OPEN NETWORKING SUMMIT 2016
MARCH 14-17, 2016 | SANTA CLARA, CA

MIND: Machine Learning based Network Dynamics

Dr. Yanhui Geng
Huawei Noah's Ark Lab, Hong Kong

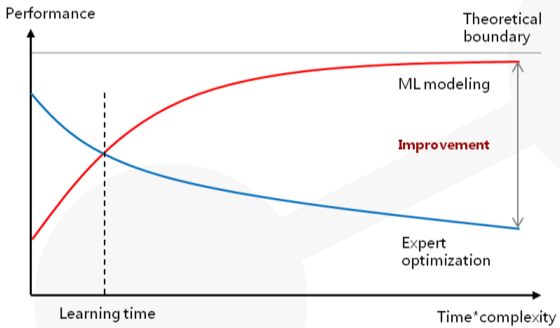
Outline

- **Challenges with traditional SDN**
- MIND architecture
- Experiment results
- Conclusion

Challenges with Traditional SDN

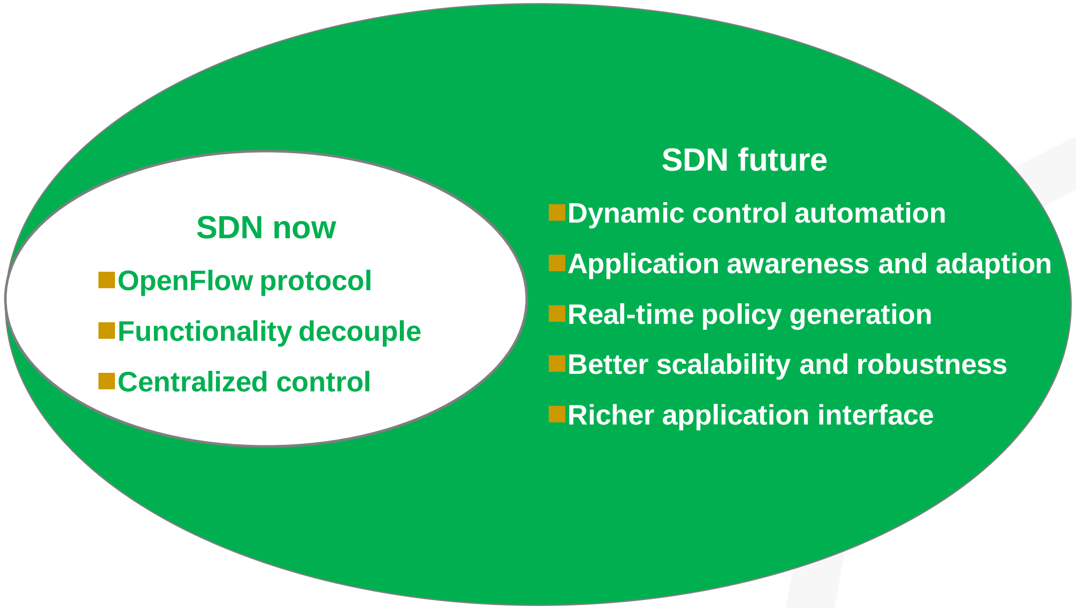
Challenges from network scale and traffic volume

- **Node #**: 1000→100,000→100,000,000 (IoT)
- **Traffic (flow/s)** : 10,000→1,000,000→1,000,000,000
- **Type of application**: 10→100→1,000 (multi-tenant)
- **Constrains**: 10→100→1,000 (dynamic)



	Control	Learning
Foundation	Causal analysis (white box)	Data association (black box)
Output	Deterministic	Stochastic
Advantage	Better interpretability, good performance under consistent scenarios	Balance between optimality and exploration, self-learning and evolving, better scalability and robustness
Limitation	Difficult to adapt when there are changes with scenarios or applications, poor scalability	Policies may be not optimal (poor ones with small probability), weak interpretability

