On Optimal Partitioning and Scheduling of DNNs in Mobile Edge/Cloud Computing

Jie Wu
Dept. of Computer and Information Sciences
Temple University, USA

ICPADS 2020

Outline

1. On Problem Solving
2. Mobile Cloud Computing (MCC)
    - DNN inferencing
    - Offloading
3. Optimal Scheduling
4. Optimal Partition and Scheduling
5. Conclusions and Future Work
6. Some Reflections

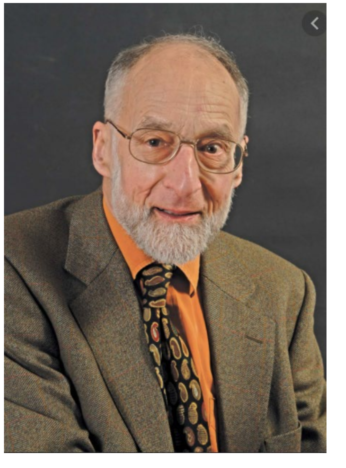
1. On Problem Solving

How to Solve It (Poyla, 1945)

If you can't solve a problem, then there is an easier problem you can solve: find it.

Is Computing An Experimental Science ? (Milner, 1986)

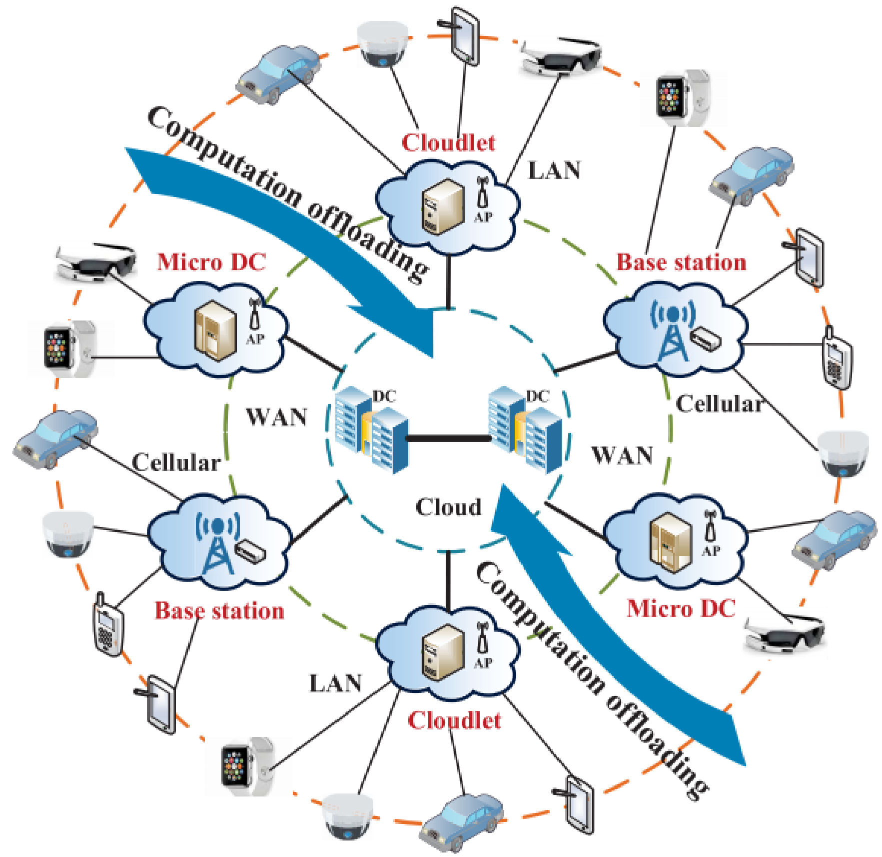
A theory can only emerge through protracted exposure to application.



Ideas and applications developed side-by-side

2. Mobile Cloud Computing (MCC)

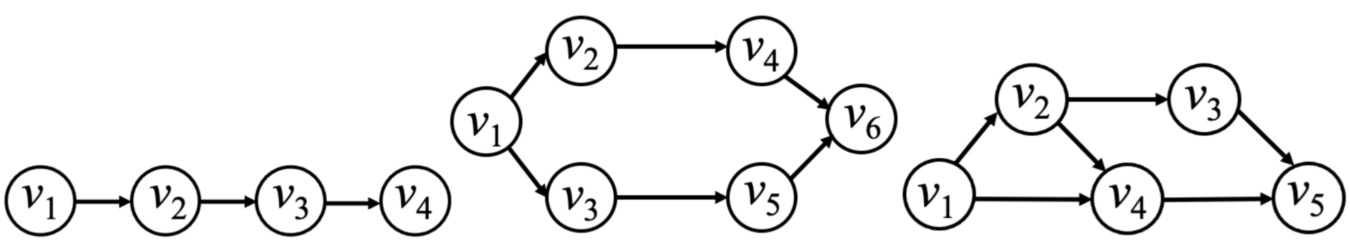
- Cloud/Edge Computing
  - Application-driven: VR/AR, video analytics using IoTs
  - Better QoE: cloud computing and mobile/edge device
  - Key indicators: latency, accuracy, energy, and privacy
  - Latency-sensitive: how to bring rich computational resources to mobile users?



