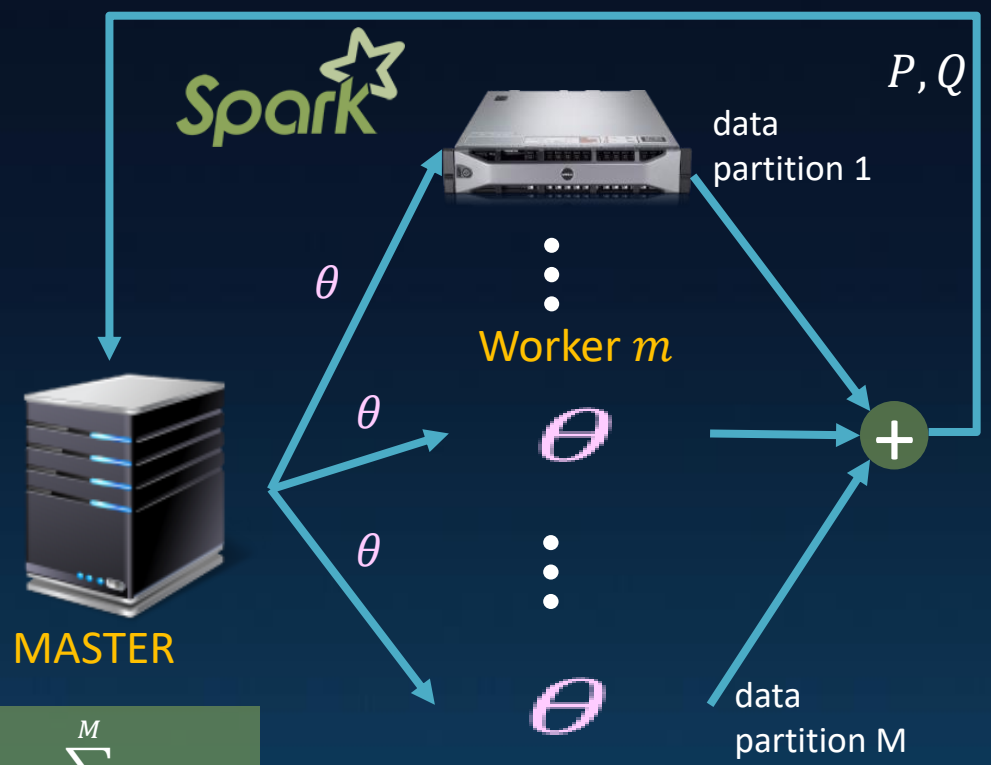
[Tu et al, ICPR16]

- **Distributed Data Augmented Support Vector Machine**
  - Sampling each λ_n is independent
  - Sample $\boldsymbol{\theta}$:
    - We have $p(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\tau}, \lambda) \propto \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.
    - Then $\boldsymbol{\theta}_{\text{MAP}} = \boldsymbol{\mu}$ where

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}[\mathbf{X}^{\mathrm{T}}(\mathbf{1} + \boldsymbol{\lambda}^{-1})]$$

$$\boldsymbol{\Sigma}^{-1} = \boldsymbol{X}^{T}\text{diag}(\boldsymbol{\lambda}^{-1})\boldsymbol{X}$$

    - Let $P = X^T\text{diag}(\lambda)X + I$ and $Q = X^T Z$ where $Z = \{-1,1\}^{NK\times 2}$ and z_{nk}=1 if $y_n = (k+1)/2$, then we have $\theta = P\backslash\mathrm{Q}$ (i.e, $P\theta_k = Q_k$)
    - Can be parallelized in a distributed system in a similar way to parallelize matrix multiplication problem



$$P = \sum_{m=1}^{M} P_m - I$$

$$Q = \sum_{m=1}^{M} Q_m$$

$$\theta = P\backslash\mathrm{Q}$$

$$\lambda_n^{-1} \sim \mathcal{IG}(|1 - y_n\theta^T x_n|^{-1}, 1)$$

$$P_m = X_m^T\text{diag}(\lambda_m)X_m$$

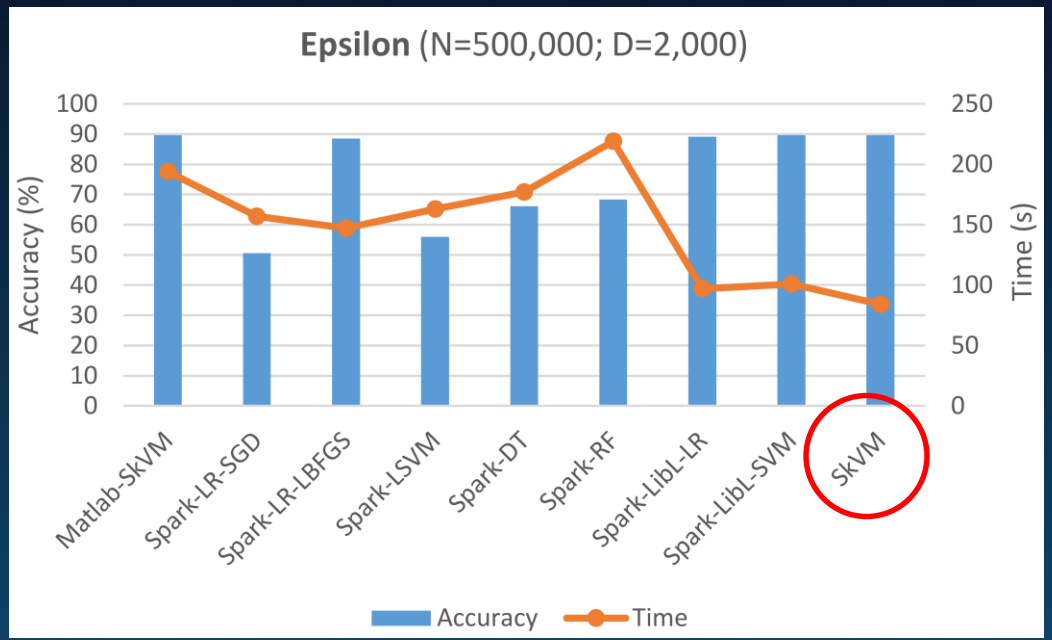
$$Q_m = X_m^T|1 + \text{diag}(\lambda_m^{-1})|Z_m$$

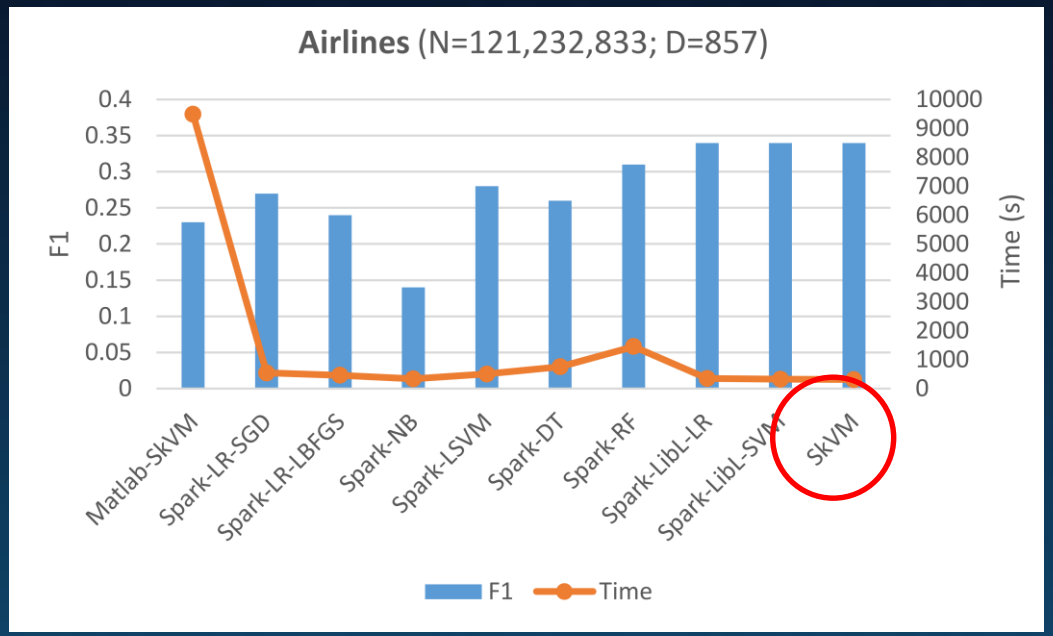
Data Augmentation

Giải bài toán SVM với hàng trăm triệu điểm dữ liệu

[Tu et al, ICPR16]

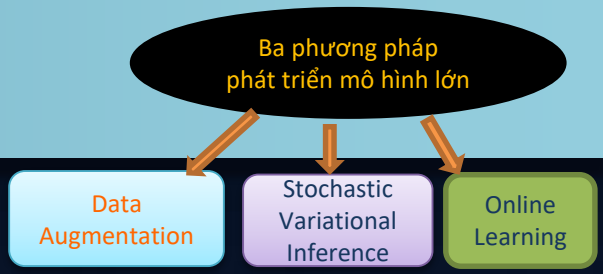
- Tập dữ liệu:
  - Epsilon: # data points = 500,000 and dimension = 2,000
  - Airline: # data points = 121,232,833 and dimension = 857





- For scaling up logistic regression, see our ICDM'16 paper.

Variational Inference
Suy diễn biến phân



- Học một mô hình θ từ dữ liệu x bằng phương pháp suy diễn Bayes cũng chính là tìm phân bố posterior $p(\theta \mid x)$
- Variational inference
  - Approximate $p(\theta \mid x)$ with a $q(\theta)$ where $q(\theta)$ is tractable
  - How to evaluate the closeness: Kullback-Leibler (KL) divergence

$$KL(q,p) = \int q(\theta) \ln \frac{q(\theta)}{p(\theta \mid x)} d\theta$$

  - Optimize KL divergence from $q(\theta)$ to $p(\theta \mid D)$ to is to maximize the Evidenced Lower Bound (ELBO)

$$\mathcal{F}(q) = \mathbb{E}_q[\ln p(x \mid \theta)] - KL(q(\theta),p(\theta \mid x))$$

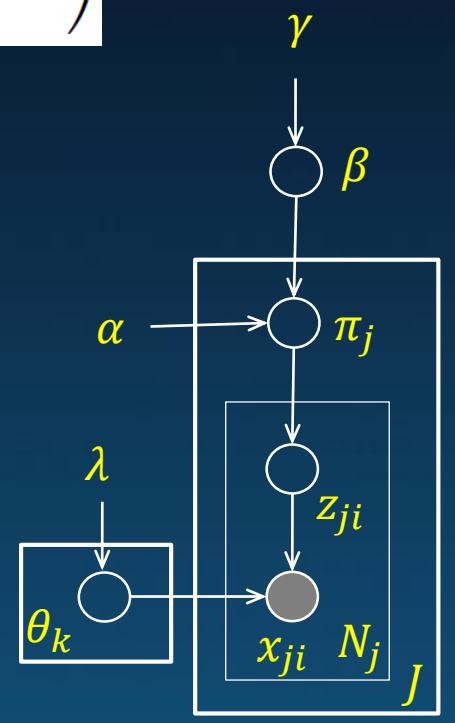
Variational Inference
Suy diễn biến phân

- Mô hình chủ đề (topic models)

$$\mathbb{P}\left(\mathbf{x} = \boxed{\begin{array}{l}\text{The William Randolph Hearst Foundation will give \$1.25 million to Lincoln Center, Metropoli-}\\ \text{tan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a}\\ \text{real opportunity to make a mark on the future of the performing arts with these grants an act}\\ \text{every bit as important as our traditional areas of support in health, medical research, education}\\ \text{and the social services," Hearst Foundation President Randolph A. Hearst said Monday in}\\ \text{announcing the grants. Lincoln Center's share will be \$200,000 for its new building, which}\\ \text{will house young artists and provide new public facilities. The Metropolitan Opera Co. and}\\ \text{New York Philharmonic will receive \$400,000 each. The Juilliard School, where music and}\\ \text{the performing arts are taught, will get \$250,000. The Hearst Foundation, a leading supporter}\\ \text{of the Lincoln Center Consolidated Corporate Fund, will make its usual annual \$100,000}\\ \text{donation, too.}\end{array}} \middle| \boldsymbol{\theta} = \begin{array}{|cccc|}\hline \text{"Arts"} & \text{"Budgets"} & \text{"Children"} & \text{"Education"}\\ \hline \text{NEW} & \text{MILLION} & \text{CHILDREN} & \text{SCHOOL}\\ \text{FILM} & \text{TAX} & \text{WOMEN} & \text{STUDENTS}\\ \text{SHOW} & \text{PROGRAM} & \text{PEOPLE} & \text{SCHOOLS}\\ \text{MUSIC} & \text{BUDGET} & \text{CHILD} & \text{EDUCATION}\\ \text{MOVIE} & \text{BILLION} & \text{YEARS} & \text{TEACHERS}\\ \text{PLAY} & \text{FEDERAL} & \text{FAMILIES} & \text{HIGH}\\ \text{MUSICAL} & \text{YEAR} & \text{WORK} & \text{PUBLIC}\\ \text{BEST} & \text{SPENDING} & \text{PARENTS} & \text{TEACHER}\\ \text{ACTOR} & \text{NEW} & \text{SAYS} & \text{BENNETT}\\ \text{FIRST} & \text{STATE} & \text{FAMILY} & \text{MANIGAT}\\ \text{YORK} & \text{PLAN} & \text{WELFARE} & \text{NAMPHY}\\ \text{OPERA} & \text{MONEY} & \text{MEN} & \text{STATE}\\ \text{THEATER} & \text{PROGRAMS} & \text{PERCENT} & \text{PRESIDENT}\\ \text{ACTRESS} & \text{GOVERNMENT} & \text{CARE} & \text{ELEMENTARY}\\ \text{LOVE} & \text{CONGRESS} & \text{LIFE} & \text{HAITI}\\ \hline \end{array}\right)$$

- Biến ẩn (latent variables): $\Phi = \{z, \pi, \theta\}$
  - Posterior: $p(\Phi \mid x)$, variational distribution $q(\Phi)$
  - Bài toán suy diễn biến phân trở thành bài toán tìm cực đại:

$$-KL(q,p) = \int_\Phi q \ln \frac{p}{q} d\Phi = \mathbb{E}_q[\ln p(\Phi, x)] - \mathbb{H}(q) + C$$



Variational Inference
Suy diễn biến phân cho mô hình chủ đề

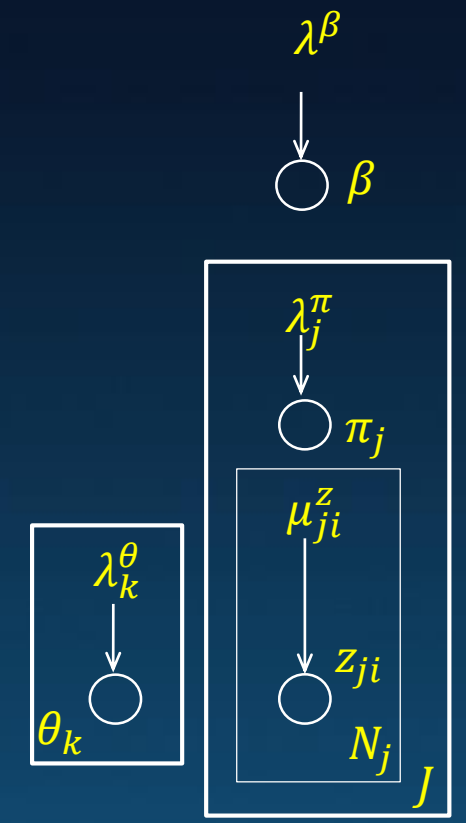
- Giả thuyết mean-field
  - $q(\Phi) = q(\beta \mid \lambda^{\beta}) \prod_{ji} q(z_{ji} \mid \mu_{ji}^{z}) \prod_{j} q(\pi_j \mid \lambda_i^{\pi}) \prod_{k} q(\theta_k \mid \lambda_k^{\theta})$

- Giải thuật suy diễn biến phân cho LDA

  Loop until convergence
  - For $j = 1{:}J$
    - For $i = 1{:}N_j$
      - Update $q(z_{ji}) \propto \exp\left(\mathbb{E}_{q_{-z_{ji}}}[\ln p(\Phi, x_{ji})]\right)$
    - Update $q(\pi_j) \propto \exp\left(\mathbb{E}_{q_{-\pi_j}}[\ln p(\Phi, x_j)]\right)$