Challenges with Traditional SDN



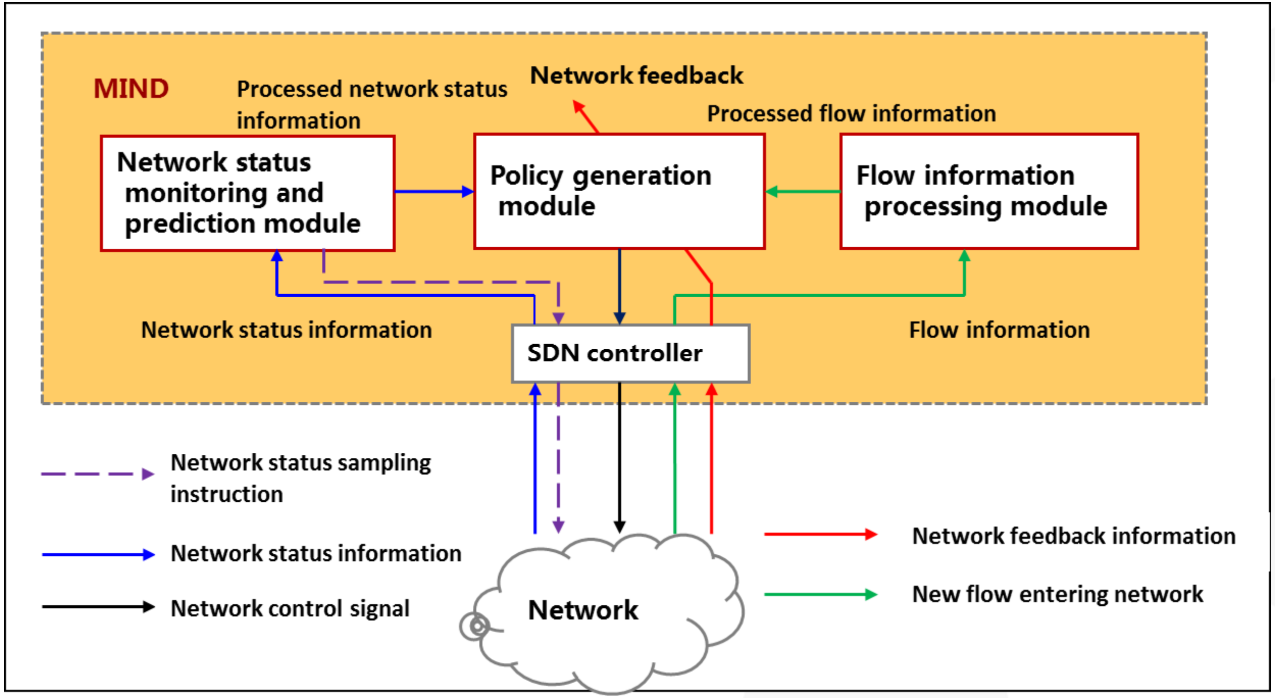
◆ The evolution **from connectivity to intelligence** will be driven by introducing machine learning and big data analytics into traditional network control and optimization!

Outline

- Challenges with traditional SDN
- **MIND architecture**
- Experiment results
- Conclusion

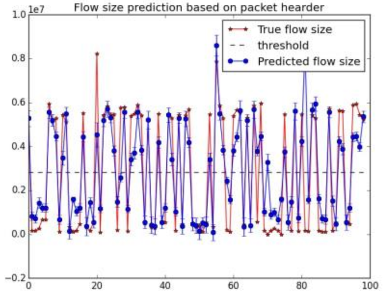
MIND architecture: automated control strategy generation enabled by machine learning

Technical foundation: reinforcement learning + data mining + optimization



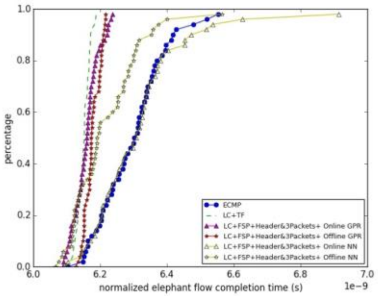
OPEN NETWORKING SUMMIT 2016
MARCH 14-17, 2016 | SANTA CLARA, CA

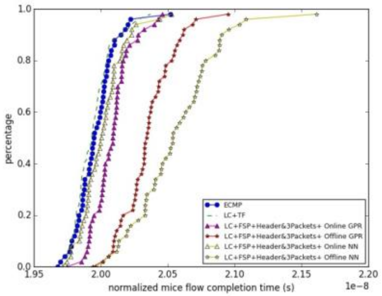
Least Congested Routing based on Elephant Flow Prediction



Performance:
TPR: 0.9787
(percentage of elephant flows correctly predicted)

FPR: 0.0577
(percentage of mice flows mistakenly predicted)





Performance:
99-th percentile of the average normalized elephant flow completion time improves over 10% compared to ECMP.