50 billion IoT devices by 2020

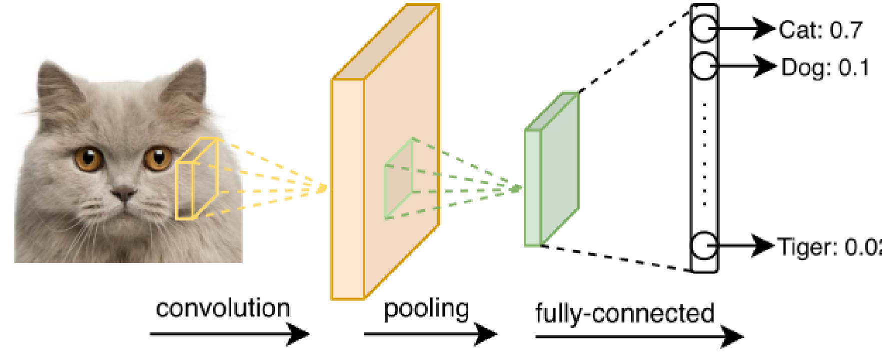
DNN Inferencing

- Deep Neural Networks (DNNs)
  - Technologies: GPU (graphic) and TPU (tensor)
- AI applications
  - Computer vision: AlexNet, VGG-16, Inception, RandWire
  - Natural language processing: GPT-3
- Graph models of DNNs

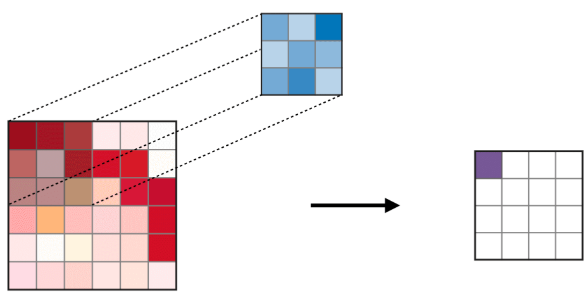


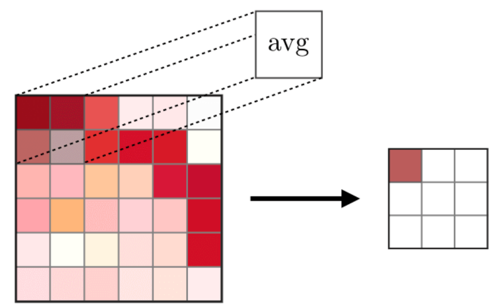
(a) line (b) multi-path (c) DAG

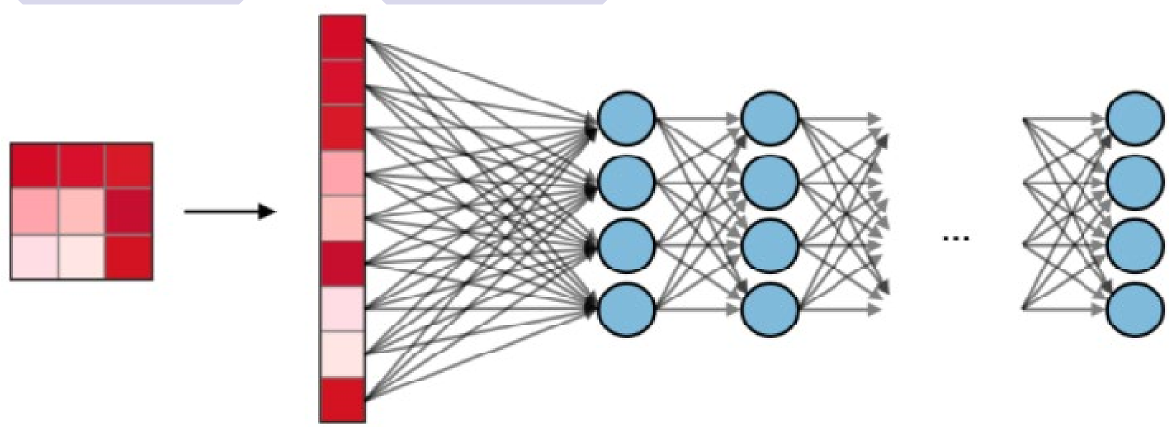
Convolution NNs



- CNNs (image classification)
- convolution (filtering), pooling (max/avg), fully-connected (neurons)