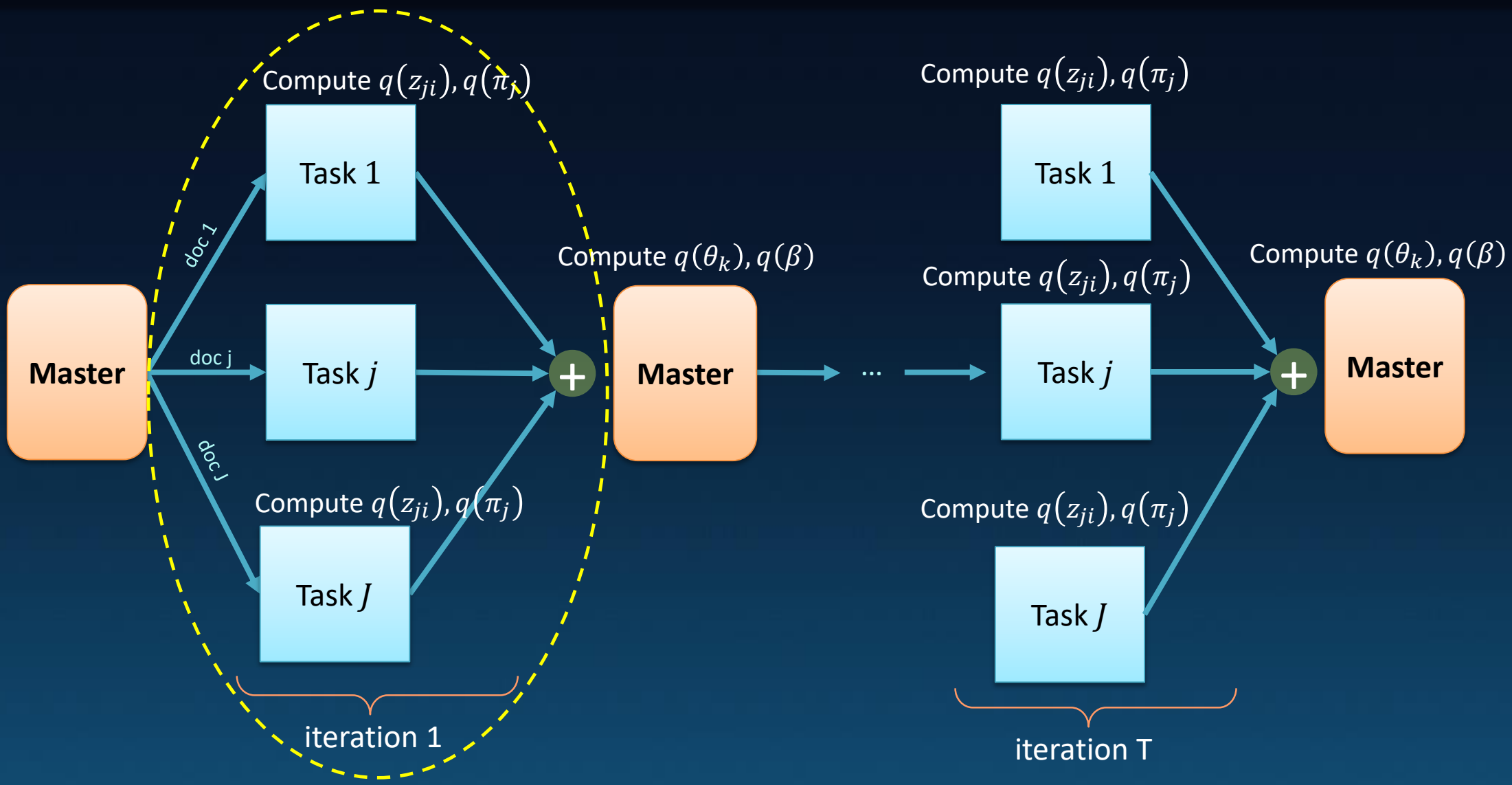
    } local variables
  - Update $q(\theta_k) \propto \exp\left(\mathbb{E}_{q_{-\theta_k}}[\ln p(\Phi, x)]\right)$
  - Update $q(\beta) \propto \exp\left(\mathbb{E}_{-\beta}[\ln p(\Phi, x)]\right)$

    } global variables



Distributed Variational Inference

Xử lý phân tán cho suy diễn biến phân cho mô hình chủ đề



Each iteration involves the computation on the entire dataset of J documents.

Stochastic Variational Inference (SVI)

Suy diễn biến phân ngẫu nhiên cho mô hình chủ đề

[Hofmann, et al., JMLR 13]

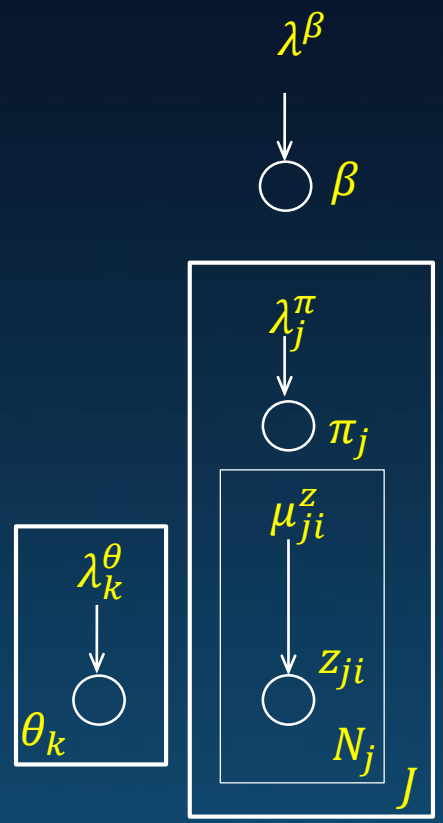
- Giải thuật suy diễn biến phân ngẫu nhiên cho LDA

  Loop until convergence
  - For $j = 1{:}J$
    - For $i = 1{:}N_j$
      - Update $q(z_{ji}) \propto \exp\left(\mathbb{E}_{q-z_{ji}}[\ln p(\Phi, x_{ji})]\right)$
    - Update $q(\pi_j) \propto \exp\left(\mathbb{E}_{q-\pi_j}[\ln p(\Phi, x_j)]\right)$

  } local variables

  - Update $\lambda_k^{\theta(t)} = \lambda_k^{\theta(t-1)} + \rho^{(t)} \frac{|D_t|}{|D|} \frac{\partial F(q(\Theta), D_t)}{\partial \lambda_k^{\theta}}$
  - Update $\lambda^{\beta(t)} = \lambda^{\beta(t-1)} + \rho^{(t)} \frac{|D_t|}{|D|} \frac{\partial F(q(\Theta), D_t)}{\partial \lambda^{\beta(t)}}$

  } global variables

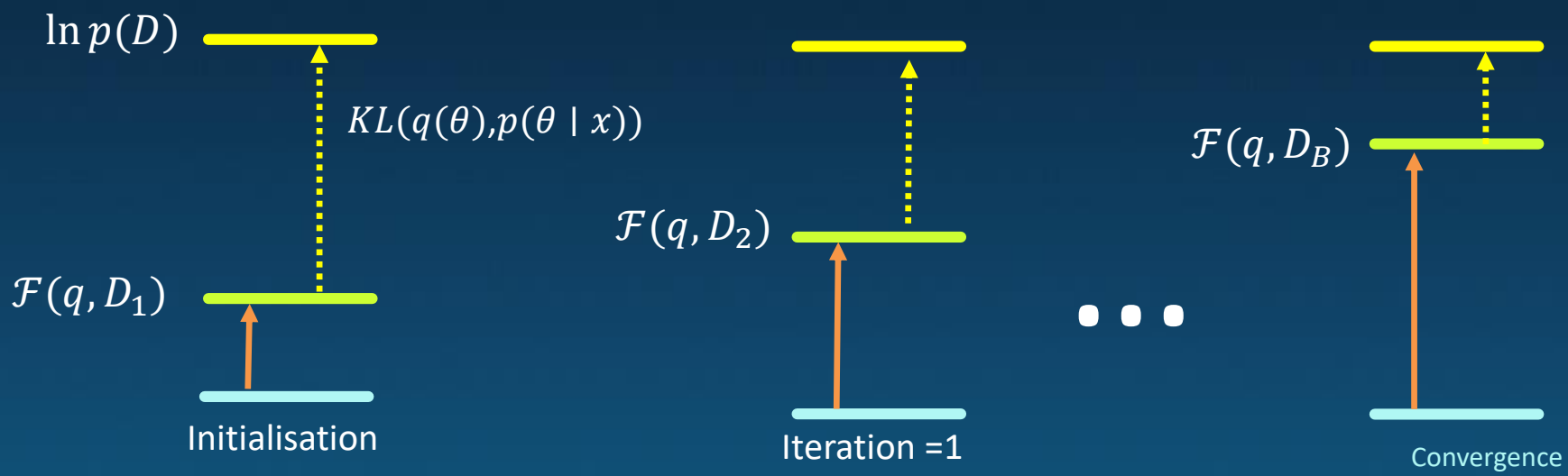


Where D_t is a (much smaller and randomly sampled) subset of entire dataset D, while $S = |D_t|$ and $J = |D|$ are sizes of two data sets.

Stochastic Variational Inference (SVI)

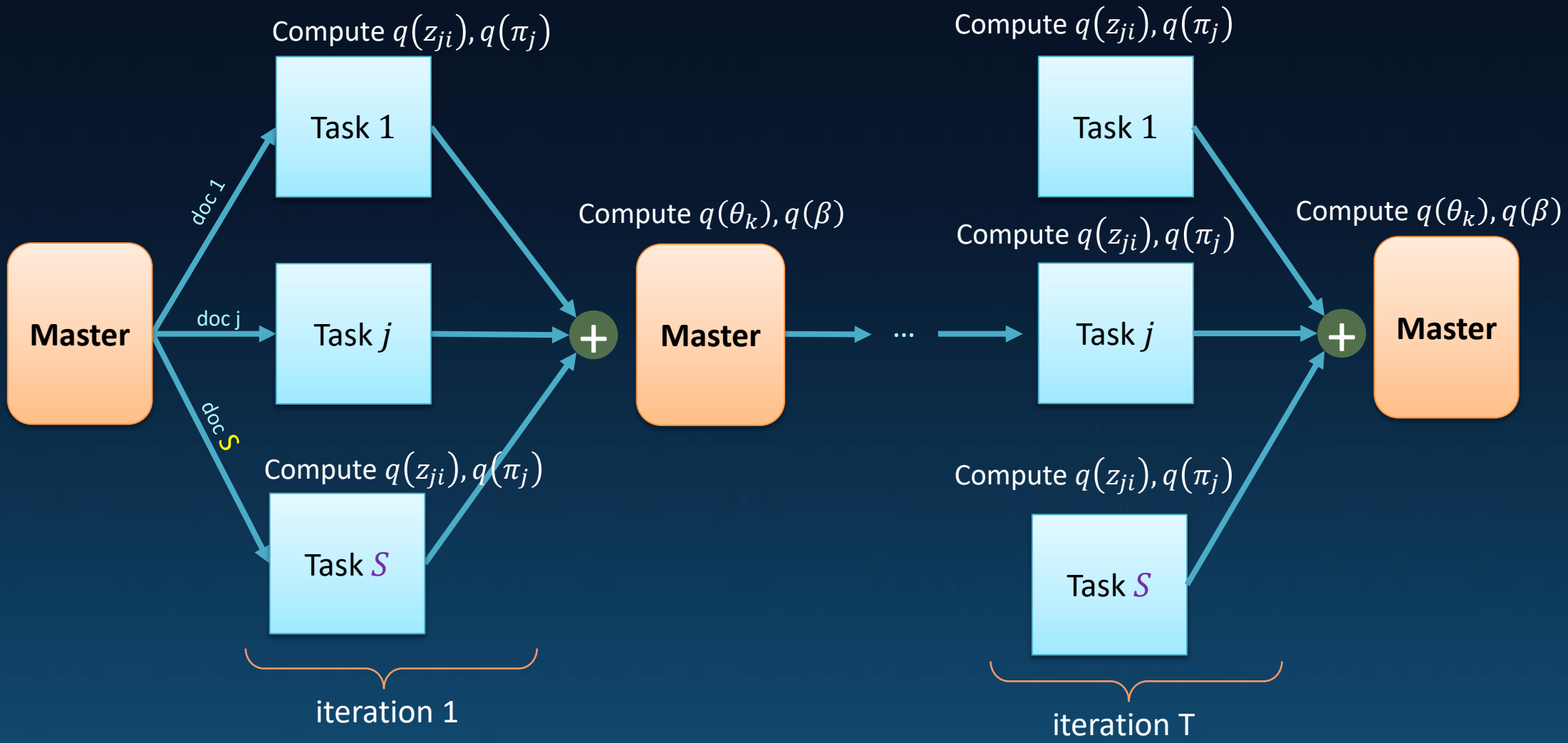
Suy diễn biến phân ngẫu nhiên

- Variational Inference
  - Each step to update lower bound requires entire data D
  - D is large-scale → bottleneck
- Stochastic variational inference
  - use a (randomly chosen small) subset of D, $D_B \subset D$ in each iteration
  - Update the local parameters using coordinate ascent
  - update the global parameters using stochastic gradient ascent



Distributed SVI

Xử lý phân tán cho suy diễn biến phân ngẫu nhiên



Now, each iteration involves the computation on much smaller sub dataset D_t $(S \ll J)$.

Stochastic Variational Inference (SVI)

Suy diễn biến phân ngẫu nhiên

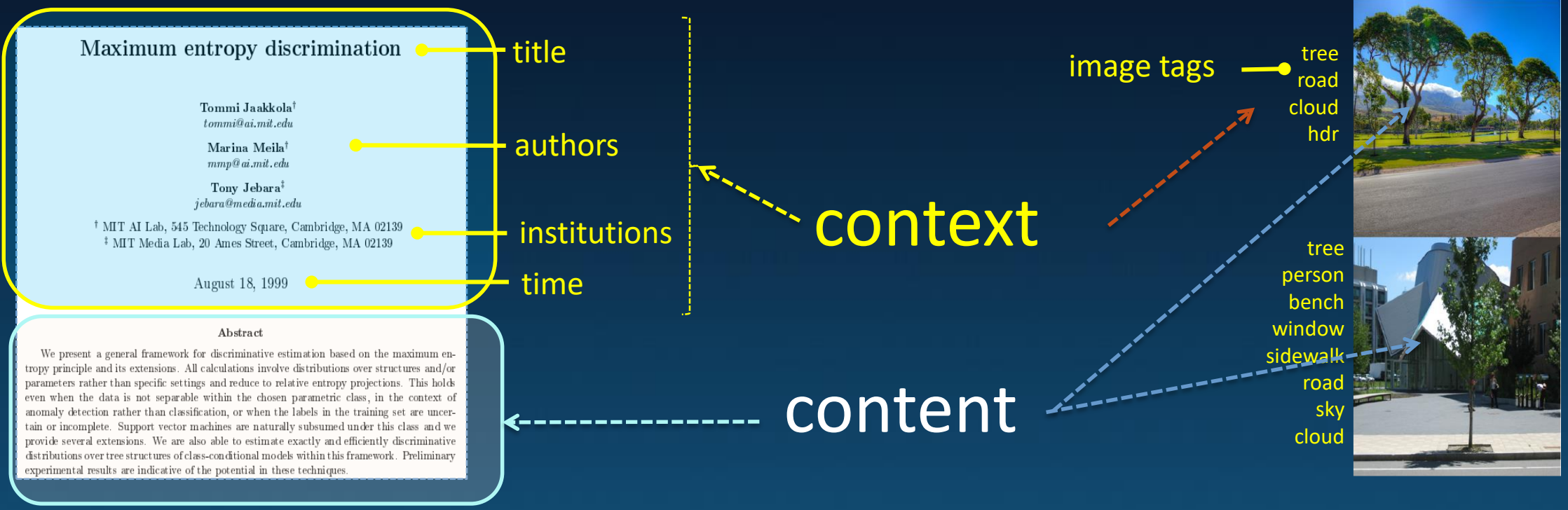
- Stochastic variational inference
  - Lý thuyết graphical models và XSTK cho phép ta xây dựng những mô hình tinh vi (sophisticated models).
  - Nhưng những mô hình này (vd: mô hình chủ đề, topic models) không thể áp dụng vào dữ liệu lớn vì độ phức tạp suy diễn quá lớn.
  - SVI là một phương pháp cực kỳ hiệu quả giúp chúng ta vượt qua trở ngại này.
  - SVI có thể áp dụng vào hầu hết các loại graphical models.
- Ví dụ: làm thế nào để mở rộng mô hình chủ đề (LDA) để không chỉ tìm ra chủ đề ẩn, mà đồng thời phân cụm các tập dữ liệu?
  - Huynh, Phung, Nguyen, Hofman and Bui, *Scalable Nonparametric Bayesian Multilevel Clustering*, UAI'16.