Key Contributions :
◆Propose flow size prediction based on packet header.
◆Employ Gaussian Process Regression to train the prediction model.
◆ Develop an online GPR algorithm.
◆ Develop Least Congested Routing Algorithm based on Elephant Flow Prediction.

Highlight 1: Flow size prediction based on pattern similarities

$$f_* \mid \mathbf{X}, \mathbf{y}, \mathbf{x}_* \sim N(\bar{f}_*, \mathrm{cov}(f_*))$$

$$\bar{f}_* = E[f_* \mid \mathbf{X}, \mathbf{y}, \mathbf{x}_*] = K(\mathbf{x}_*, \mathbf{X})\underbrace{[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1}\mathbf{y}}_{\boldsymbol{\alpha}}$$

$$\mathrm{cov}(f_*) = K(\mathbf{x}_*, \mathbf{x}_*) - K(\mathbf{x}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} K(\mathbf{X}, \mathbf{x}_*)$$

$$\bar{f}_{SR}(\mathbf{x}_*) = \mathbf{k}_m(\mathbf{x}_*)^T \boldsymbol{\alpha}_{n+1}$$

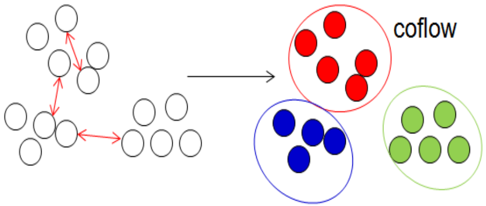
Highlight 2: Online Learning

$$\boldsymbol{\alpha}_{n+1} = \left(\mathbf{I} - \frac{\boldsymbol{\Pi}_n \mathbf{k}_{n+1} \mathbf{k}_{n+1}^T}{1 + \mathbf{k}_{n+1}^T \boldsymbol{\Pi}_n \mathbf{k}_{n+1}}\right)\boldsymbol{\alpha}_n + y_{n+1}\left(\mathbf{I} - \frac{\boldsymbol{\Pi}_n \mathbf{k}_{n+1} \mathbf{k}_{n+1}^T}{1 + \mathbf{k}_{n+1}^T \boldsymbol{\Pi}_n \mathbf{k}_{n+1}}\right)\boldsymbol{\Pi}_n \mathbf{k}_{n+1}$$

Highlight 3: Least Congested Routing based on Elephant Flow Prediction

◆We use our develop algorithm to predict elephant flows and then route the predicted elephant flow to the least congested path and use ECMP to route mice flows

Online Coflow Identification



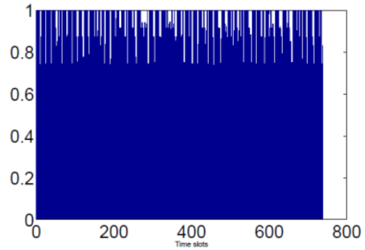


Fig1. Accuracy : three level features and weight matrix

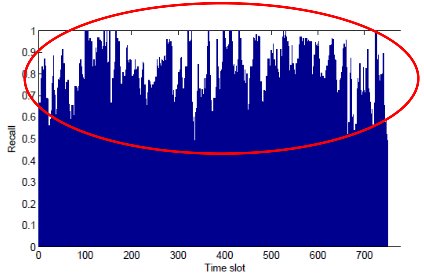


Fig2. Accuracy: without community detection

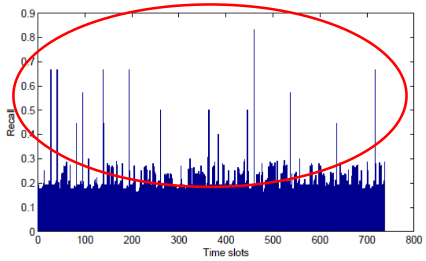


Fig3. Accuracy: without weight matrix

Key Contributions :

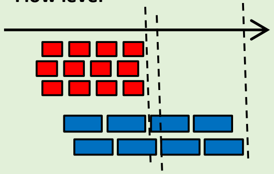
◆Develop a machine learning based method to identify coflows in network.