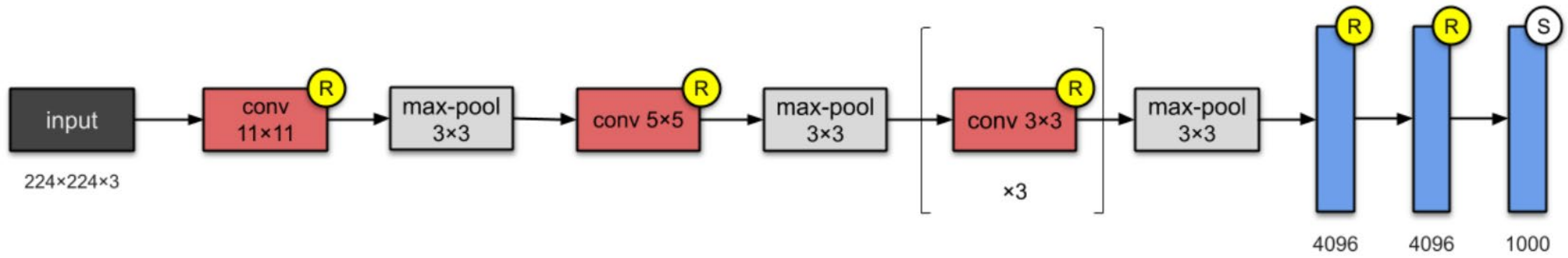




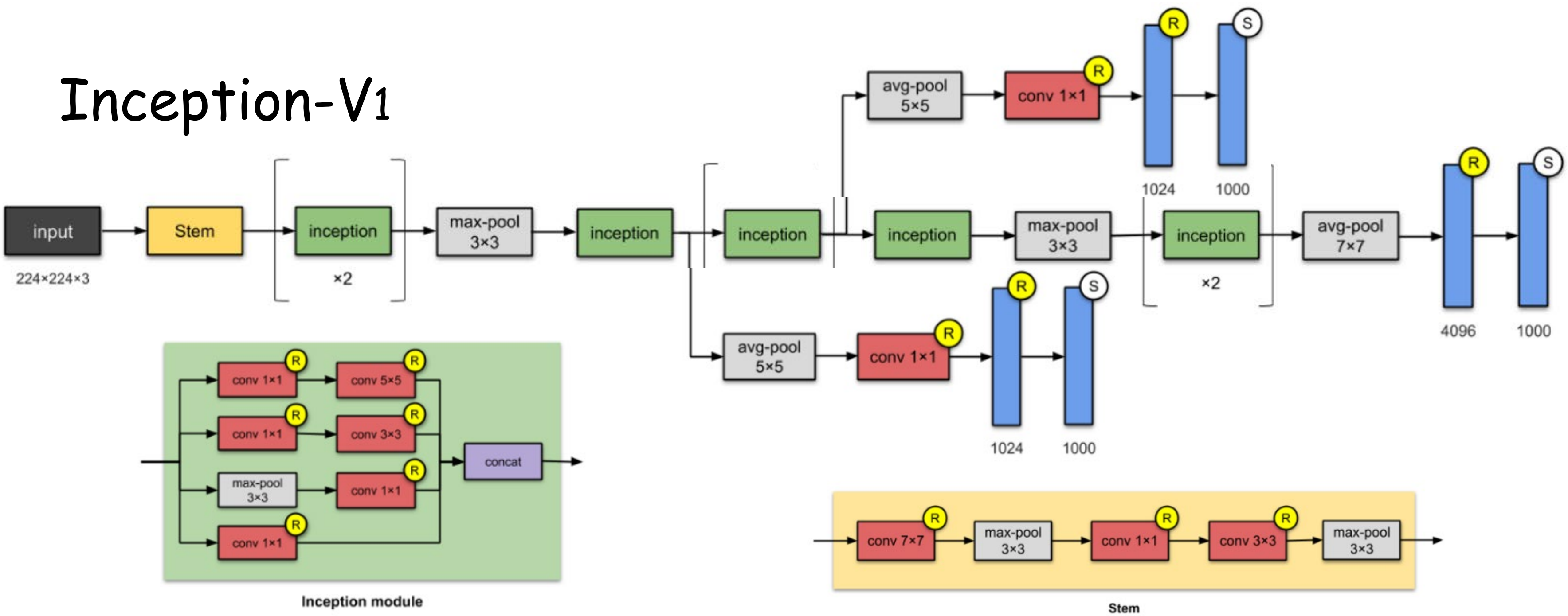


Sample CNNs

AlexNet (Red: CONV, Gray: POOL, Blue: FC)



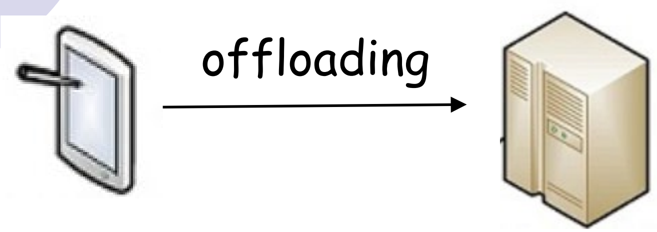
Inception-V1



Offloading

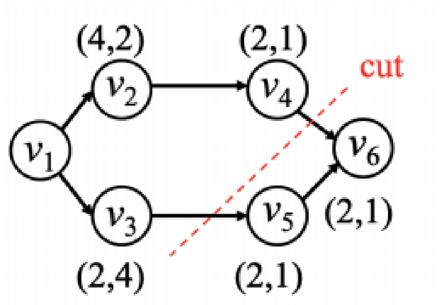
- Three-stage collaborative computation offloading
  - Local computation: processing on local devices
  - Communication: transmitting intermediate DNN layers' outputs
  - Remote computation: completing the remote processing in cloud

- Three models
  - On-device optimization
  - Cloud-only offloading
  - Mixed-mode offloading

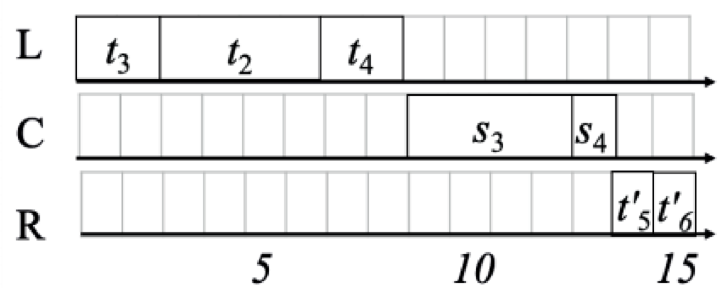


Offloading Samples

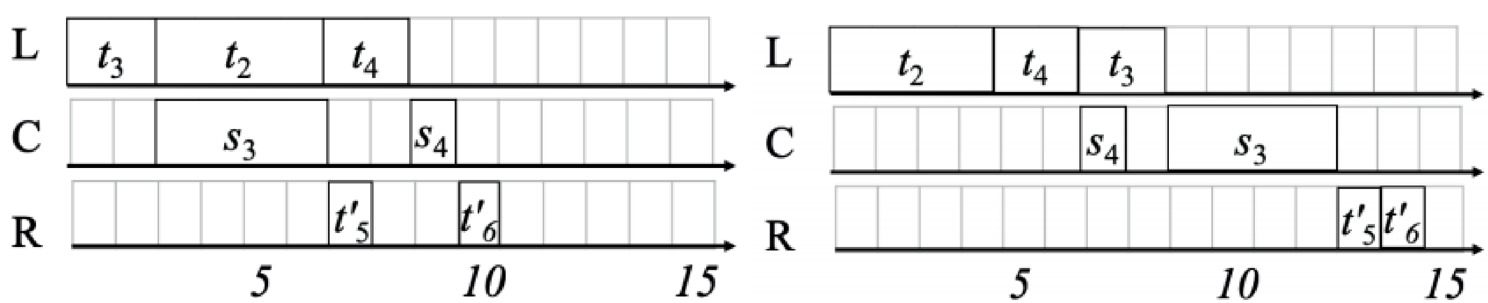
- Given a partition (i.e., cut)
  - Course-grained pipeline: local, comm, and remote
  - Fine-grained pipeline: path-based