Scalable Bayesian Multilevel Clustering

Gom cụm đa cấp với mô hình chủ đề cho DLL

[Huynh, et al, 16]

- Multilevel clustering with context (MC2)
  - Clustering content-units (words, pixels) into "topics"
  - Forming content-groups (papers, images) into clusters of data groups
    - Leveraging context-units (title, authors, tags, etc.) associated wit each data groups



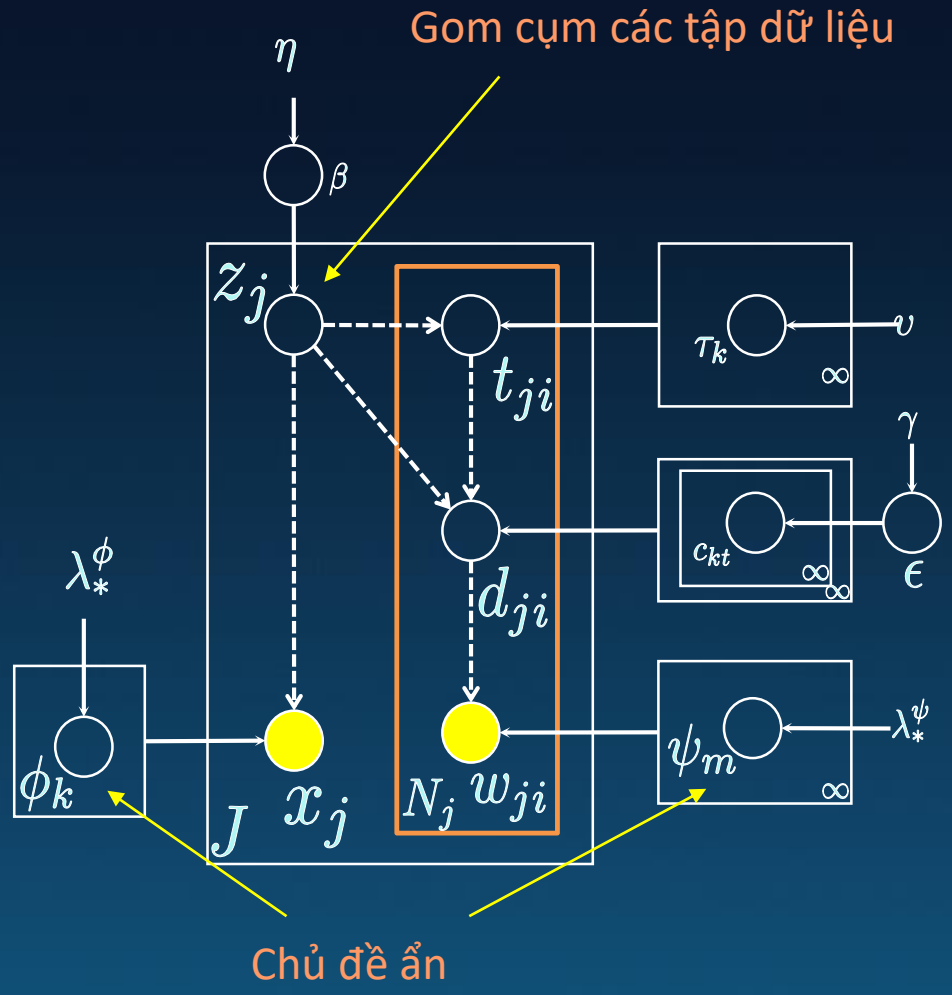
Scalable Bayesian Multilevel Clustering

Gom cụm đa cấp với mô hình chủ đề cho DLL

[Huynh, et al, 16]

- Data $D = \{x_j, w_{ji}\}$
- Parameters $\Theta = \{\beta, z, t, d, c, \tau, \psi, \phi\}$
- Inference:
  - Computing the posterior $\mathrm{p}(\Theta \mid D)$
  - $\mathrm{p}(\Theta \mid D) = \frac{\mathrm{p}(D|\Theta)}{\int \mathrm{p}(D,\Theta) d\Theta}$ (intractable)
- Phương pháp suy diễn
  - Stochastic Variational Inference (SVI)

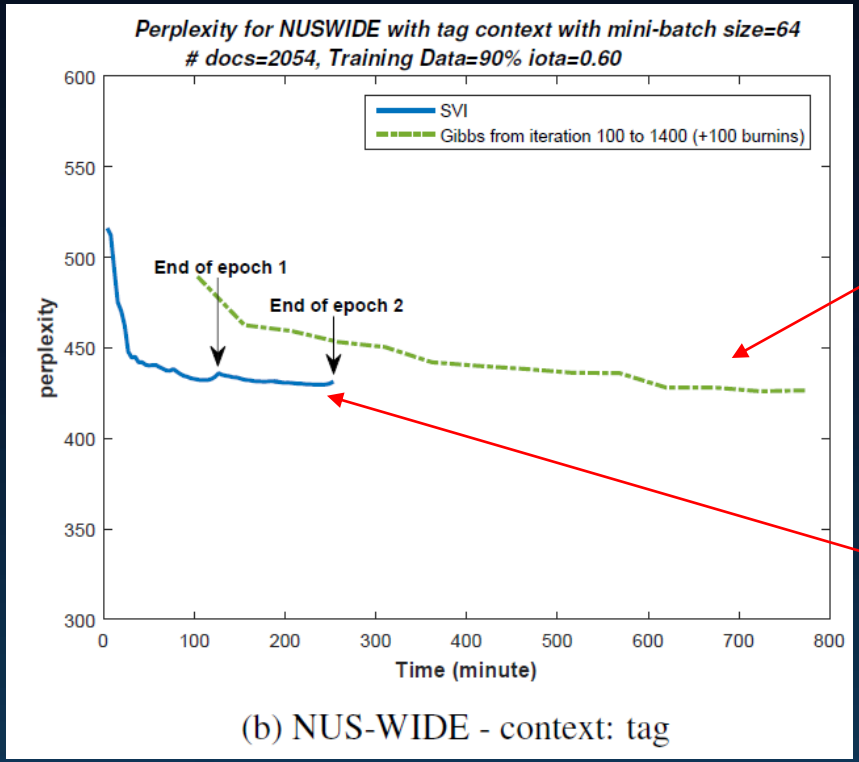
Mô hình MC2



Scalable Bayesian Multilevel Clustering

Tăng tốc giải thuật suy diễn cho MC2 dùng SVI và xử lý song song

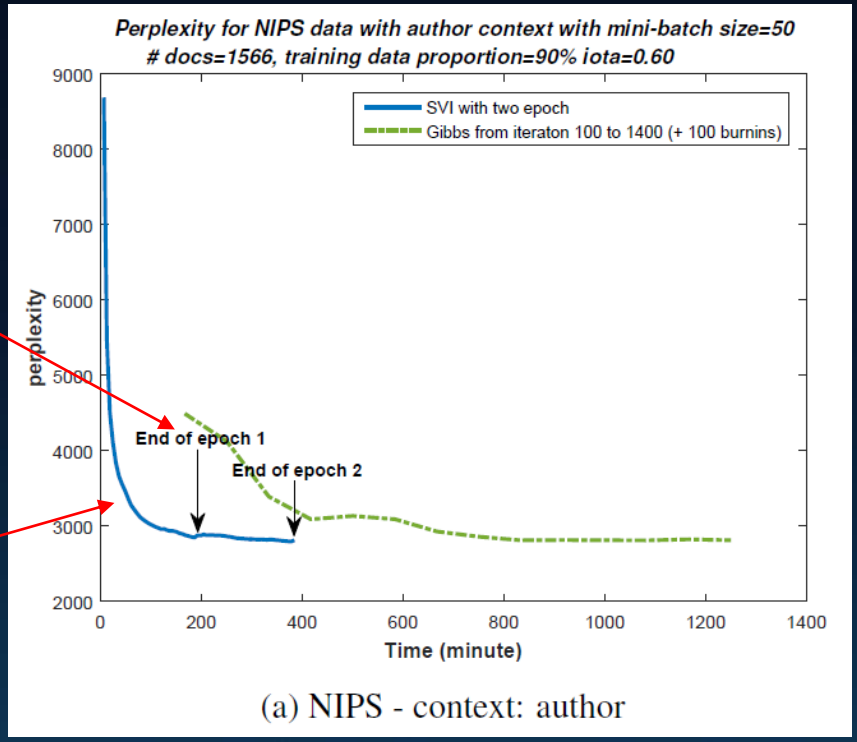
[Huynh, et al, 16]



(b) NUS-WIDE - context: tag

Gibbs sampler

SVI



(a) NIPS - context: author

NIPS dataset

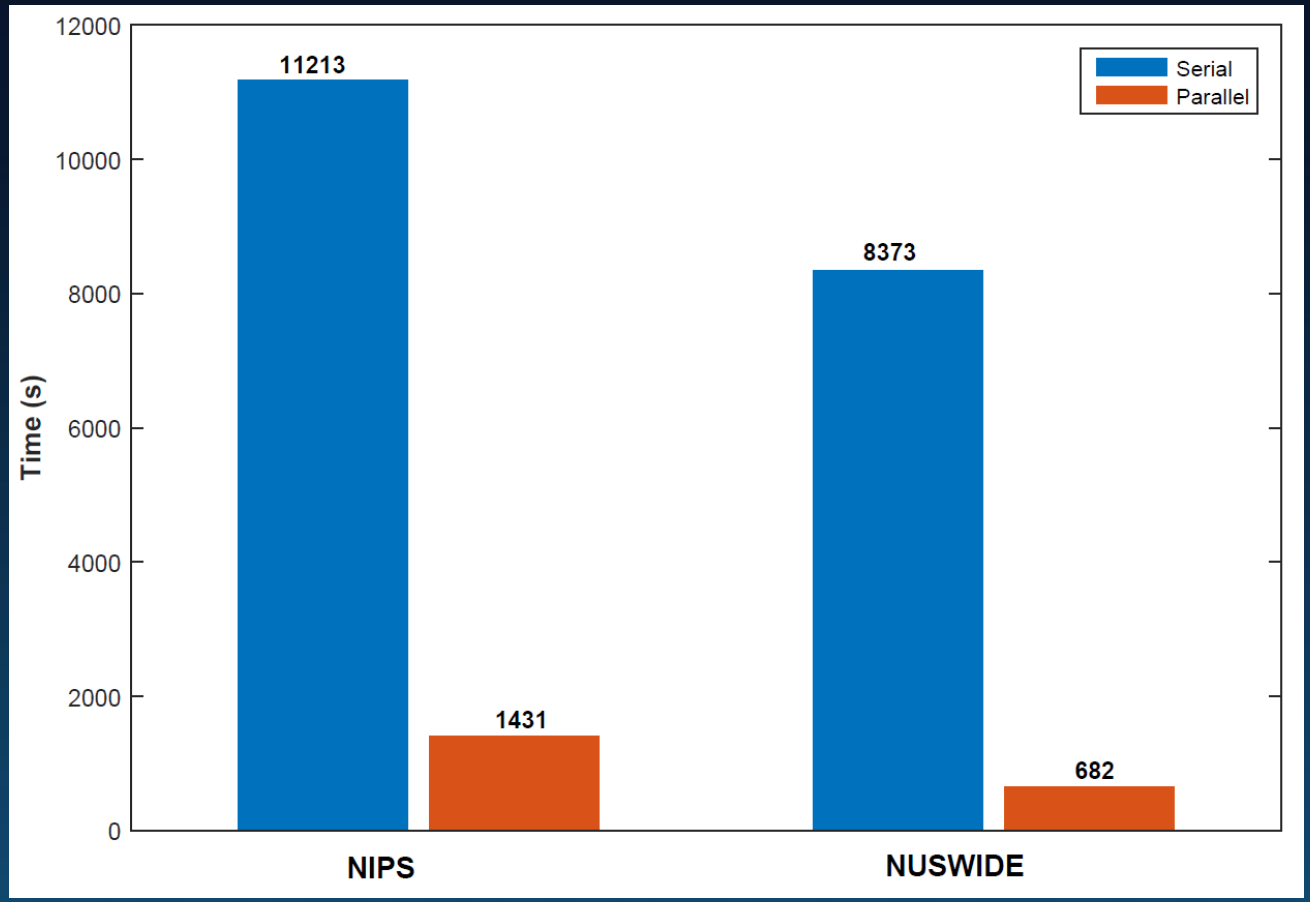
- NIPS Conference Papers Volumes 0-12 (1987-1999)
- 1740 papers, 13649 vocabularies
- Context information: Authors, Institutions, Published year, Title

Scalable Bayesian Multilevel Clustering

Tăng tốc giải thuật suy diễn cho MC2 dùng SVI và xử lý song song

[Huynh, et al, 16]

Parallel vs sequential implementation of SVI

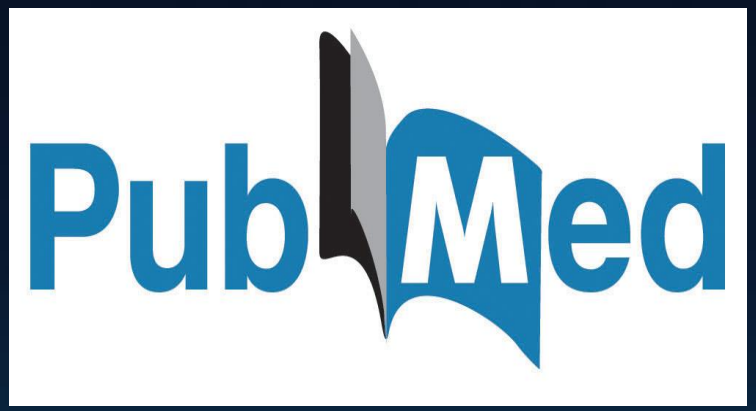


Scalable Bayesian Multilevel Clustering

Áp dụng cho các tập dữ liệu lớn

[Huynh, et al, 16]

- 1.1 million documents (billions of data points)
- Vocabulary size 10K
- Contexts:
  - first writer
  - top categories



- 1.4 million abstracts (billions of data points)
- Vocabulary size 10K
- Context:
  - Medical Subject Headings

Application Usage Activity

- >1.1 million users (billions of data points)
- ~ 10K functionalities
- Context:
  - applications
  - paid/trial users

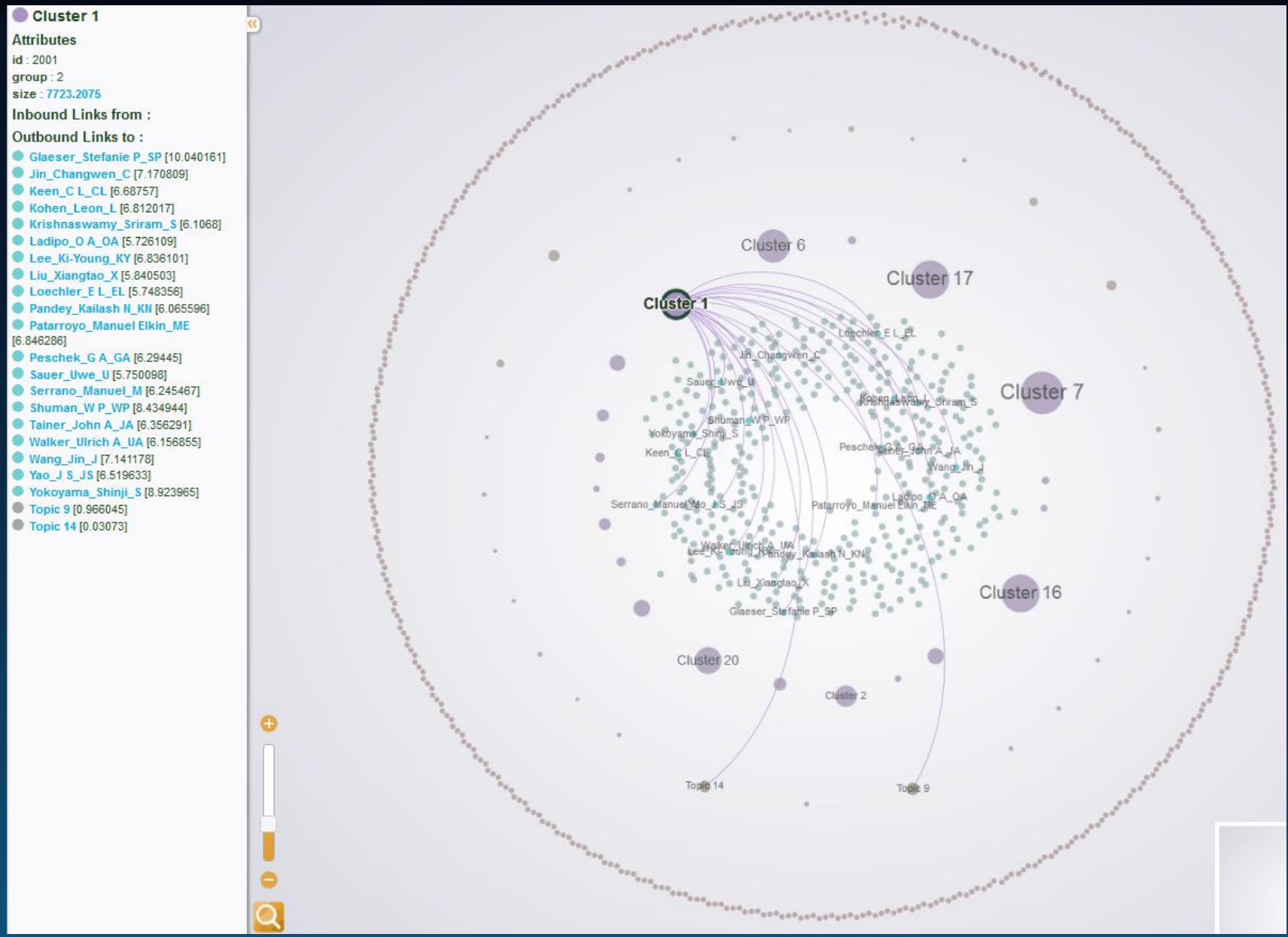
Inference: each iteration update billons of (distributions) of variables !!!