◆Investigate features in three different levels.

◆Develop a learning based method to determine weights for different features.

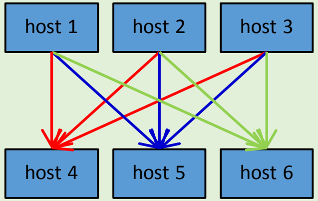
Highlight 1: Features in three levels

Flow level

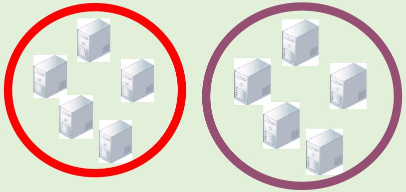


$$D_{flow}(i,j) = \begin{bmatrix} D_{time}(i,j) \\ D_{size}(i,j) \\ \vdots \\ D_{\text{int}}(i,j) \end{bmatrix}$$

Application level



Community level



$$D_{com}(i,j) = \begin{cases} 0 & \text{if } i \text{ and } j \in Cmnty \\ 1 & otherwise \end{cases}$$

$$D_{app}(i,j) = \begin{cases} 0 & \text{if } i \text{ and } j \in AGG \\ 1 - J(i,j) & otherwise \end{cases}$$

Highlight 2: Weight matrix

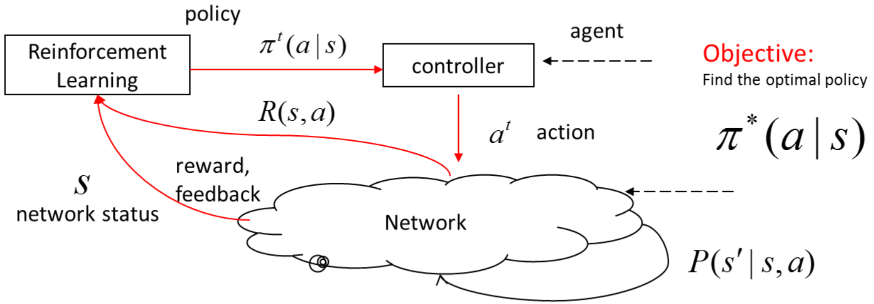
$$d_{ij} = \|D(i,j)\|_A = \sqrt{D(i,j)^T A\, D(i,j)}$$

$$A^* = \arg\min_A \sum_{(f_i,f_j)\in S} d_{ij} - \log\Big(\sum_{(f_i,f_j)\in D} d_{ij}\Big)$$

◆ learn optimal A from training data

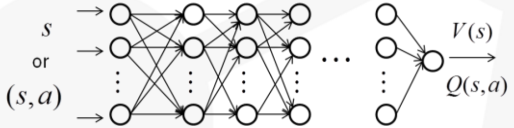
OPEN NETWORKING
SUMMIT 2016
MARCH 14-17, 2016 | SANTA CLARA, CA

Reinforcement Learning in SDNs for Routing



$$Q(s,a) \leftarrow Q(s,a) + \eta[R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)]$$

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$



$$\boldsymbol{w} = \boldsymbol{w} - \eta \cdot \left(\tilde{Q}(s_t, a, \boldsymbol{w}) - \left(R_{t+1} + \gamma \cdot \max_{a \in A} \tilde{Q}(s_{t+1}, a, \boldsymbol{w})\right)\right)$$

Idea：

◆We approximate the Q-function with a neural network .

◆We update the neural network using stochastic gradient descent.

◆Deep architectures can compactly represent functions that may need a very large shallow architecture.