(a) a DNN



(b) no fine-grained pipeline



(c) fine-grained pipeline

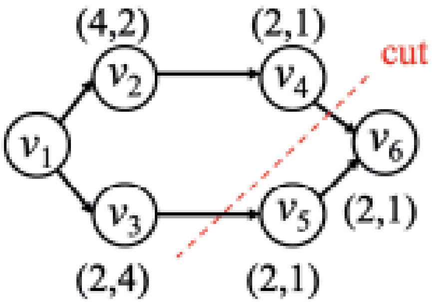
3. Optimal Scheduling

- DNN Computation Offloading Optimization (DCOO)
  - DCOO: optimal scheduling (in terms of minimum makespan) for a given partition (i.e., cut).
- Cases of DNN
  - Line-structure: trivial
  - Multi-path: hard
  - DAG: hard

Theorem 1: DCOO is NP-hard for a multi-path DNN.

Proof: Reduce 3-machine flow-shop to DCOO.

Multi-Path Scheduling



- Single-path
  - Straightforward solution, even without a given cut
- Multi-path
  - Path: a path from input to cut or from cut to output
  - Non-overlaps among paths (except input and output)
  - E.g., v_1-v_2-v_4, v_1-v_3, v_6, v_5-v_6

Theorem 2: In multi-path DNNs, the optimal schedule can be achieved via the non-preemptive path-based schedule.

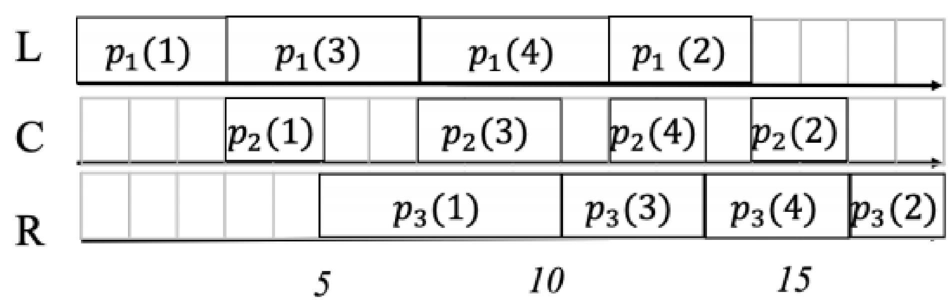
Extended Johnson Algorithm (EJA)

- Path p(i) in three stages
  - $P_1(i)$, $P_2(i)$, $P_3(i)$
- Linear solution (EJA)
  - Dividing paths into H and L
  - E.g., H = {1}, L = {3, 4, 2}

Algorithm 1 Extended Johnson Algorithm (EJA)

$H \leftarrow L \leftarrow \phi$
for $i = 1$ to m **do**
  if $p_1(i) + p_2(i) \leq p_2(i) + p_3(i)$ **then**
    $H = H \cup p(i)$
  else
    $L = L \cup p(i)$
Sort H increasingly based on $p_1(i) + p_2(i)$
Sort L decreasingly based on $p_2(i) + p_3(i)$
Concatenate H and L to obtain σ

Path	$p_1(i)$	$p_2(i)$	$p_3(i)$
$i = 1$	3	2	5
$i = 2$	3	2	2
$i = 3$	4	3	3
$i = 4$	4	2	3



Optimality

Theorem 3*: If stage 2 is dominated by either stage 1 or 3, $\max\{\min p_1(i), \min p_3(i)\} \geq \max p_2(i)$, EJA is optimal.

If Theorem 3 fails, EJA still achieves an approximation ratio of 5/3+.

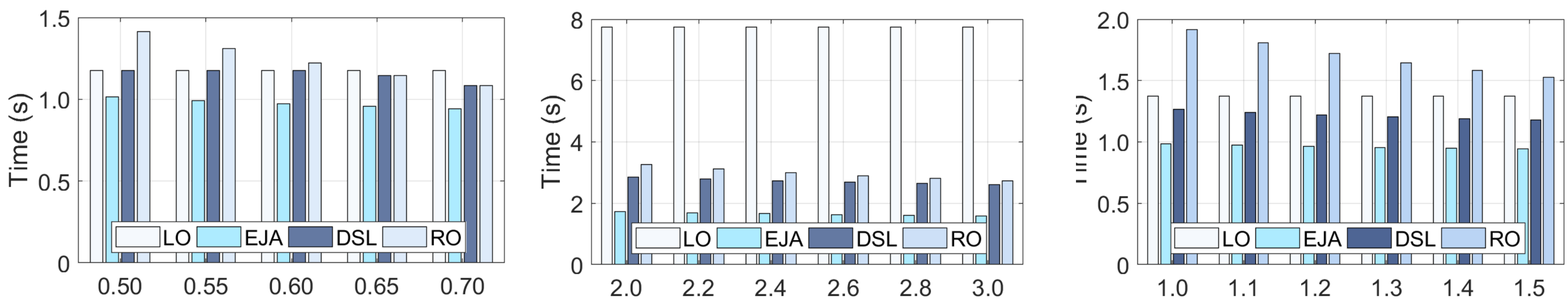
Path	$p_1(i)$	$p_2(i)$	$p_3(i)$
$i = 1$	3	2	5
$i = 2$	3	2	2
$i = 3$	4	3	3
$i = 4$	4	2	3

*Chen et al, *A new heuristic for three-machine flow shop scheduling*, *OR*, 1996.

+Framinan et al, *A review and classification of heuristics for permutation flow shop scheduling with makespan objectives*, *JORS*, 2004.