Scalable Bayesian Multilevel Clustering

Kết quả với tập dữ liệu PubMed

[Huynh, et al, 16]

- Nodes
  - Outermost ring: top vocabularies in the corpus.
  - Second outermost ring: topics discovered by MC2 model.
  - Third outermost ring: cluster of documents discovered by MC2 model.
  - Innermost ring: top authors of the papers in the corpus.
- Edges between
  - vocabularies and topics: corresponding vocabularies in topics where edge weights denote words' contribution to topics
  - clusters and topics: topics contributes in each cluster of documents
  - clusters and authors: authors contribute clusters of documents
- When clicking on each node, e.g. cluster 1, the left panel shows nodes connected to active node and normalized weight

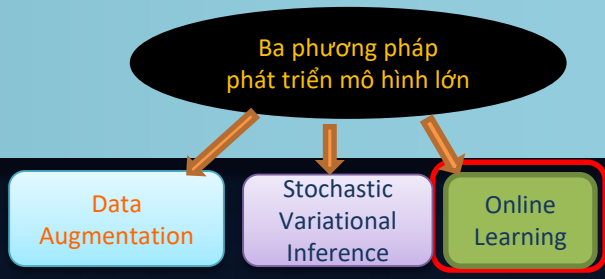


Stochastic Variational Inference

Tài liệu tham khảo thêm và mã nguồn

- More on (stochastic) variational inference for graphical models from our work:
  - Huynh, V., Phung, D., Svetha, V., Nguyen, X.L, Hoffman, M. and Bui, H., *Scalable Nonparametric Bayesian Multilevel Clustering*, **UAI'16**.
  - Huynh, Viet and Phung, Dinh, *Streaming Clustering with Bayesian Nonparametric Models*, **Neurocomputing, 2017**.
  - Nguyen, V., Phung, D., Venkatesh, S. Nguyen, X.L. and Bui, H, *Bayesian Nonparametric Multilevel Clustering with Group-Level Contexts (MC2)*, **ICML'14.**
- Source code:
  - https://github.com/VietHHuynh/MC2SVI
- PubMed visualization
  - https://cdn.rawgit.com/VietHHuynh/MC2SVI/7fb53550/visualization/index.html

Online learning



- Online learning là gì?
  - Trong các hệ thống máy hiện đại, dữ liệu đến hệ thống một cách liên tục.
  - Làm thế nào để phát triển mô hình học cập nhật một cách liên tục?
  - Online learning: họ các mô hình học gia tăng dần từ dãy dữ liệu nhận được.
- Tại thời điểm $(t-1)$, thu được tập dữ liệu D_{t-1} và gọi θ_t là tham số mô hình suy diễn
  - Tại thời điểm t, tiếp nhận dữ liệu mới (x_t, y_t)
  - Cập nhật mô hình theo dữ liệu mới:

$$\theta_{t+1} = h(\theta_t, x_t, y_t)$$

Dòng dữ liệu đến liên tục và không ngừng chuyển động

Online learning

- Online learning
  - Làm thế nào để đánh giá chất lượng của mô hình học theo cách online?
  - Answer: Regret represents the retrospect suffered from the online learning algorithm if the optimal model is used to predict the processed data

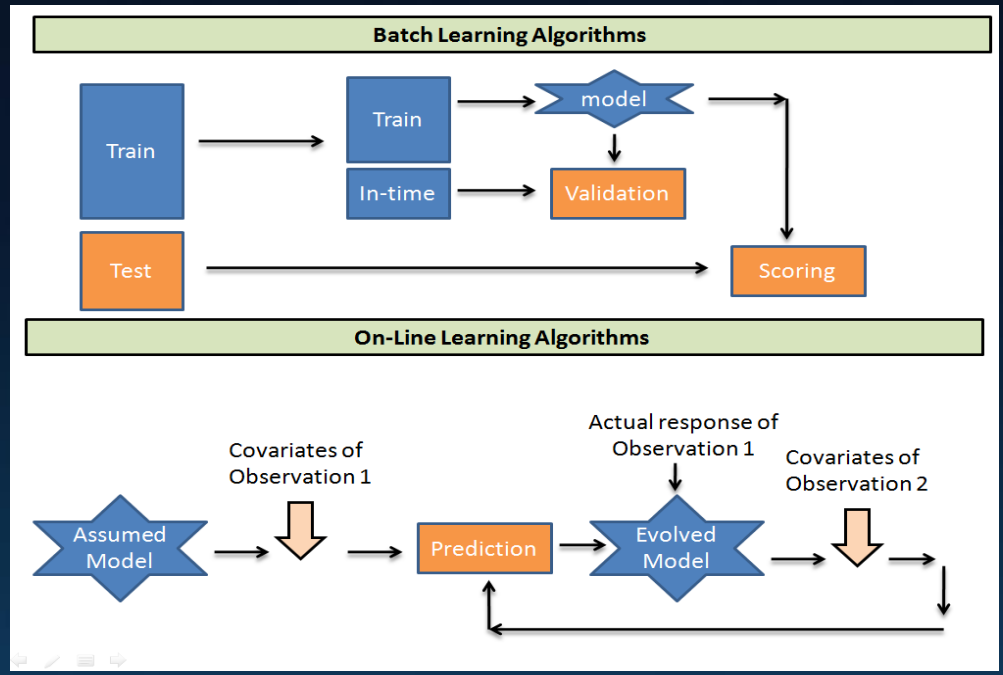
$$\boldsymbol{Regret(u,T) = \sum_{t=1}^{T} \left(l_t(w_t) - l_t(u) \right)}$$

T: số lần cập nhật online
l_t: loss function (e.g., prediction error for point (x_t, y_t)

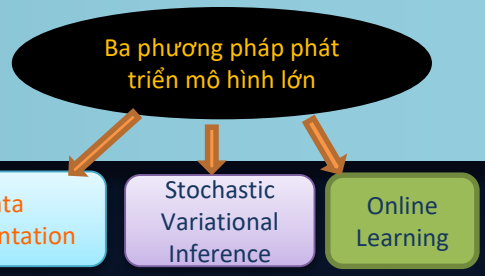
$$\boldsymbol{Regret(T) = \sup_u \left[\sum_{t=1}^{T} \left(l_t(w_t) - l_t(u) \right) \right]}$$

Online learning

- Online learning
  - Giá trị trung bình của regret có thể bị chặn về mặt lý thuyết.
  - **Những tốc độ hội tụ lý tưởng:**
  - $$\frac{Regret(T)}{T} \in O\left(\frac{1}{\sqrt{T}}\right), O\left(\frac{\log T}{T}\right), O\left(\frac{1}{T}\right)$$
  - Như vậy, nếu khéo léo ta có thể thiết kế mô hình bảo đảm tính hội tụ đến lời giải tốt ưu với tốc độ hội tụ nhanh khi T tăng dần.

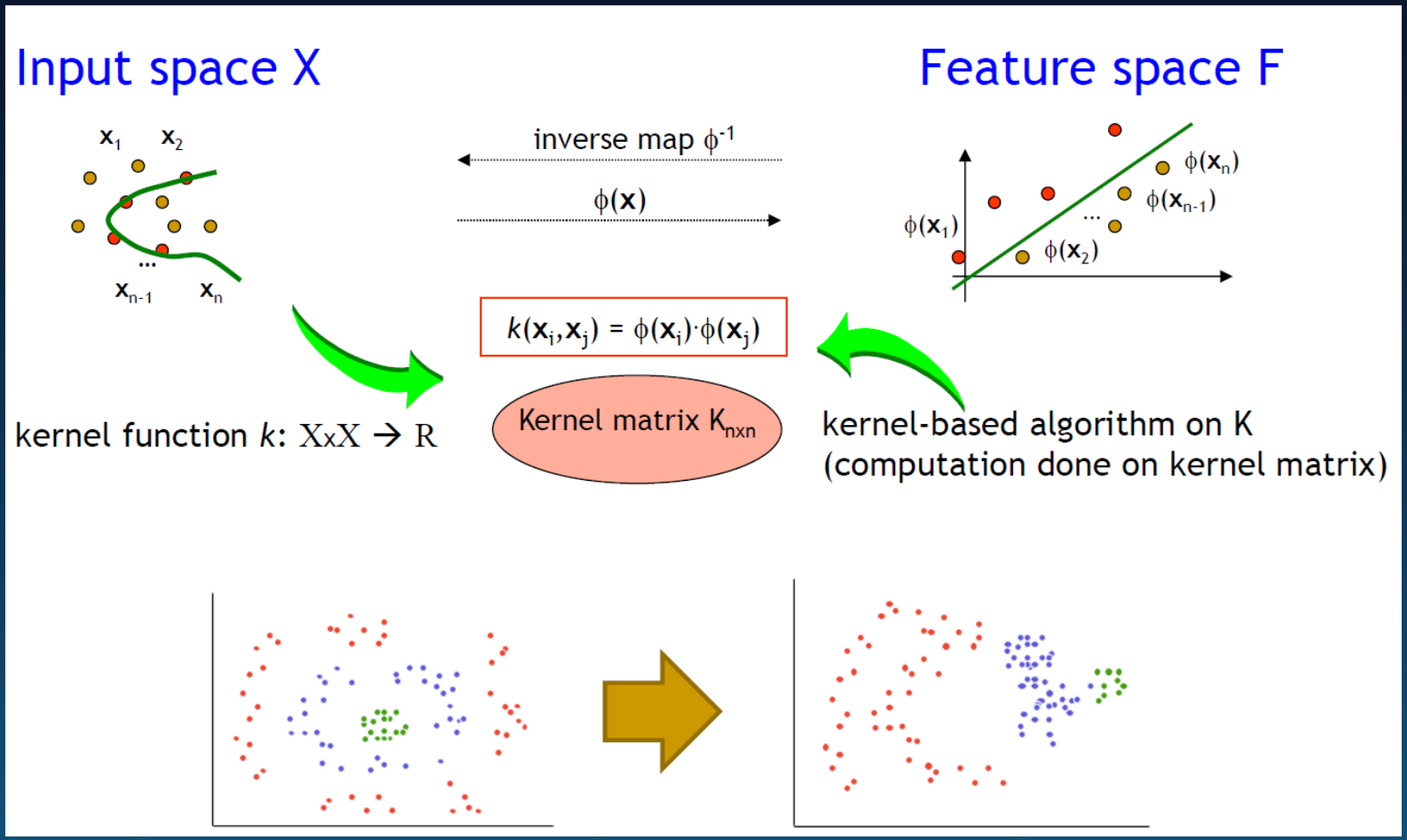


Kernel Online learning



- Kết hợp phương pháp kernel và online learning

Kernel trick: ta chỉ cần định nghĩa kernel function mà không cần chỉ ra dạng tường minh của hàm chiếu $\phi(\cdot)$



Kernel online learning

- Kết hợp thế mạnh của mô hình kernel và online learning.
- Typically, solution for a kernel method applied to online case:
  - Current model $\boldsymbol{\theta}_t = \sum_{i=1}^{t} \alpha_i \phi(\boldsymbol{x}_i)$ where $\phi(.)$ is feature map
  - Model size is defined as $\|\alpha\|_0$
  - Problem: curse of kernelization, i.e., model size increases linearly over time.
  - $\phi(\cdot)$ lies in infinite-space, hence we can't store it directly – all computation is done via the kernel function and α.
- Two approaches to combat the curse of kernelization
  - Budgeted [Wang et al 2012]
    - Limit model size to a predefined budget B (removal, merging, projection)
  - Random features [Lu et al 2016]
    - Express $\phi(\cdot)$ explicitly via Bochner's theorem

Kernel online learning

Bochner's theorem [Bochner, 1959]: *every shift-invariant kernel $k(x, x')$ can be represented as an inverse Fourier transform of a proper distribution $p(\omega)$ as below*:

$$k(x, x') = k(u) = \int p(\omega) e^{i\omega^\top u} du \text{ where } u = x - x'$$

- Consequently, given the form of the kernel, one can derive an explicit distribution for ω: $p(\omega) = \left(\frac{1}{2\pi}\right)^d \int k(u) e^{-iu^\top \omega} du$
  - E.g, if the kernel is a Gaussian kernel, then ω follows a Gaussian distribution.

Using Euler's formula, one can approximate: $k(x, x') \approx \tilde{k}(x, x') = \tilde{\phi}^\top(x)\tilde{\phi}(x')$

where $\tilde{\phi}(x) = \sqrt{\frac{2}{D}}\left[\cos\left(\omega_j^\top x\right), \sin\left(\omega_j^\top x\right)\right]_{j=1}^{D}$

and $\omega_j \overset{iid}{\sim} p(\omega), j = 1,2, \dots, D$

Kernel online learning

Bochner's theorem [Bochner, 1959]: *every shift-invariant kernel $k(x, x')$ can be represented as an inverse Fourier transform of a proper distribution $p(\omega)$ as below*:

$$k(x, x') = k(u) = \int p(\omega) e^{i\omega^\top u} du \text{ where } u = x - x'$$

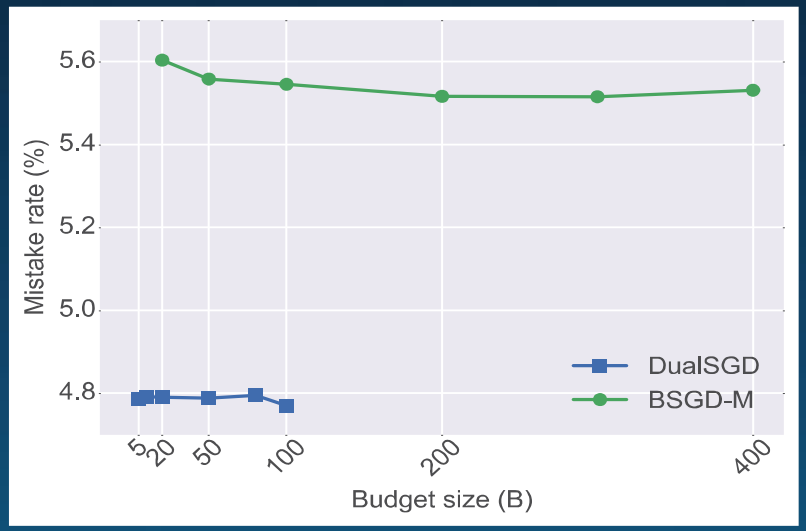
- Định lý Bochner cho phép ta xây dựng hàm chiếu $\phi(\cdot)$ một cách tường minh.
- Kết quả này giúp chúng ta xây dựng nhiều mô hình mới!
  - *Large-scale Online Kernel Learning with Random Feature Reparameterization* [Tu et al., IJCAI 17]
    - Kernel Online learning for both model parameters and kernel parameters.
  - *Dual Space Gradient Descent for Online Learning* [Le et al, NIPS16]
    - Advanced kernel online learning method that ensures theoretical convergence and quality.

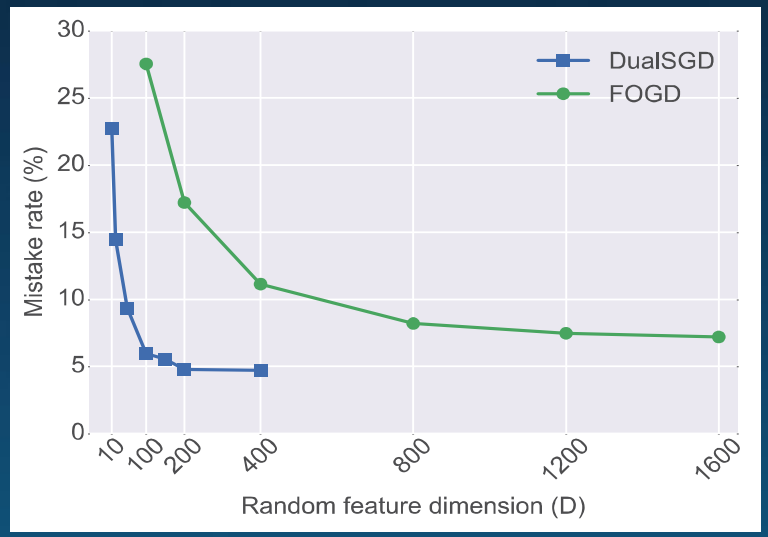
Gradient Descent trên không gian kép

Dual Space Gradient Descent for Online Learning

[Le et al., NIPS16]

- Random feature approach [Lu et al 2015] operates on *random-feature space* (using Bochner's theorem), and then perform SGD in the feature space.
- Problem: excessive number of random features is required
- DualSGD [Le et al. NIPS16]: use both original feature space and random feature space
  - Key intuition: use *a provision vector* in the random-feature space to *store* the information of all vectors being removed via *k-merging* technique.
  - Guarantee in terms of theory for convergence and accuracy.





Tham khảo
References

- For more online learning papers from our group, see:
  - Trung Le, Tu Nguyen, Vu Nguyen and Phung, Dinh, *Approximation Vector Machines*, Journal of Machine Learning Research (**JMLR, 2017**)
  - Trung Le, Tu Nguyen, Vu Nguyen and Dinh Phung, *Dual Space Gradient Descent for Online Learning*, **NIPS'16**.
  - Trung Le, Vu Nguyen, Tu Nguyen and Dinh Phung, *Nonparametric Budgeted Stochastic Gradient Descent*, **AISTATS'16.**
  - Trung Le, Duong Phuong, Dinh Mi, Tu Nguyen and Dinh Phung, *Budgeted Semi-supervised Support Vector Machine*, **UAI'16.**
  - Tu Nguyen, Trung Le, Hung Bui and Dinh Phung, *Large-scale Online Kernel Learning with Random Feature Reparameterization*, **IJCAI'17**.