◆We argue that SDNs are characterized by an ***inherent locality*** that we can exploit when designing a *deep architecture with local connectivity patterns*.

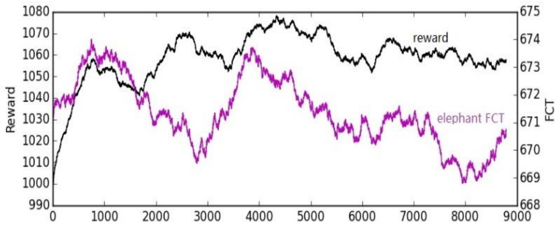
◆*Options*: stochastic RMSProp, prioritized experience replay, target network optimizations, double Q-learning.

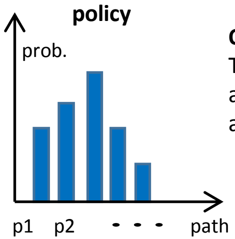
Advantages

Learn changing traffic patterns:
Adaptive approach.

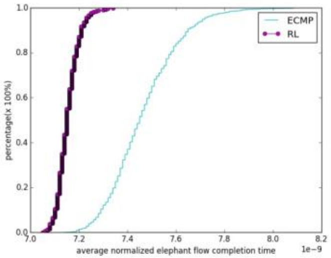
Tailored to the needs of network operator or applications:
All you need to do is define proper reward.

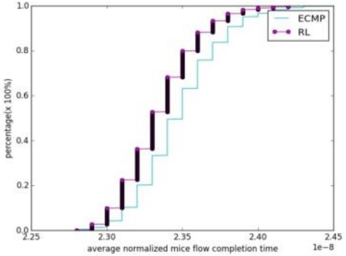
Routing by Online-REPS-RKHS





Observations:
The policy keeps improving as reward keeps increasing and FCT keeps decreasing.





Key Contributions:

◆Develop an online direct policy search algorithm based on the state-of-the art REPS-RKHS.
◆Investigation on the network control problem using Reinforcement learning.
◆Apply the Online-RKHS-REPS to network routing control.

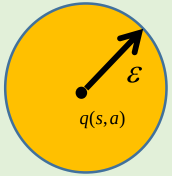
Highlight 1: Online-REPS-RKHS algorithm

$$\max_{\pi,\mu_\pi} J(\pi) = \max_{\pi,\mu_\pi} \iint_{S\times A} \pi(a\mid s)\mu_\pi(s) R_s^a \, dsda$$

$$s.t. \iint_{S\times A} \pi(a\mid s)\mu_\pi(s)\, dsda = 1$$

$$\iint_{S\times A} P(s'\mid s,a)\pi(s\mid a)\mu_\pi(s)\, dsda = \mu_\pi(s')$$

$$KL(\pi(s\mid a)\mu_\pi(s) \,\|\, q(s,a)) \le \varepsilon$$



Data Block	Sample 1	Sample 2	
π_0 →	π_1 →	π_2	. . .

Highlight 2: Online-REPS-RKHS algorithm to Network Routing Control

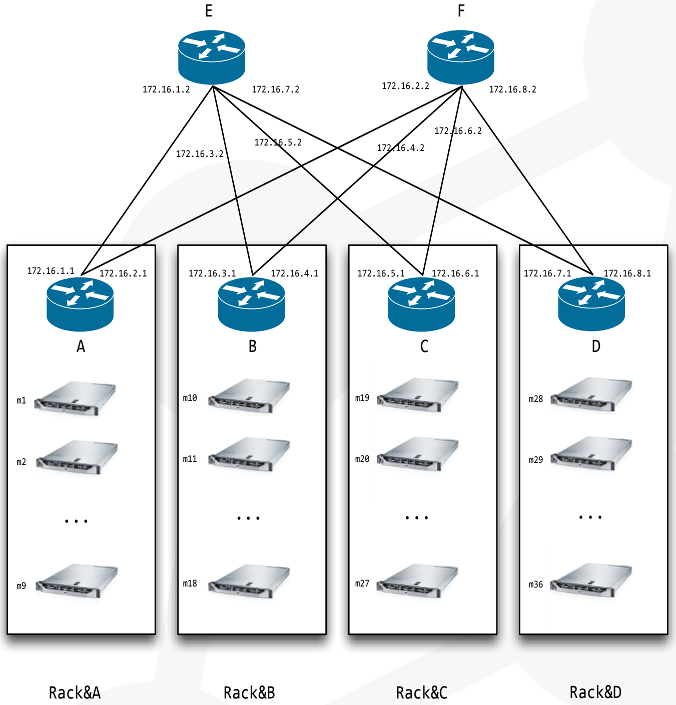
◆We use Online-REPS-RKHS algorithm to learn the probability distribution of choosing the top-k best path.