Simulation

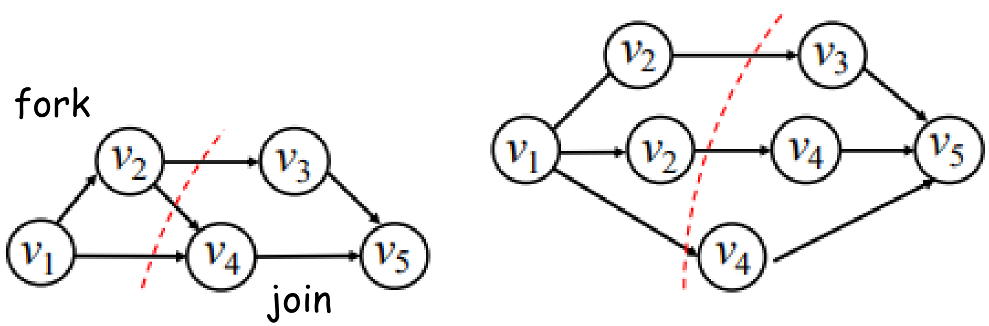
- Local and Cloud
  - Local: Raspberry Pi, Cloud: Amazon EC2
- Algorithms
  - LO: local only, EJA: Extended Johnson's Algorithm, DSL: no fine-grained pipeline, RO: remote only



(a) AlexNet: Bandwidth (MB/s) (b) Inception: Bandwidth (MB/s) (c) RandWire: Bandwidth (MB/s)

Extensions

- General structure: DAG
  - Conversion to multi-path
  - Replicated nodes at join and fork
- Heuristic solution
  - Scheduling: EJA on multi-path
  - Execution: Replicated node executed once (the first time)



(a) Before Conversion, G (b) After Conversion, G'

Multiple DNNs Offloading to Edges

Internet of Vehicles: smart city

- Autonomous driving systems: perception is a key
- Multiple cameras/sensors: multiple (identical) DNNs
- V2X: V (vehicle), I (infrastructure), N (network), P (pedestrian)

4. Optimal Partition and Scheduling

- Multiple line-structure DNNs
  - AlexNet and VGG-16
  - Video analytics and VR/AR
- Optimal partition and scheduling
  - Brute force: $O(k^n)$

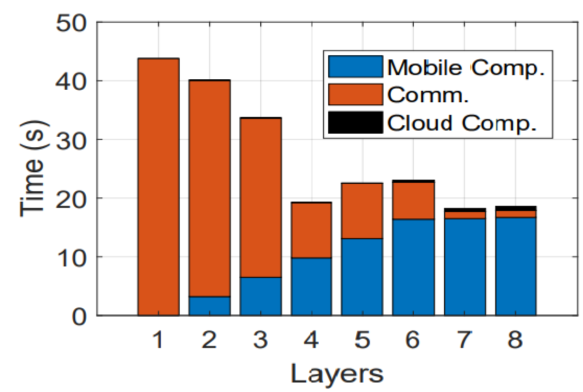
    n: # of copies, k: # of layers
- Existence of a better solution?
  - Exploring special application properties

Johnson Algorithm (JA)

- Closer look at the optimality for EJA
  - $\max\{\min p_1(i), \min p_3(i)\} \geq \max p_2(i)$
- However, $p_3(i) \approx 0$, reduced to 2-stage pipeline

Algorithm 2 Johnson Algorithm (JA)

```
H ← L ← ϕ
for i = 1 to m do
    if p1(i) ≤ p2(i) then
        H = H ∪ p(i)
    else
        L = L ∪ p(i)
Sort H increasingly based on p1(i)
Sort L decreasingly based on p2(i)
Concatenate H and L to obtain σ
```

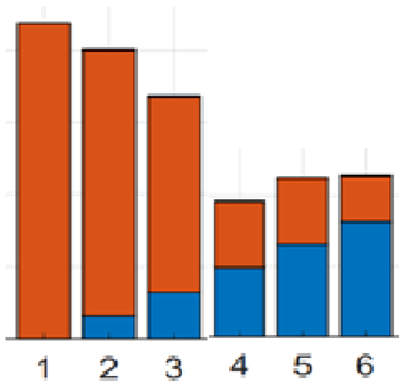


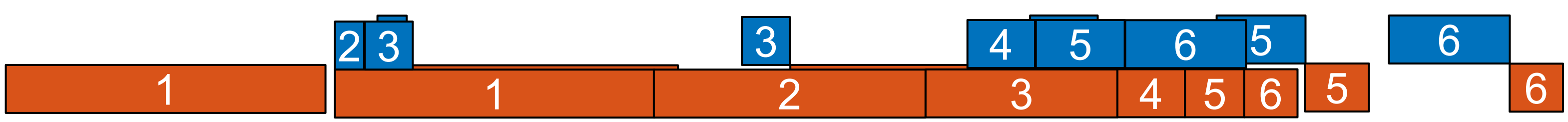
AlexNet

Johnson, Optimal Two- and Three-Stage Production Schedules With Set-up Time Included, *Naval Research Logistics Quarter, 1954.*

JA in Illustration

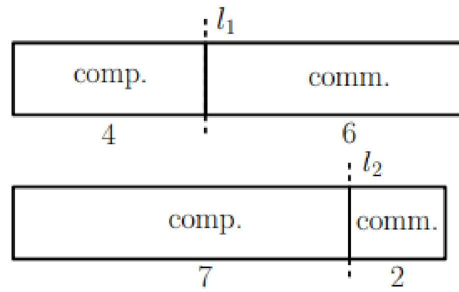
- Optimality is guaranteed: JA on 2-stage pipeline
- First six layers of AlexNet
  - One copy for each partition 6 copies
  - H = {1, 2, 3}, increasing order of blue
  - L = {4, 5, 6}, decreasing order of red
  - Comm.-domination (or comp.-domination)