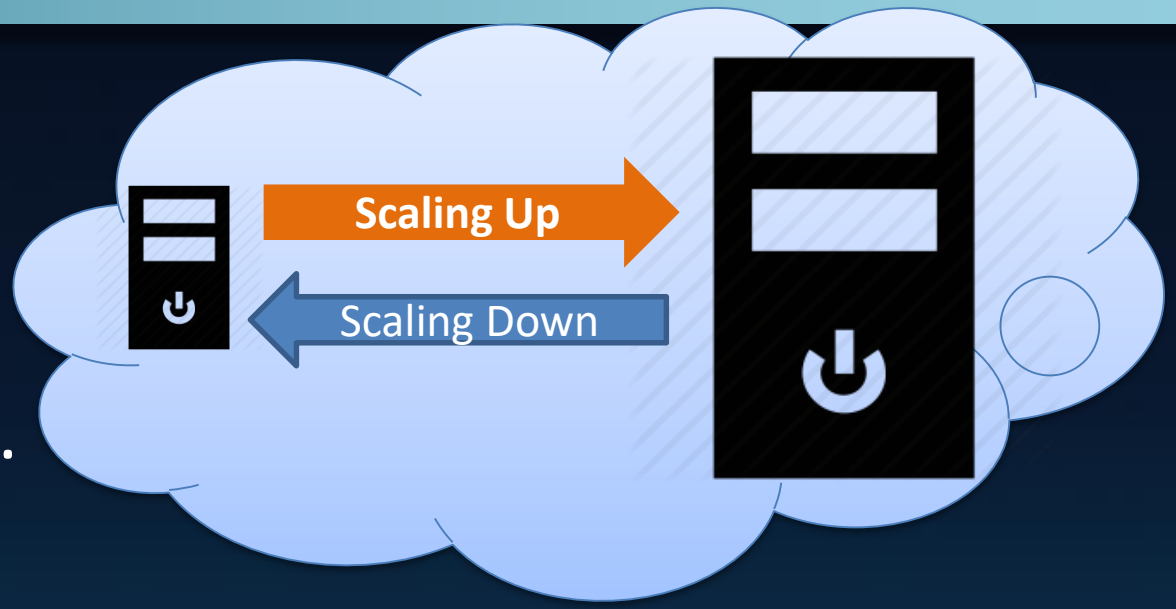
Công nghệ nào cho mô hình lớn?

Platform for big models

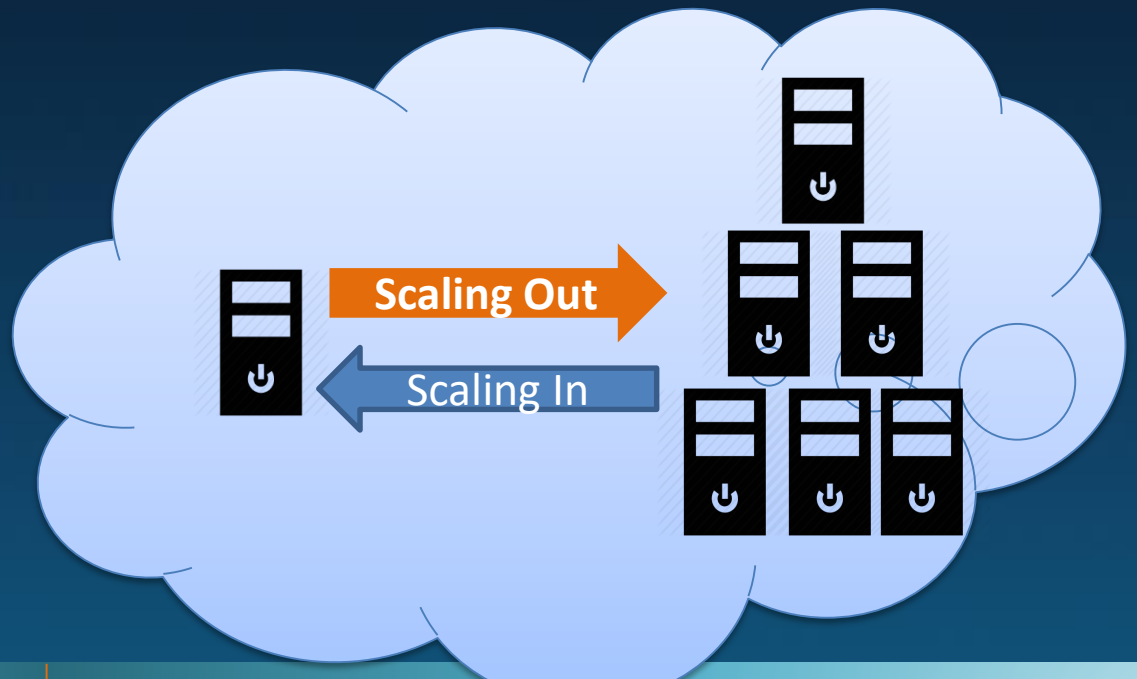
- Mô hình lớn cho dữ liệu lớn
  - Three approaches to scale up big models
    - Scale up Gradient Descent Methods
    - Scale up Probabilistic Inference
    - Scale up Model Evaluation (bootstrap)
- Ba cách tiếp cận để phát triển mô hình lớn
  - Data augmentation
  - Stochastic Variational Inference for Graphical Models
  - Stochastic Gradient Descent and Online Learning
- Công nghệ nào cho mô hình lớn?
  - Hadoop, Spark, TensorFlow

Platform for Big Data

- Scaling up (Vertical Scaling): adding more power (CPU, RAM) to an existing machine.



- Scaling out (Horizontal Scaling): adding more machines into the pool of resources.

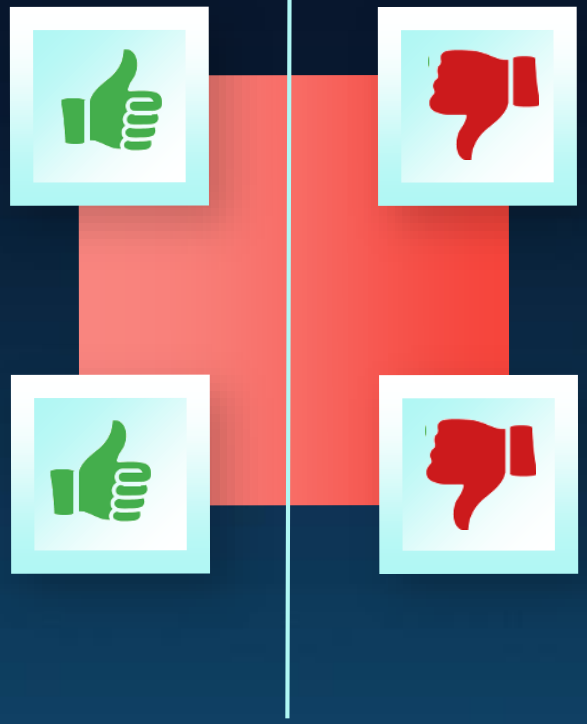


Platform for Big Data

Ưu và nhược điểm của mở rộng theo chiều ngang và chiều dọc

Horizontal Scaling

Increases performance in small steps as needed
Financial investment to upgrade is relatively less
Can scale out the system as much as needed



Horizontal Scaling

Software has to handle all the data distribution and parallel processing complexities
Limited number of softwares are available that can take advantage of horizontal scaling

Vertical Scaling

Most of the software can easily take advantage of vertical scaling
Easy to manage and install hardware within a single machine

Vertical Scaling

Requires substantial financial investment
System has to be more powerful to handle future workloads and initially the additional performance in not fully utilized
It is not possible to scale up vertically after a certain limit

[Nguồn: Singh and Reddy: A survey on platforms for big data analytics. Journal of Big Data 2014 1:8]

Platform for Big Data

Horizontal Scaling with Peer-to-Peer Networks

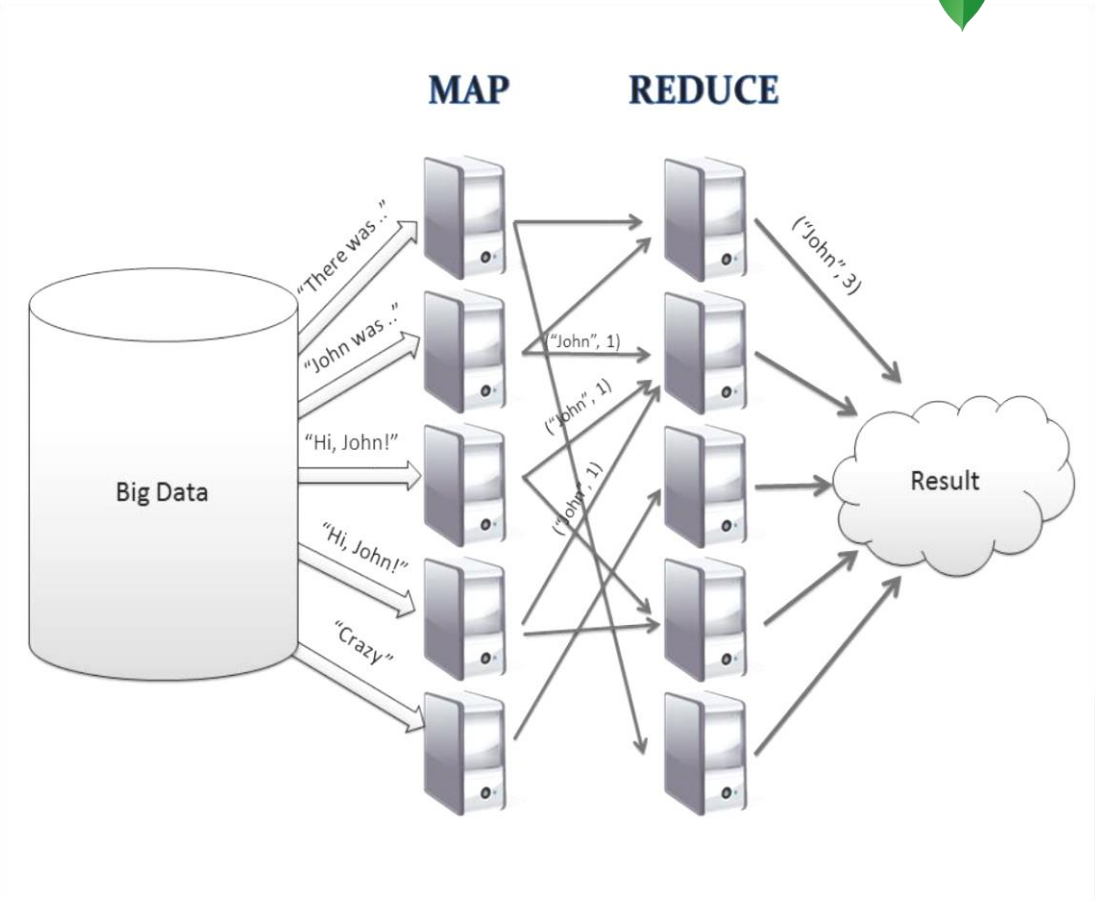
- Peer-to-peer networks
  - millions of machines connected in a network
  - communication scheme exchange the data: Message Passing Interface (MPI)
  - Advantages:
    - well suited for iterative processing
    - available for many programming languages
    - useful for dynamic resource allocation with the master–slave model
  - Disadvantages:
    - no mechanism to handle faults
    - more complex with lots of functions

Horizontal Scaling

Công nghệ MapReduce

Parallel Processing with MapReduce

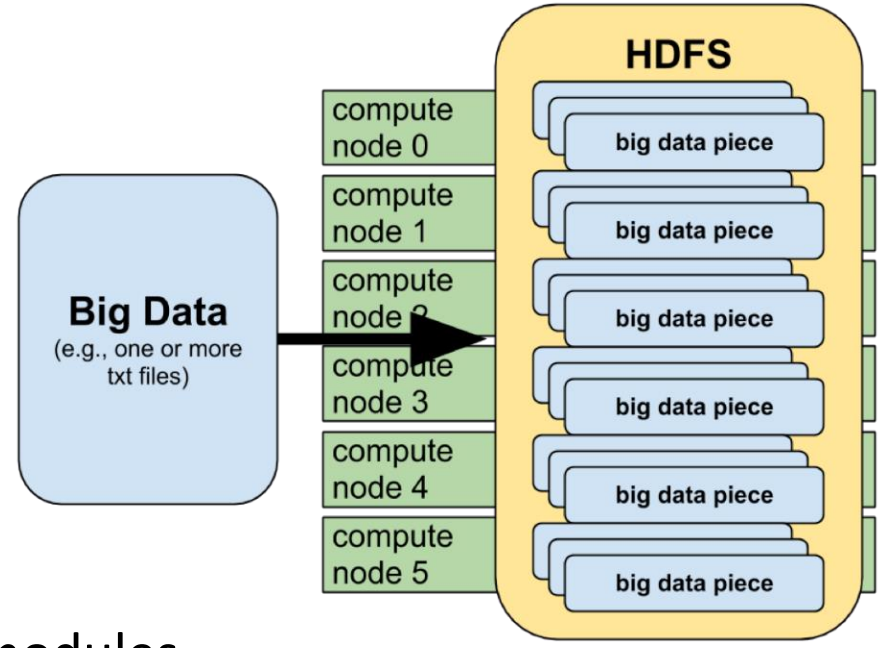
- Nhu cầu xử lý DLL tăng nhanh
- MapReduce
  - invented by Google in 2004 and used in Hadoop.
  - breaking the entire task into two parts: mappers and reducers.
  - mappers: read the data from HDFS, process it and generate some intermediate results.
  - reducers: aggregate the intermediate results to generate the final output.
- Key Limitations
  - inefficiency in running iterative algorithms.
  - Mappers read the same data again and again from the disk.



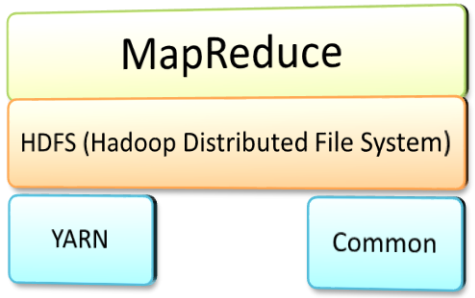
Data **parallelization** | **Distributed** execution | Outcome **aggregation**

Source: https://www.supinfo.com/articles/single/2807-introduction-to-the-mapreduce-life-cycle

- Apache Hadoop:
  - an open source framework for storing and processing large datasets using clusters of commodity hardware
  - highly fault tolerant
  - scaling up to 100s or 1000s of nodes



- Hadoop components:
  - **Common**: utilities that support the other Hadoop modules
  - **YARN**: a framework for job scheduling and cluster resource management.
  - **HDFS**: a distributed file system
  - **MapReduce**: computation model for parallel processing of large datasets.



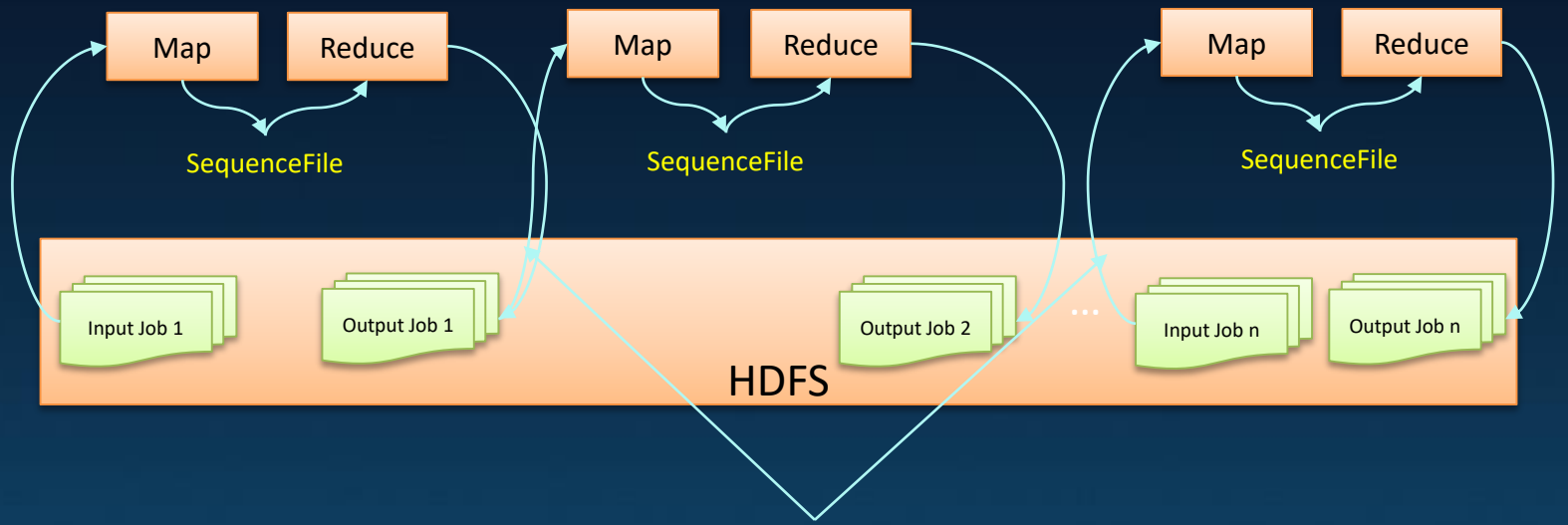
- Key limitation: not suitable for iterative-convergent algorithm!