Outline

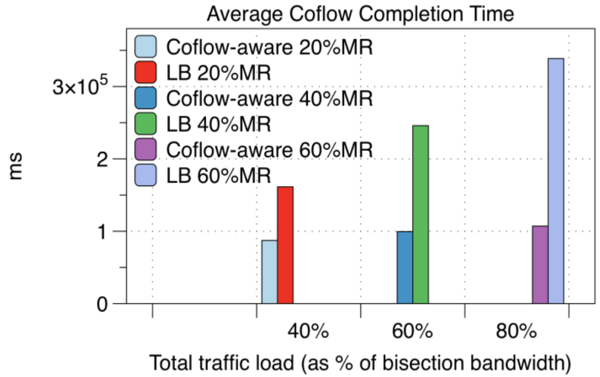
- Challenges with traditional SDN
- MIND architecture
- **Experiment results**
- Conclusion

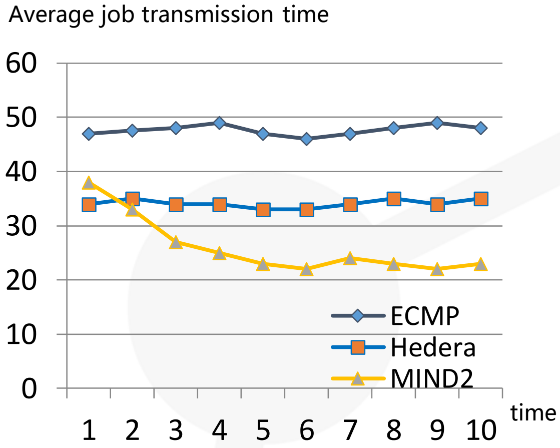
Experiments on test bed: configuration

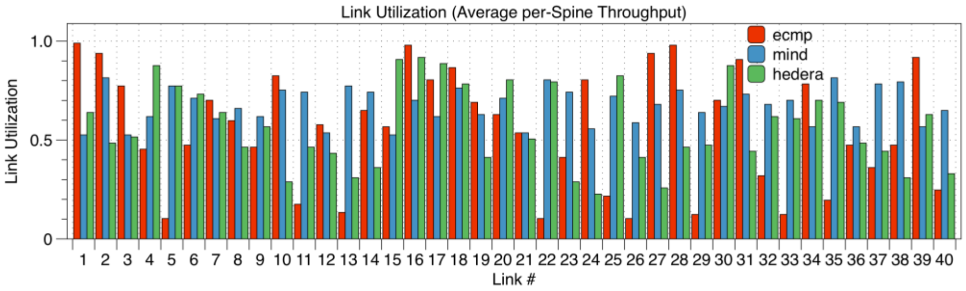
- **Scenarios**: DC with 800 virtual computation nodes, spine-leaf topology (40+80 SDN switches), 10G Ethernet connection
- **Application/Traffic**: real trace from Jiangsu Telecom and Facebook, 40% MR job, 30% storage, 20% short packets, 10% web query
- **Performance metric**: average job transmission time.



Experiments on test bed: results







Outline

- Challenges with traditional SDN
- MIND architecture
- Experiment results
- **Conclusion**

Conclusion

- Propose and implement data-driven network control architecture MIND;
- Online machine learning techniques for prediction/inference the spatial-temporal traffic information;
- Reinforcement learning for optimal routing strategy based on traffic data and network state;
- Experiment results with a real SDN test bed demonstrate the feasibility and effectiveness of the self-learning SDN paradigm;

Thank you!