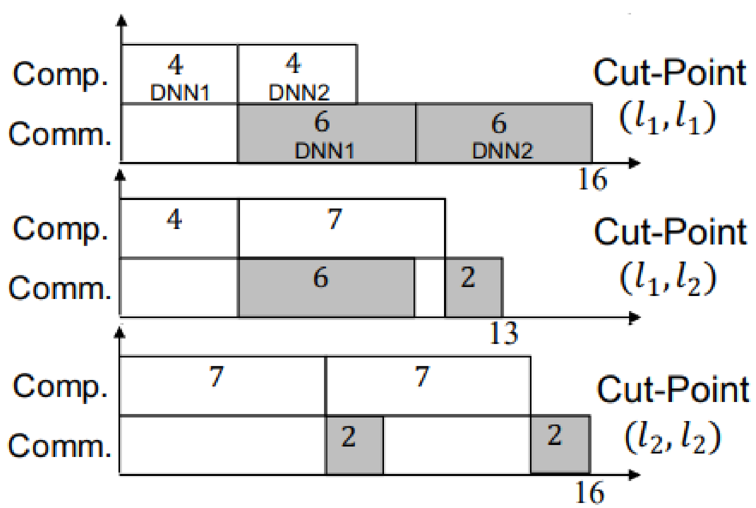


Multiple Line-Structure Example

- Two copies of line-structure DNN

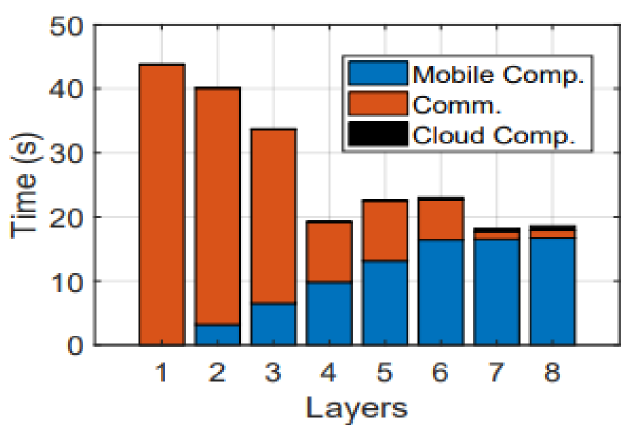


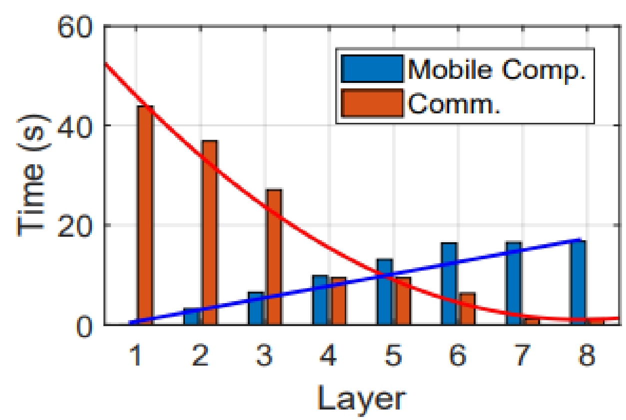
- Three possible partitions and scheduling



Special Application Property

- Line-structure (as the layer increases)
  - Computation time: linear increasing (convex) function
  - Communication time: monotonic decreasing convex function
- Computation vs. communication
  - Data size: 2 – 12 MB
  - Speed (uplink): 2-5 Mpbs (4G) and 6-54 Mpbs (WiFi)





Optimization Approximation

- Two functions on the continuous space
  - Both comp. and comm. are convex
  - One increasing and one decreasing

Theorem 4: A uniform partition of n line DNNs at the intersection will guarantee an approximation of $1 + \frac{1}{n}$.

Formal Proof: convex optimization

- Intersection point has the

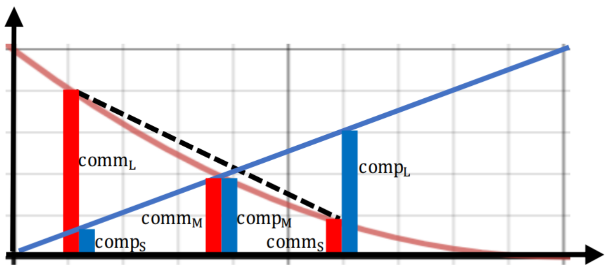
  min {max {comp., comm.}} for n copies
- Strong duality, then KKT, the uniform partition has the least max { ∑comp., ∑comm. }

Optimization Approximation (cont'd)

- Informal proof
  - Pair-wise "merge" and "replaced" by the middle-point

$$\frac{f(x)+f(x')}{2} \geq f(\frac{x+x'}{2})$$

  - Height of the intersection x* ≤ any max {comp., comm.}



  - Two gaps: first pair in comm. and last pair in comp.