Source: opensource.com

Platform for Big Data

Hadoop có phù hợp với các giải thuật lặp lại?

- Limitations of MapReduce
  - inefficiency in running iterative algorithms
  - Mappers read the same data again and again from the disk

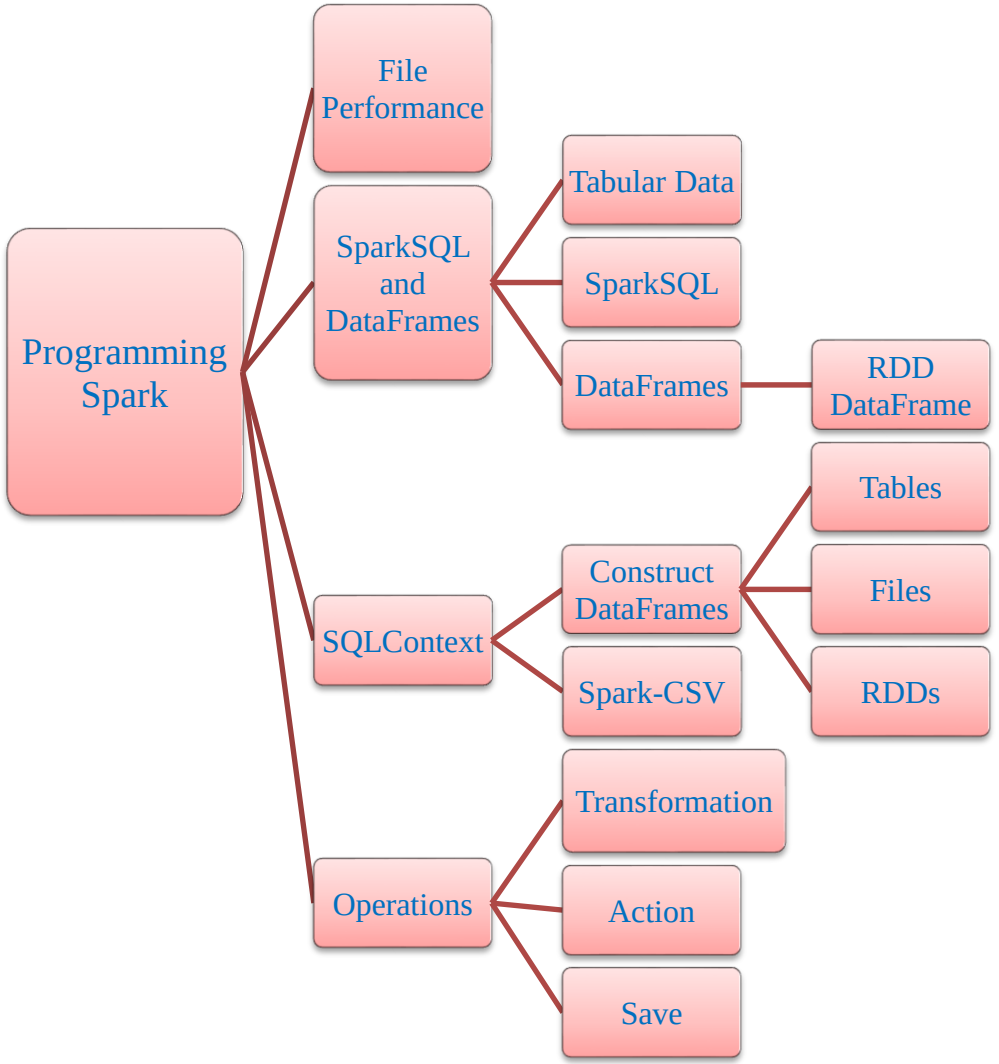


File access overhead!!!

How to reduce this latency? → memory caching

Hadoop có phù hợp với các giải thuật lặp lại? Không thật sự phù hợp

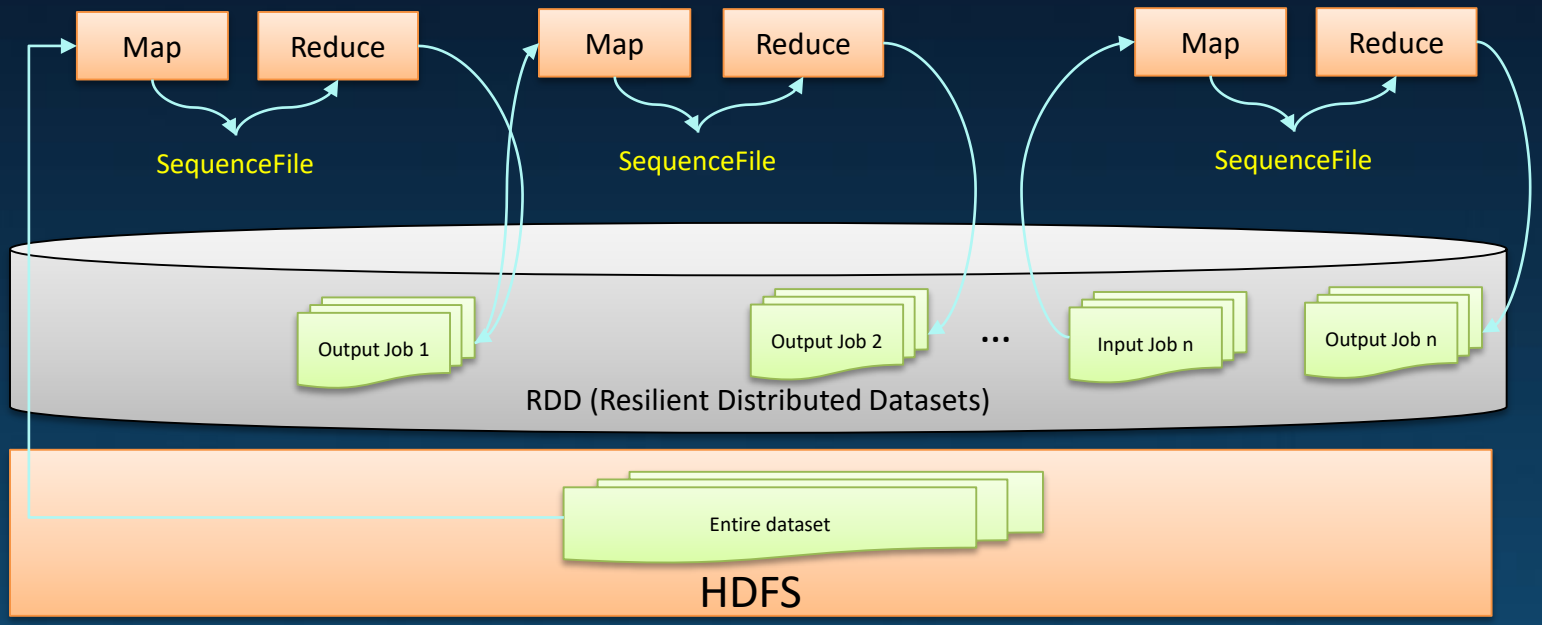
- Key motivation: suitable for iterative-convergent algorithms!
- Spark key features:
  - Resilient Distributed Datasets (RDD)
    - Read-only, partitioned collection of records distributed across cluster, stored in memory or disk.
    - Data processing = graph of transforms where nodes = RDDs and edges = transforms.
  - Benefits:
    - Fault tolerant: highly resilient due to RDDs
    - Cacheable: store some RDDs in RAM, hence faster than Hadoop MR for iteration.
    - Support MapReduce.

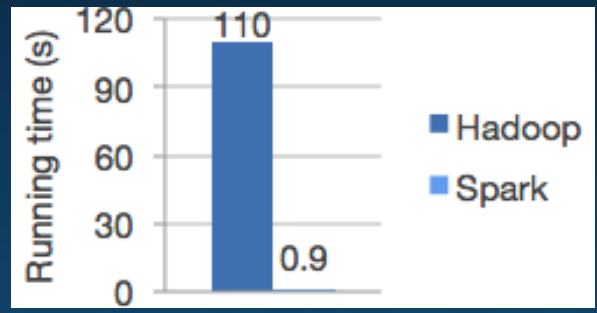


Platform for Big Data

Apache Spark thích hợp cho tính toán lặp lại

- Apache Spark
  - an alternative to Hadoop and MapReduce
  - perform in-memory computations



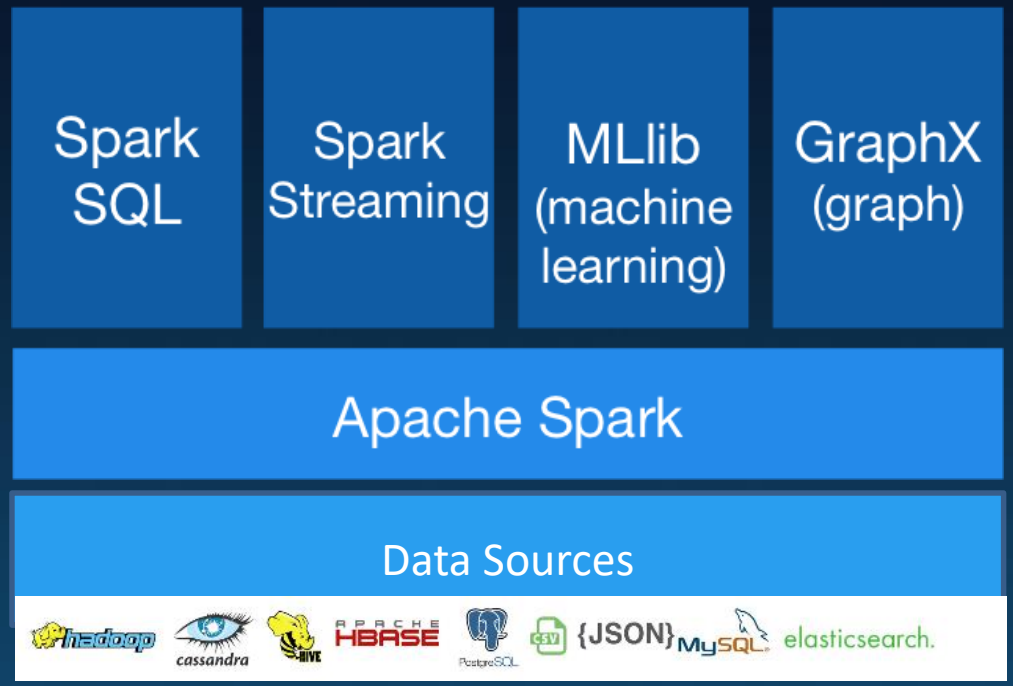


Apache Spark thích hợp cho tính toán lặp lại!

Apache Spark

Các thành phần của Apache Spark

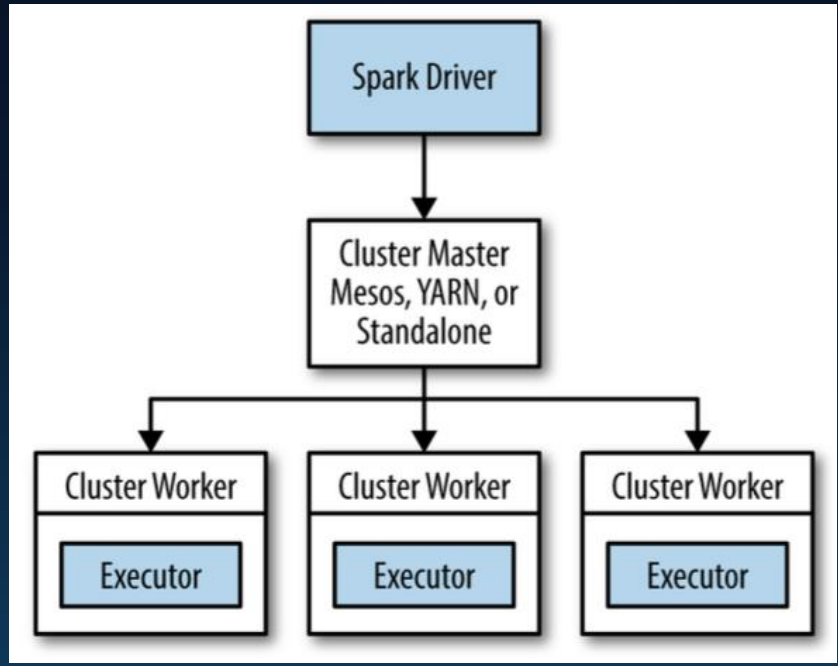
- Spark Core includes
  - components for task scheduling, memory management, fault recovery, interacting with storage systems
  - the API for resilient distributed datasets (RDDs)
- Spark SQL
  - package for working with structured data
- Spark Streaming
  - component for processing of live streams of data
- MLlib
  - library containing common machine learning (ML) functionality
- GraphX
  - library for manipulating graphs and performing graph-parallel computations



Apache Spark

Kiến trúc thực thi của Spark

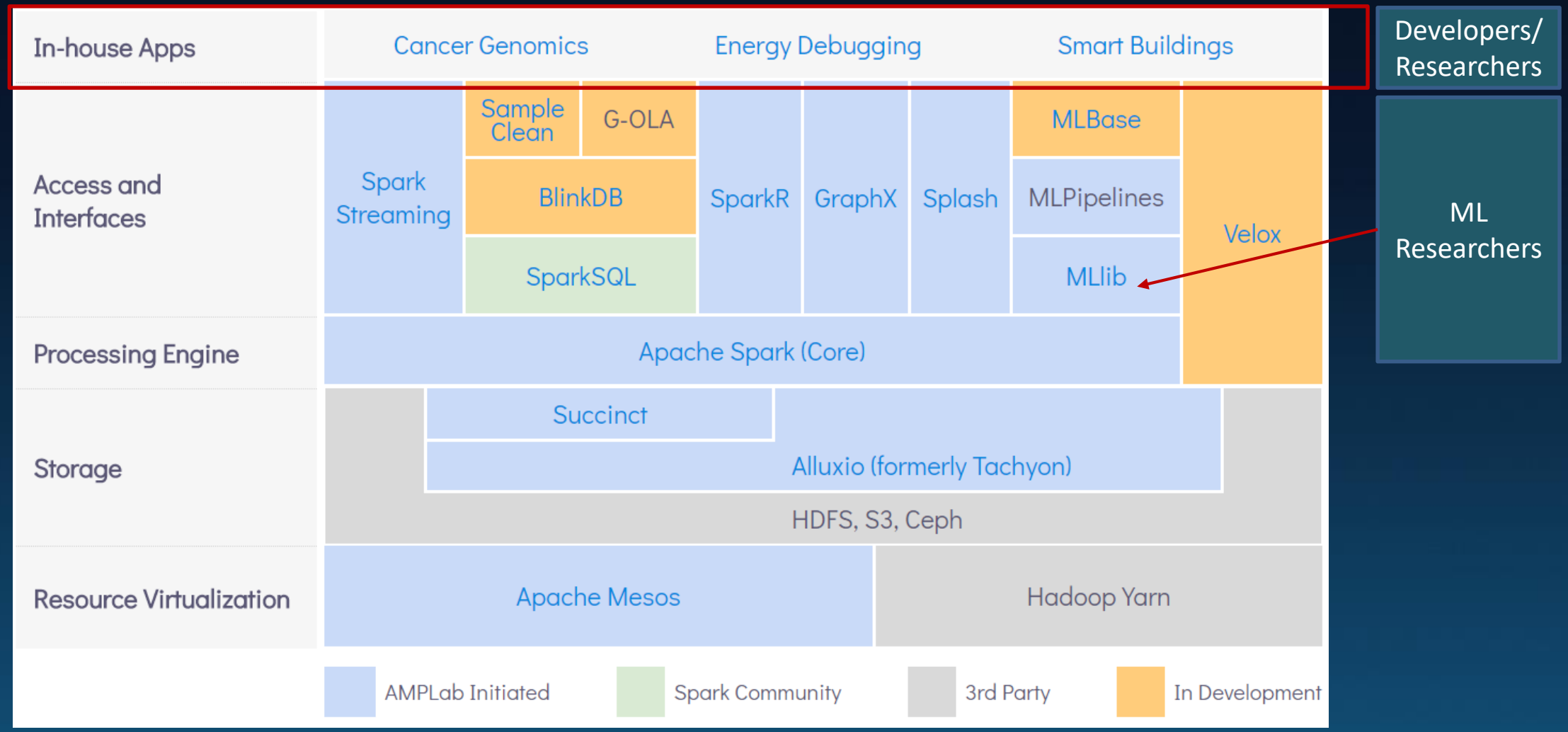
- In distributed mode, Spark uses a master/slave architecture
  - one central coordinator (**driver**) and
  - many distributed workers (**executors**)



