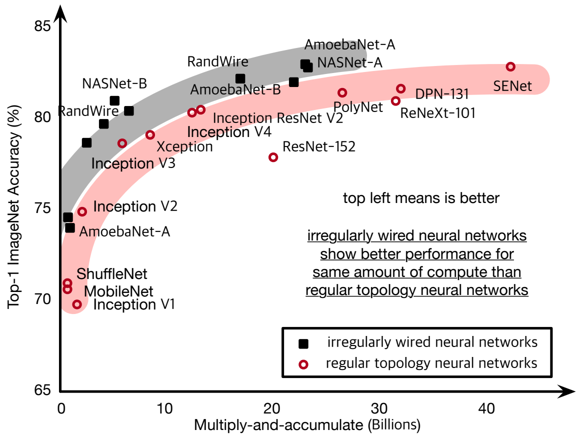
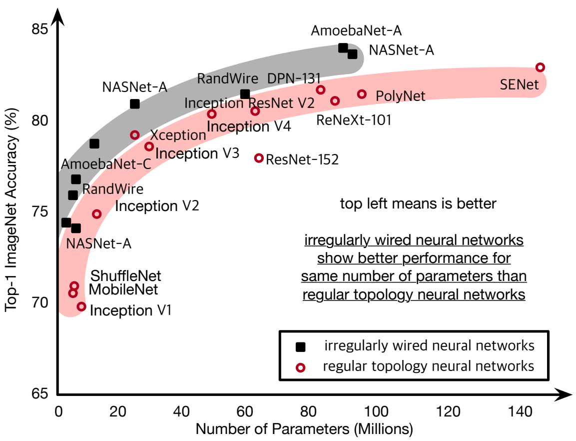
A Comparison between Irregularly Wired Neural Networks and Conventional Regular Topology Neural Networks



(a) ImageNet accuracy vs number of multiply-and-accumulate.



(b) ImageNet accuracy vs number of parameters.

Figure 14. ImageNet accuracy vs number of multiply-and-accumulate or parameters, where irregularly wired neural networks show higher performance for same amount of compute or number of parameters than regular topology neural networks.

B Proof for Optimal Peak Memory Footprint from the Dynamic Programming-based Scheduling

Here we prove the optimality of the above dynamic programming-based scheduling algorithm.

THEOREM 1. *In order to find a schedule s^* with an optimal peak memory consumption μ^*, it is sufficient to keep just one schedule-peak memory pair $(s_i,\ z_i)$ in S_{Ti} for each zero-indegree set z_i, and to append subsequent nodes on top of s_i to get s_{i+1} in each search step.*

Proof. If $i = 0$, the optimal s_0 is an empty sequence and μ_0 must be 0. On the other hand, if $i \geq 1$, assume that (*suboptimal*) v_i constitutes s^*, substituting $u_i^* \in z_i$ and achieves μ^*. In such case, let v_i be replaced with (*optimal*) u_i^*, which will result in $\mu_{peak} \leftarrow \min(\mu_i + \prod v_i.\text{shape}, \mu_i + \prod u_i^*.\text{shape})$, and μ_{i+1} is calculated by deducting $\prod p_i.\text{shape}$, $\forall p_i \in (u_i.\text{preds} \cap \text{zero-outdegree}(s_{i+1}, \mathcal{G}))$. By recursively applying u_k for rest of the search steps k, the algorithm should find an alternative sequence $s^{*\prime}$ with $\mu^{*\prime} \leq \mu^*$ due to the $\min$ operator above, contradicting the original assumption on the optimality of s^*. Therefore, our algorithm finds a schedule with an optimal peak memory consumption. ■

C Complexity Analysis of the Dynamic Programming-based Scheduling and Proof

We compare the complexity of exhaustively exploring $\mathcal{S}_T$ and our dynamic programming-based scheduling. While the algorithm both lists candidate schedules and calculates their peak memory footprint, we consider the peak memory footprint calculation as one operation while deriving the complexity. In order to visualize the analysis, we invent $\mathcal{G}$ in Figure 15 to demonstrate the upper bound complexity of each algorithm. It has a single entry node and a single exit node Ⓐ and Ⓩ, respectively, and all other nodes constitute independent branches between the entry and the exit node.

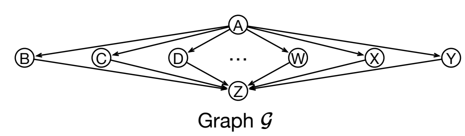


Figure 15. Topology of $\mathcal{G}$ to demonstrate the upper bound complexity of each algorithm.

First, we demonstrate the complexity of the recursive topological sorting that exhaustively explores $\mathcal{S}_T$. Since there is a single entry node and a single exit node, there will be $|V - 2|$ remaining nodes and these nodes can be scheduled independently of one another, thereby the number of candidate schedules become $\langle |V - 2|! \rangle$ and the overall complexity becomes $\mathcal{O}(|V|!)$, where $|V|$ denotes the number of nodes. On the other hand, for the dynamic programming we calculate the number of candidates by utilizing the number of schedules that gets memoized. Our memoization takes advantage of the zero-indegree sets z for each search step. Following first demonstrate the number of z in each search step which also means the number of nodes scheduled.

$$
\begin{aligned}
&\text{Search step 0:} && 1 \\
&\text{Search step 1:} && 1 && \text{, single entry node.} \\
&\text{Search step 2:} && \binom{|V|-2}{1} \\
&\text{Search step 3:} && \binom{|V|-2}{2} \\
& && \vdots \\
&\text{Search step } |V|-2\text{:} && \binom{|V|-2}{|V|-1} \\
&\text{Search step } |V|-1\text{:} && \binom{|V|-2}{|V|-2} \\
&\text{Search step } |V|\text{:} && 1 && \text{, single exit node.}
\end{aligned}
$$

On top of this, each step would make an iteration over the set of candidate nodes to discover the next search step's z. Therefore, search step 1 would explore $|V|-2$ nodes and the search steps 2 to $|V|-1$ would iterate over $|V|-1-i$ nodes. Summarizing this would yield:

$$
\begin{aligned}
&1+1\times(|V|-2)+\binom{|V|-2}{1}\times(|V|-3)+ \\
&\qquad ...+\binom{|V|-2}{|V|-2}\times 0+1 \\
&=1+\binom{|V|-2}{0}\times(|V|-2)+\binom{|V|-2}{1}\times(|V|-3)+ \\
&\qquad ...+\binom{|V|-2}{|V|-2}\times 0+1 \\
&=2+\sum_{i=0}^{|V|-2}\binom{|V|-2}{i}\times(|V|-2-i) \\
&=2+(|V|-2)\times 2^{|V|-3} \\
&\leq(|V|-2)\times 2^{|V|-2} \qquad \text{, for } |V|\geq 4 \\
&\leq |V|\times 2^{|V|}
\end{aligned}
$$

As a result, we can see that our dynamic programming-based scheduling algorithm is bounded by $\mathcal{O}(|V|\times 2^{|V|})$. By using Stirling's approximation on the complexity of the recursive topological sorting, we can prove that the dynamic programming-based scheduling algorithm should be significantly faster than the recursive topological ordering.