    Only one is counted in comm.-domination or comp.-domination

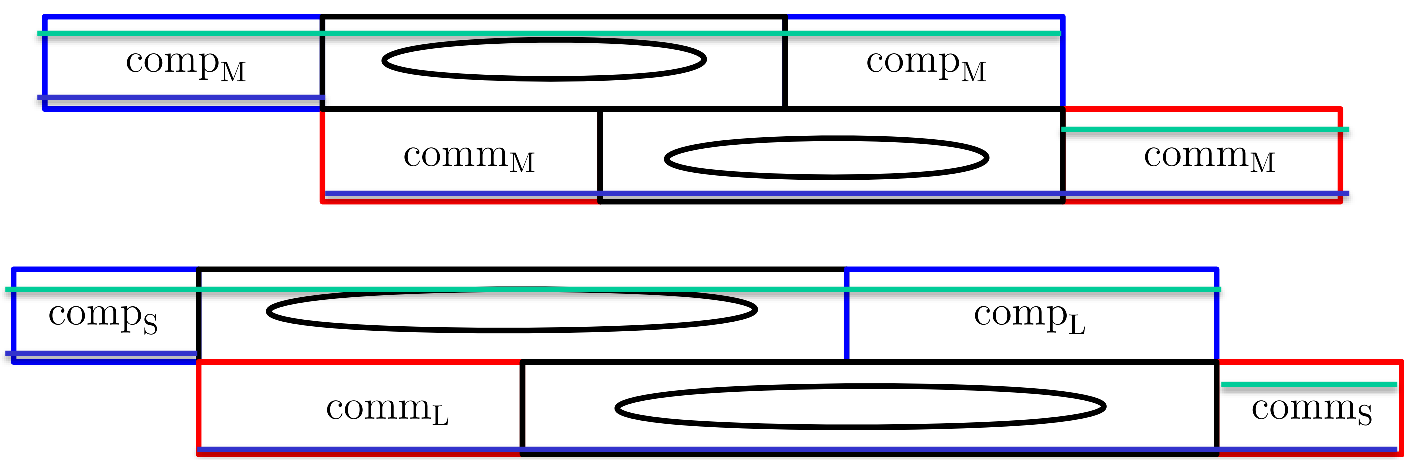
  - When n → ∞, $1+\frac{1}{n}=1$

Sufficient Condition for Optimality

- For a given set of partitions
  - Left/right most partition: $(comp_s, comm_l)$ / $(comp_l, comm_s)$
- Intersection partition: $(comp_m, comm_m)$

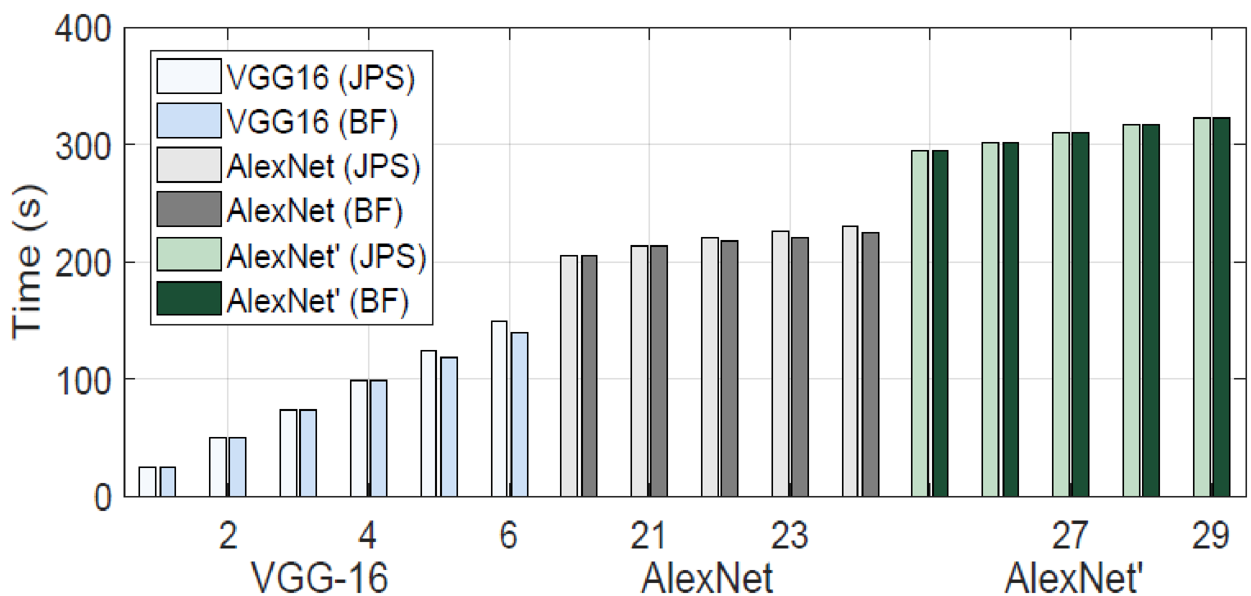
Theorem 5: The uniform partition beats the given set if

$3comp_m < comp_s + comp_l + comm_s$ and $3comm_m < comp_s + comm_l + comm_s$



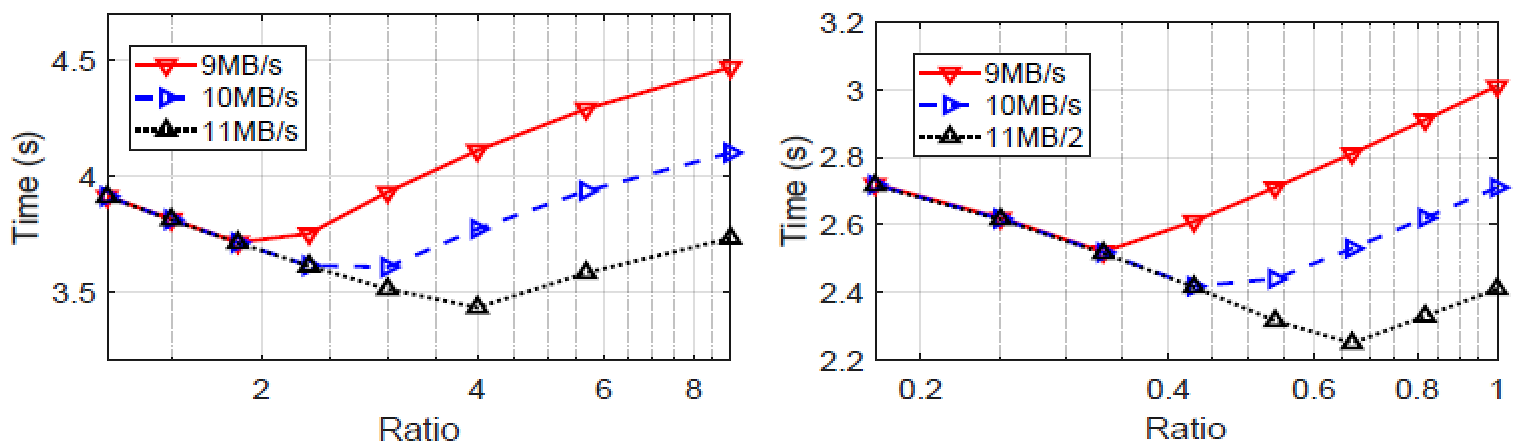
Simulation (cont'd)