- Spark Application contains
  - The driver runs in its own Java process
  - Each executor is a separate Java process

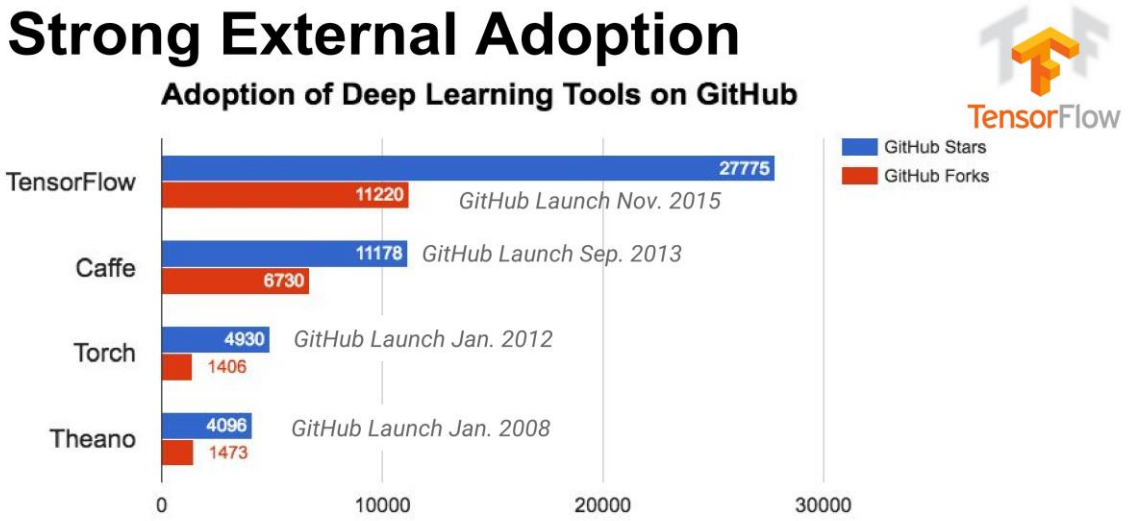
Ví dụ: BDAS, Berkeley Data Analytics Stack

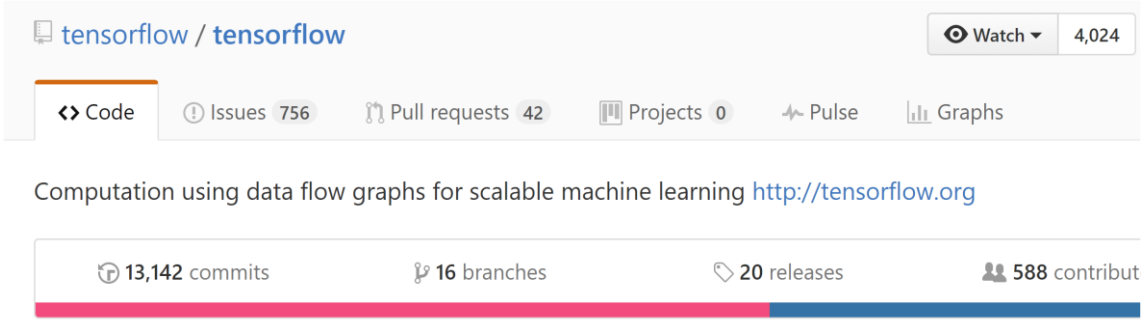
Bộ công cụ tích hợp cho phân tích dữ liệu lớn



[Nguồn: https://amplab.cs.berkeley.edu/software/]

- TensorFlow
  - open-source framework for deep learning, developed by the GoogleBrain team.
  - provides primitives for defining functions on tensors and automatically computing their derivatives.

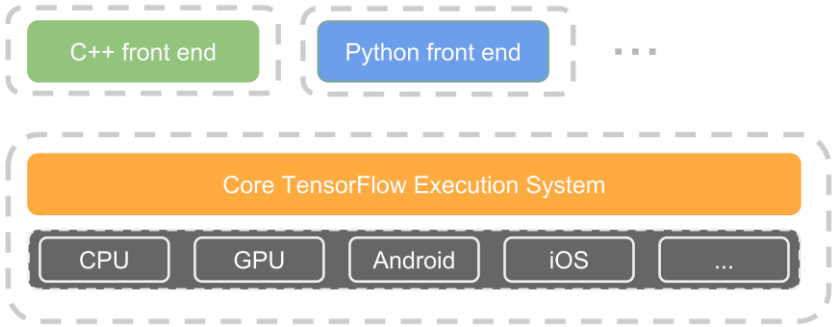




Công nghệ TensorFlow

Parallel Processing with TensorFlow

- Back-end in C++: very low overhead
- Front-end in Python or C++: friendly programming language



- … and even in more platforms

Linux CPU	Linux GPU	Mac OS CPU	Windows CPU	Android
build passing	build passing	build passing	build passing	build passing

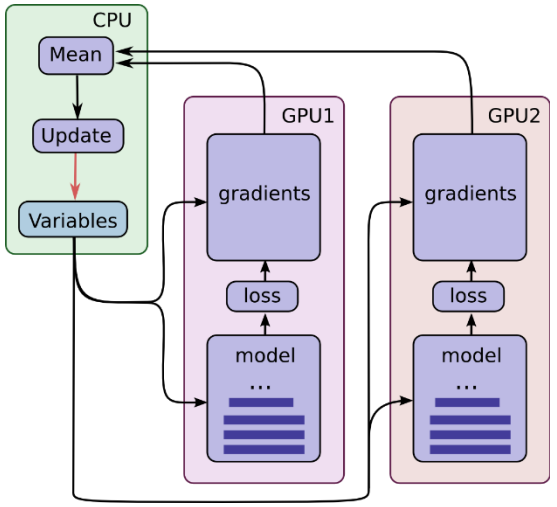
- Switchable between CPUs and GPUs

```
import tensorflow as tf
with tf.device('/cpu:'):
    a = tf.constant(1.0)
```

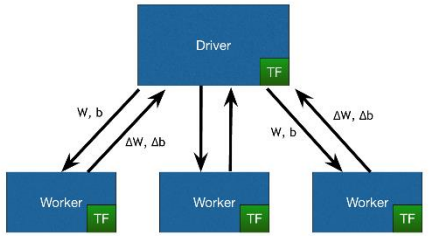
⟺

```
import tensorflow as tf
with tf.device('/gpu:0'):
    a = tf.constant(1.0)
```

- Multiple GPUs in one machine or distributed over multiple machines




```
import tensorflow as tf
with tf.device('/gpu:0'):
    a = tf.constant(1.0)
with tf.device('/gpu:1'):
    b = tf.constant(2.1)
```



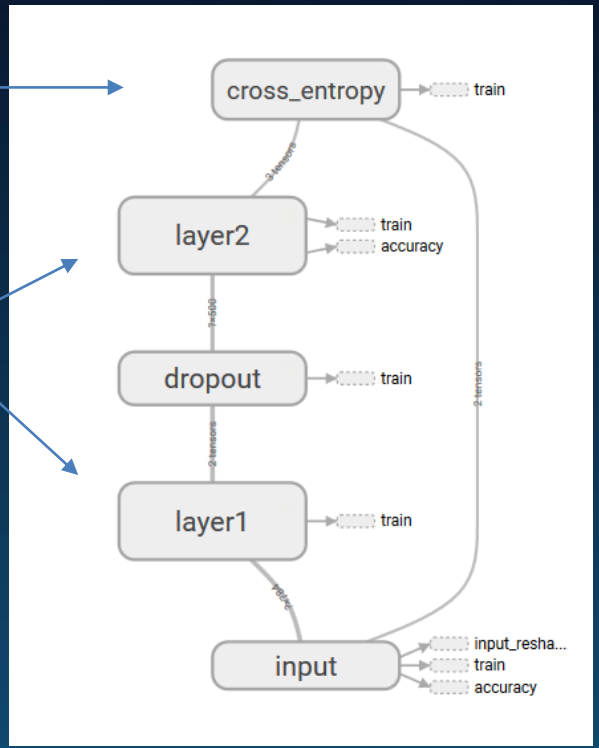
Công nghệ TensorFlow

TensorFlow– Đứng từ quan điểm nghiên cứu viên

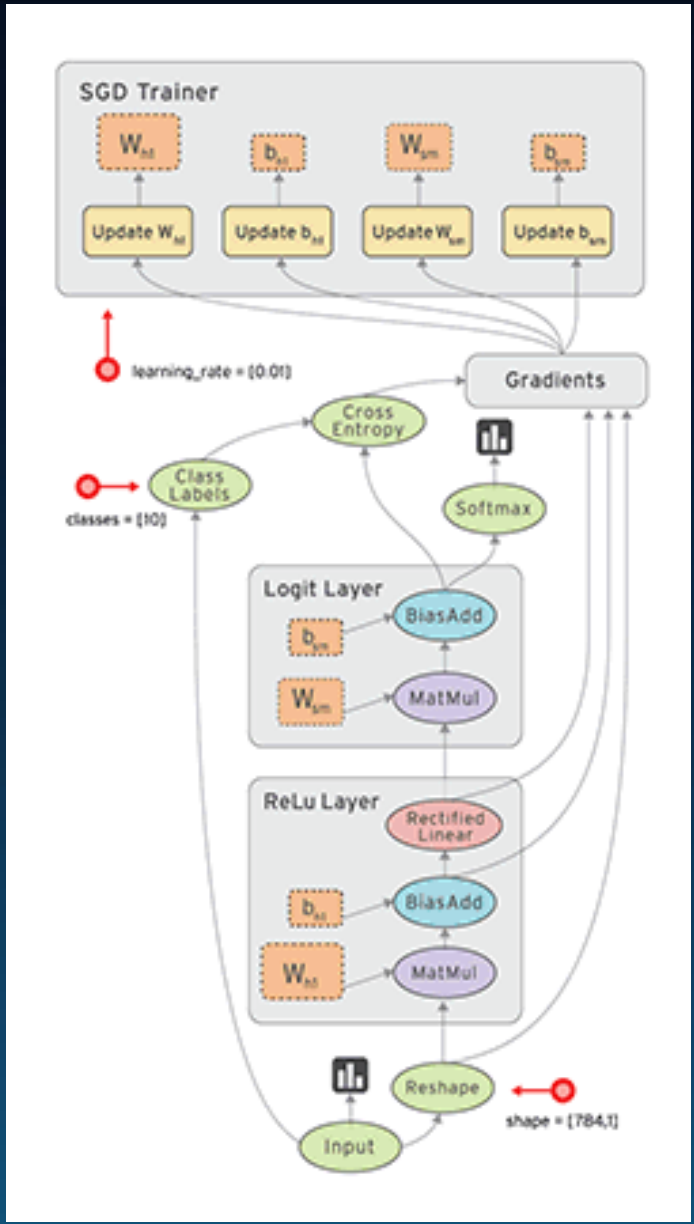
- Computational graph as dataflow graph
  - Construction phase: assemble a graph to represent the design, and to train the model
  - Execution phase: repeatedly execute a set of training ops in the graph

Loss function

Hidden layers



2-layer Neural Net



- Flexible, intuitive construction
- Launch the graph and run the training ops in a loop

Công nghệ TensorFlow

Một ví dụ tính toán đạo hàm tự động với TensorFlow

- The most attractive feature: Automatic Differentiation
- Automatically add ops to calculate symbolic gradients of variables w.r.t loss function, then pply these gradients with an optimization algorithm.

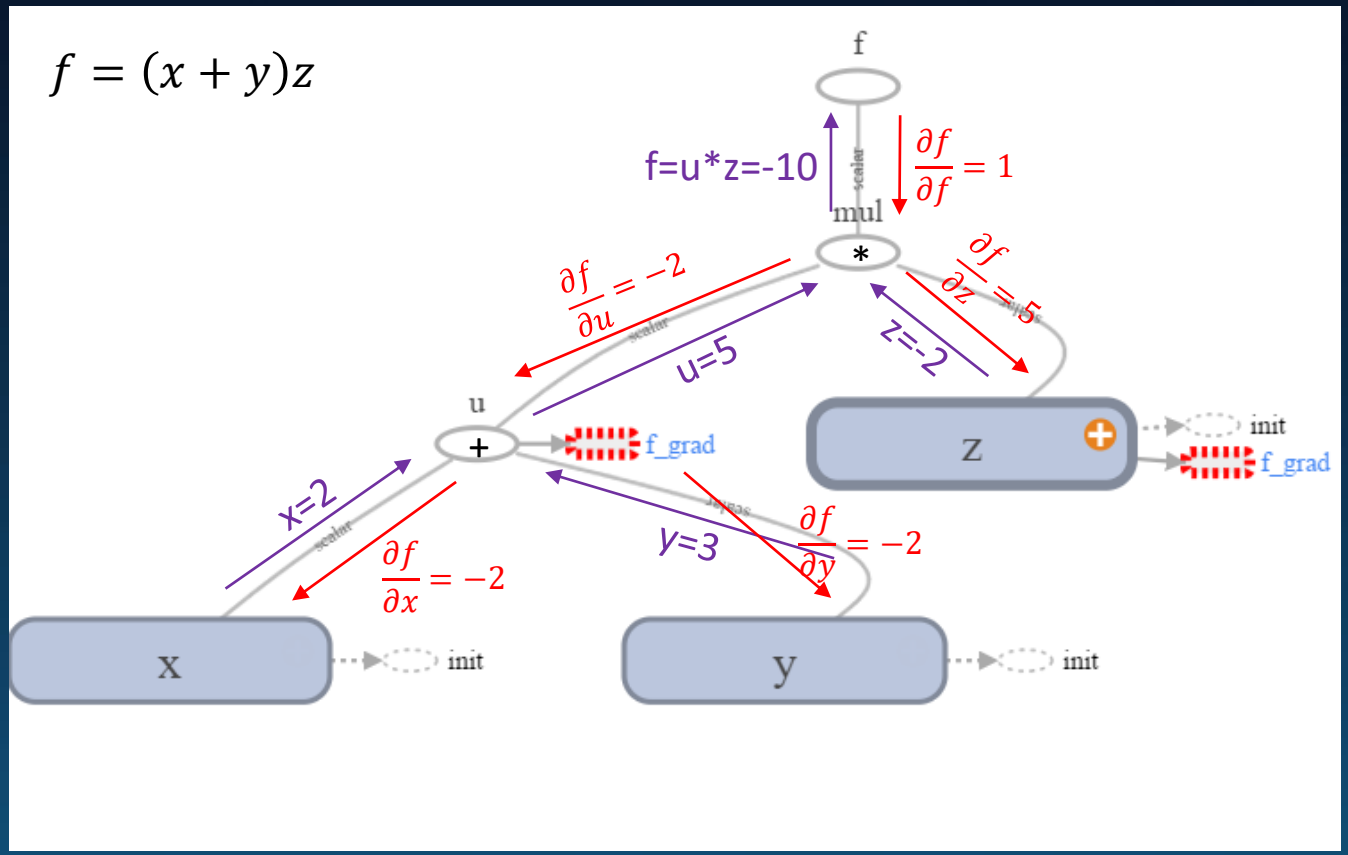
$x = 2$

$y = 3$

$u = x + y = 5$

$z = -2$

$f = u * z = -10$



$$\frac{\partial f}{\partial f} = 1$$

$$\frac{\partial f}{\partial u} = \frac{\partial f}{\partial f}\frac{\partial f}{\partial u} = 1 * z = -2$$

$$\frac{\partial f}{\partial z} = \frac{\partial f}{\partial f}\frac{\partial f}{\partial z} = 1 * u = 5$$