- Partition methods
  - Joint Partition and Scheduling: JPS, Brute Force: BF
- Application
  - VGG-16, AlexNet, and AlexNet' (curve fitting) with n = 1, …, 29



Simulation (cont'd)

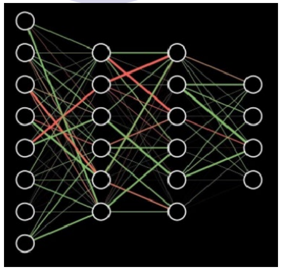
- Discrete version of intersection
  - Intersection: x^*, right: $\lceil x^* \rceil$ and left: $\lfloor x^* \rfloor$
  - Ratio: $|x^* - \lceil x^* \rceil| : |x^* - \lfloor x^* \rfloor|$



  - AlexNet VGG-16

5. Conclusions and Future Work

- Offloading as a Service
  - Edge/cloud networks

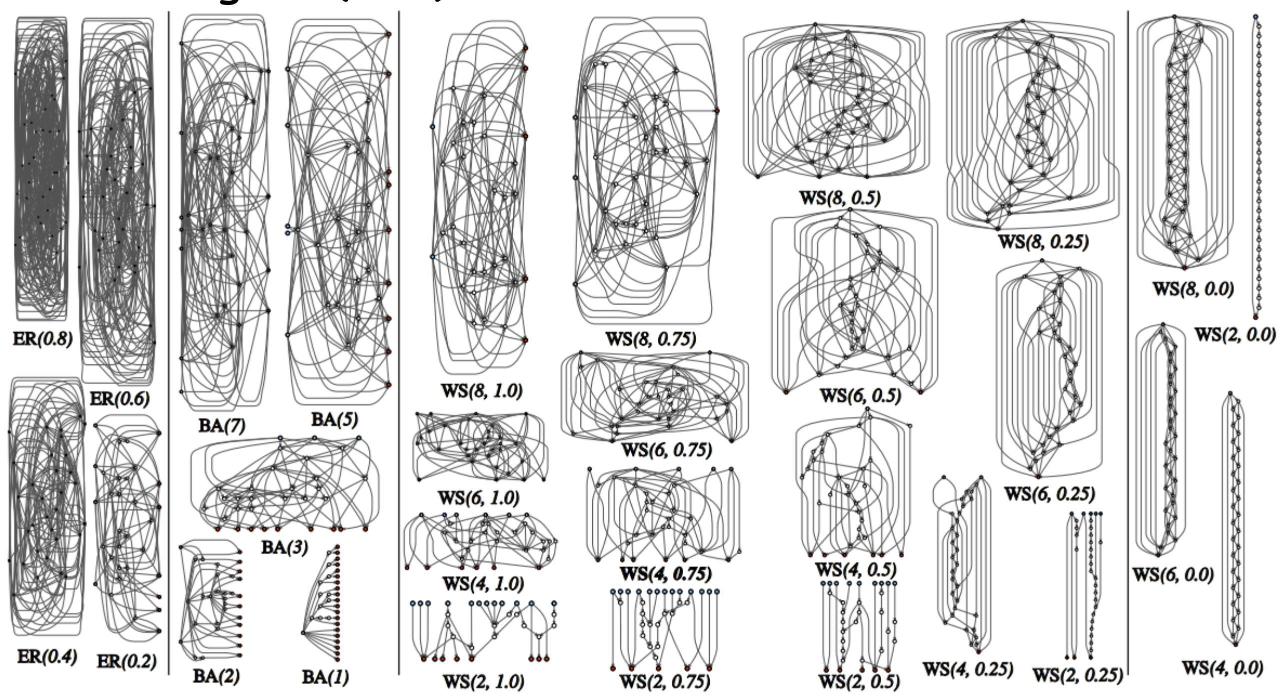


- Different DNNs
  - Single path, multi-path, and DAG
- Joint partition and scheduling
  - Johnson's rule and its extensions using pipelines
  - Unique properties of comp. and comm. of DNNs
- Future work
  - Multi-tier offloading: edge and then cloud pipeline
  - Optimal partition and scheduling of special DAGs

6. Some Reflections

Back to the past: interconnection networks

- Randomly wired NNs (random graphs): neuroscience
- Erdos-Renyi (ER): random, Barabasi-Albert (BA): preferential
- Watts-Strogatz (WS): small-world



Xie et al, Exploring Randomly Wired Neural Networks for Image Recognition, *ICCV'19*

Avoiding: Reinventing the Wheel

Reference searching practice

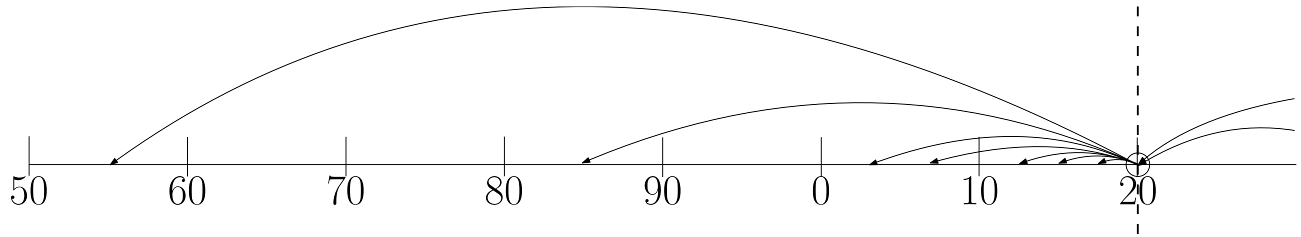
- 1 to 3 iterative process: references, references' references

Knowledge span

- Career span: 5 to 7 years in MS + PhD period

Art of citations

- Good practice for citation: yearly distributions



Zheng and Wu, Snowballing Effects in Preferential Attachment, *ICCCN'15*

Questions

Collaborators: Ning Wang (Rowan U.) and Yubin Duan (Temple U.)