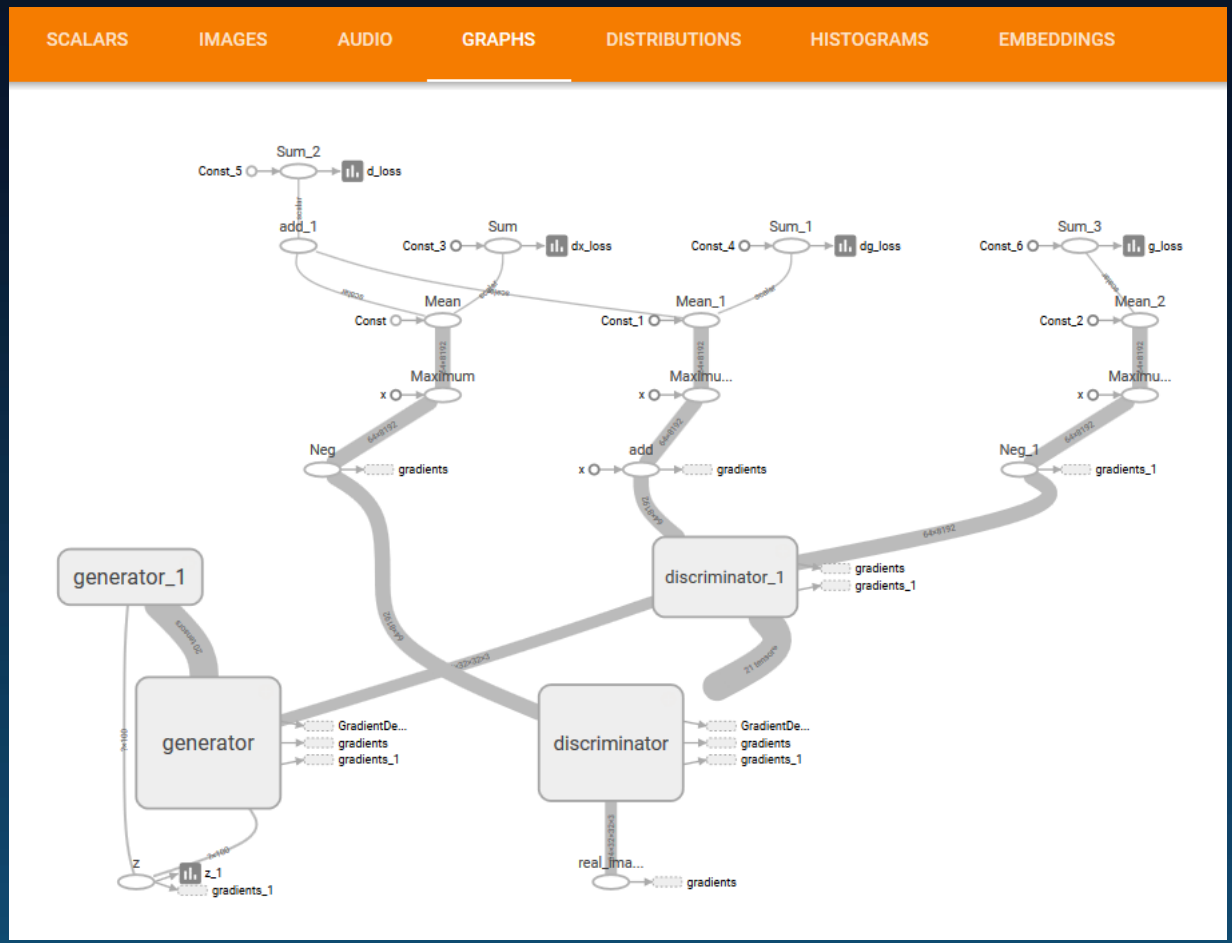
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial u}\frac{\partial u}{\partial x} = -2 * 1 = -2$$

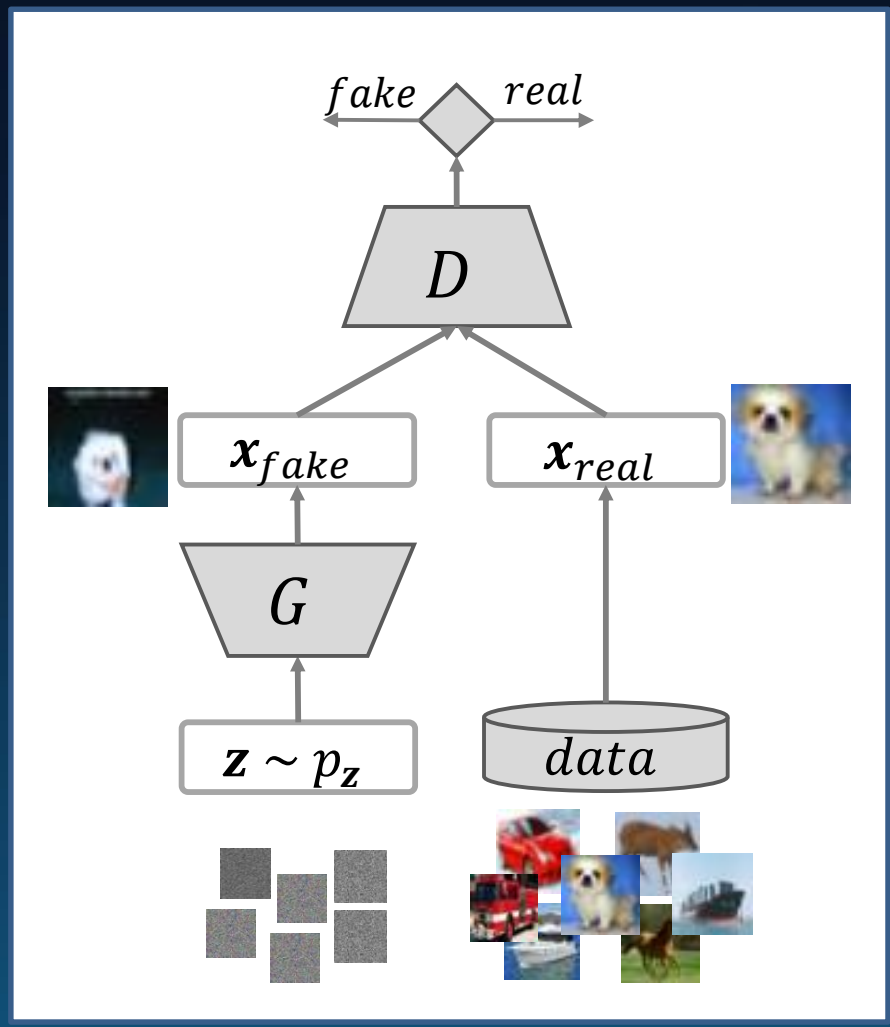
$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial u}\frac{\partial u}{\partial y} = -2 * 1 = -2$$

Công nghệ TensorFlow

TensorFlow– Một sốđồ thị tính toán cho mô hình nhiều tầng

- Computational graph of a Generative Adversarial Net (GAN)





Tổng kết

- Mô hình lớn cho dữ liệu lớn
  - Why traditional methods might fail on big data?
  - Three approaches to scale up big models
    - Scale up Gradient Descent Methods
    - Scale up Probabilistic Inference
    - Scale up Model Evaluation (bootstrap)
- Ba cách tiếp cận để phát triển mô hình lớn
  - Data augmentation
  - Stochastic Variational Inference for Graphical Models
  - Stochastic Gradient Descent and Online Learning
- Công nghệ nào cho mô hình lớn?
  - Hadoop, Spark, TensorFlow

Tài liệu tham khảo

Acknowledgement and References

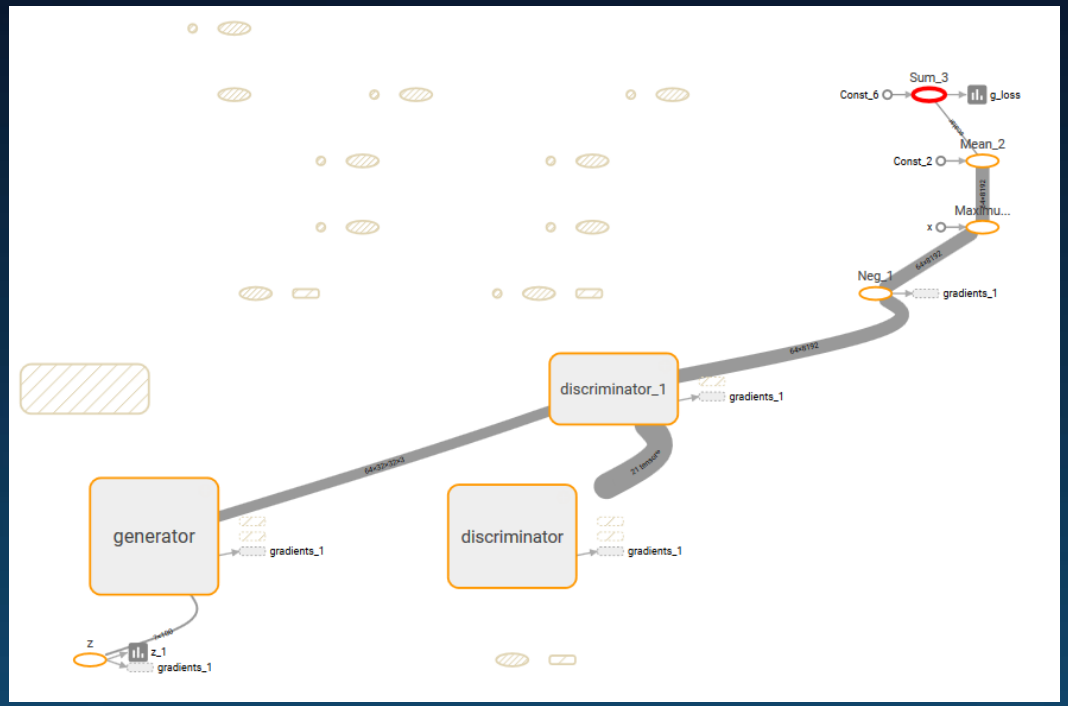
- Một số hình ảnh minh họa được download từ images.google.com theo cài đặc mặc định của search engine này.
- Eric Xing and Qirong Ho, *A New Look at the System Algorithm and Theory Foundations of Distributed Machine Learning*, KDD Tutorial 2015.
- Michael Jordan, *On the Computational and Statistical Interface and "Big Data"*, ICML Keynote Speech, 2014.
- Feng Niu, Benjamin Recht, Christopher Ré, Stephen J. Wright, *HOGWILD!: A Lock-Free Approach to Parallelizing Stochastic Gradient Descent*, CoRR abs/1106.5730, 2011.
- Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B. Gibbons, Garth A. Gibson, Gregory R. Ganger, Eric P. Xing, *More Effective Distributed ML via a Stale Synchronous Parallel Parameter Server*, NIPS 2013: 1223-1231.
- Wei Dai, Abhimanu Kumar, Jinliang Wei, Qirong Ho, Garth A. Gibson, Eric P. Xing, *High-Performance Distributed ML at Scale through Parameter Server Consistency Models*. AAAI 2015: 79-87.
- Salomon Bochner, *Lectures on Fourier Integrals.(AM-42)*. Vol. 42. Princeton University Press, 2016.
- Zhuang Wang, Koby Crammer, Slobodan Vucetic, *Breaking the curse of kernelization: budgeted stochastic gradient descent for large-scale SVM training*, Journal of Machine Learning Research 13: 3103-3131, 2012.
- Jing Lu, Steven C. H. Hoi, Jialei Wang, Peilin Zhao, Zhi-Yong Liu, *Large Scale Online Kernel Learning*, Journal of Machine Learning Research (JMLR) 17(47) 1-43 2016.

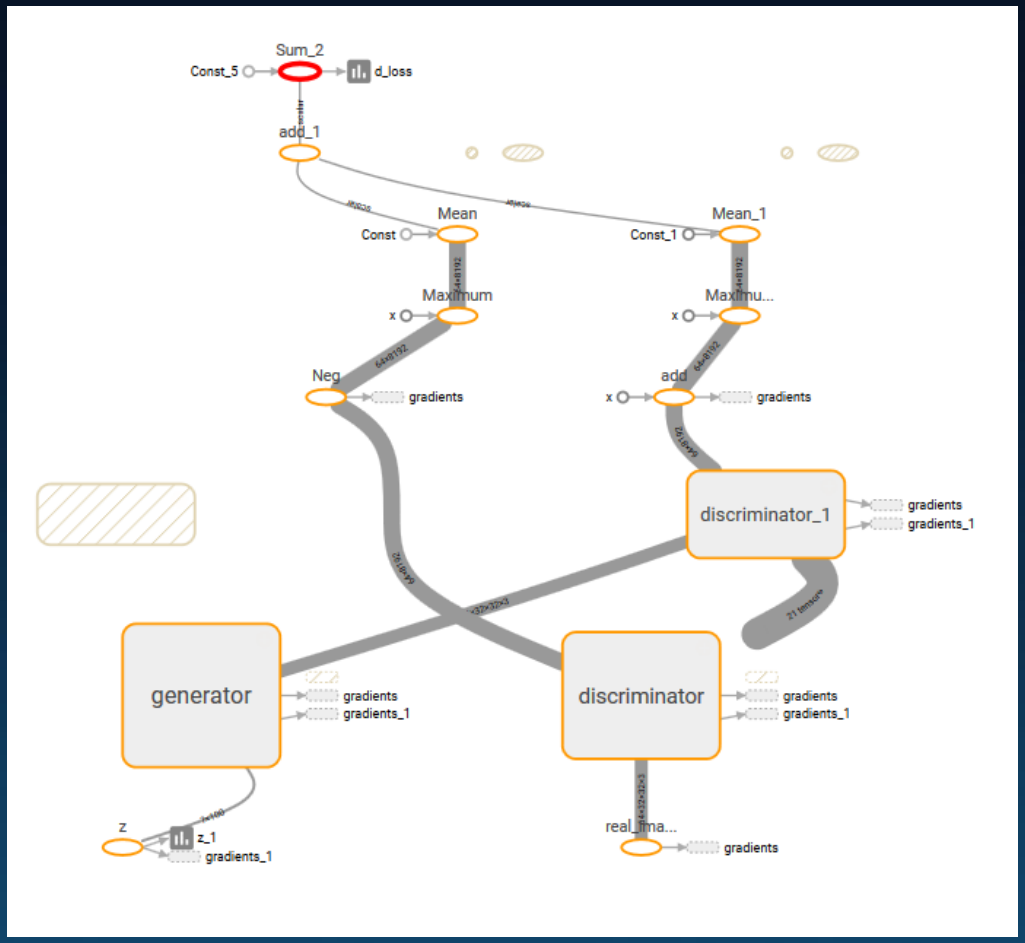
Xin cảm ơn các anh, chị đã lắng nghe!

Công nghệ TensorFlow

Debug với Tensorflow

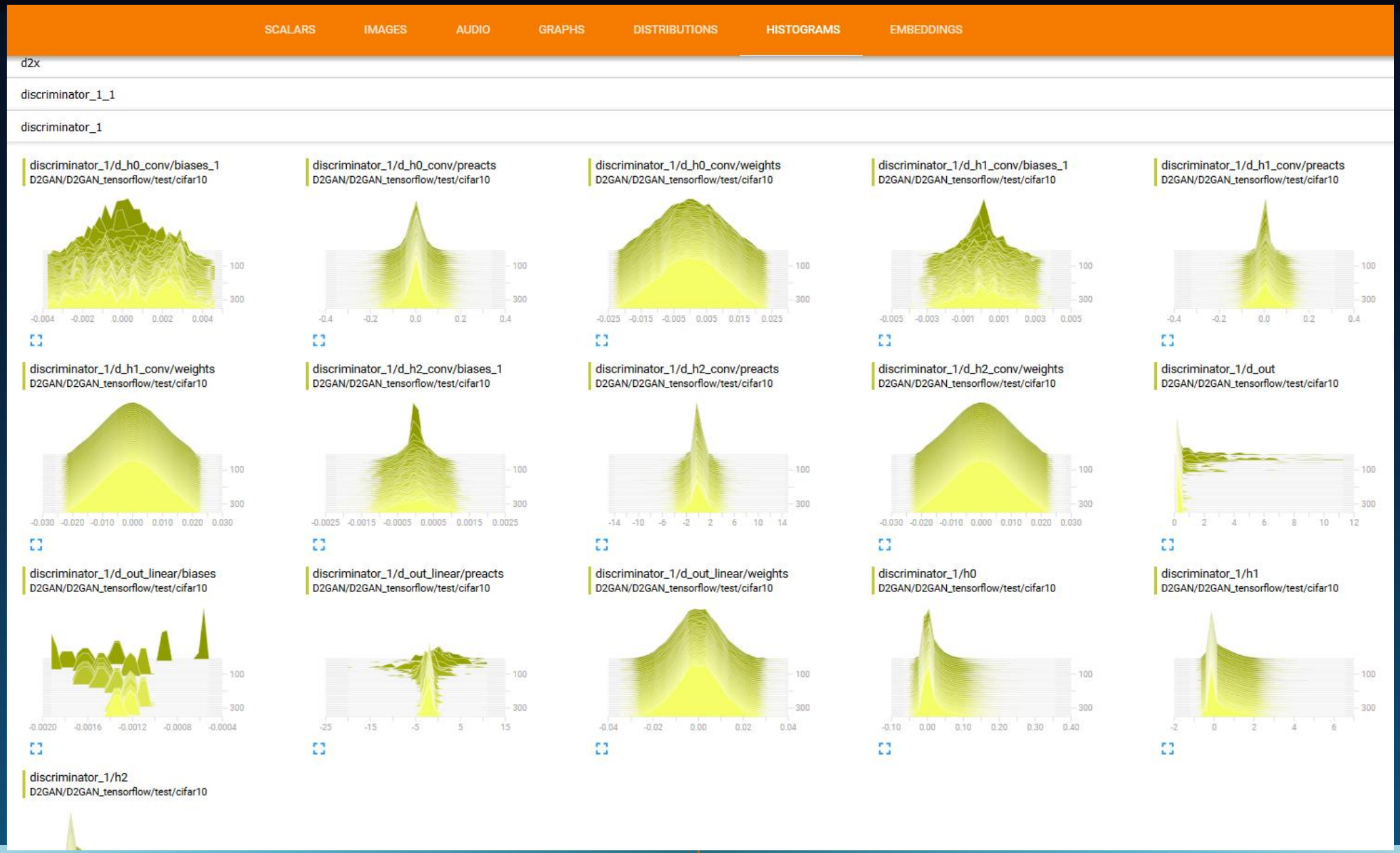
- Computational Graph
  - Trace connections and input





Công nghệ TensorFlow

Debug với Tensorflow



Debug với Tensorflow