GeneSys: Enabling Continuous Learning through Neural Network Evolution in Hardware

Ananda Samajdar, Parth Mannan, Kartikay Garg, and Tushar Krishna

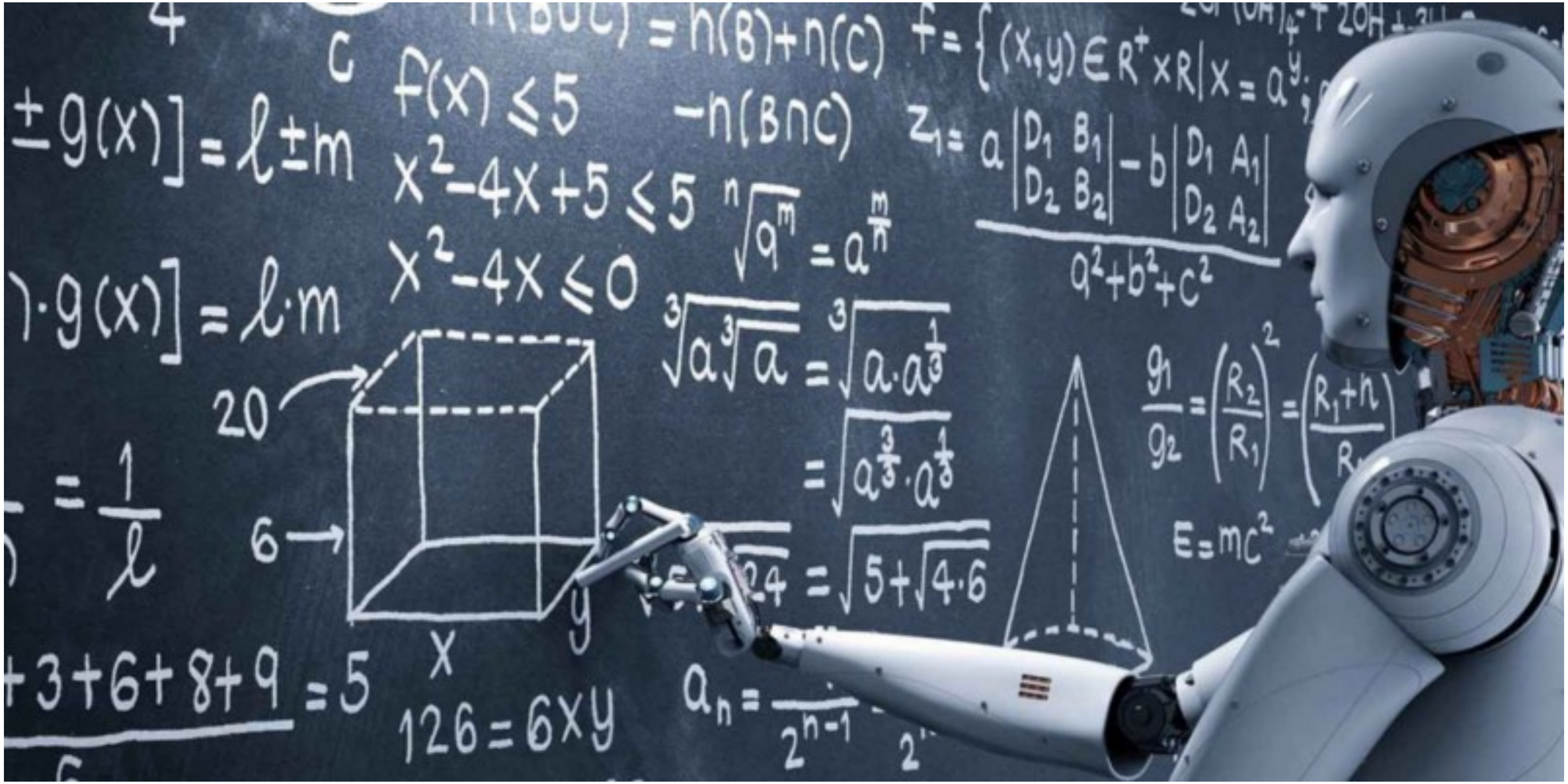
Georgia Institute of Technology

Synergy Lab [http://synergy.ece.gatech.edu]

anandsamajdar@gatech.edu

October 24, 2018

The Dream!



What is Continuous Learning?

Robotic cook @ Bosch Amusement Park, Sasebo

Cooks savory pancakes

Can it gain expertise with experience?

Learn new recipes

Deep Learning Land

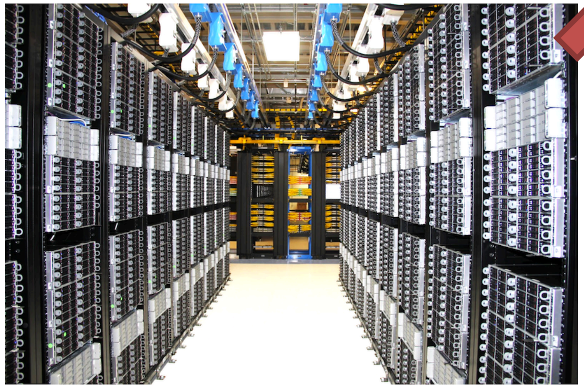
Not viable for continuous learning

What happens if

ImageNet Dataset

Massive amounts of **structured, labelled** data

No dataset

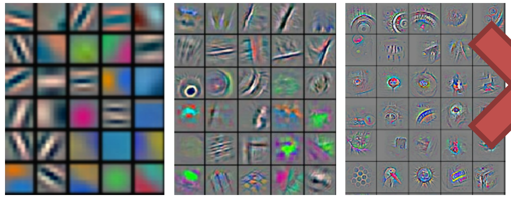


High performance cluster

No access to large compute

Training

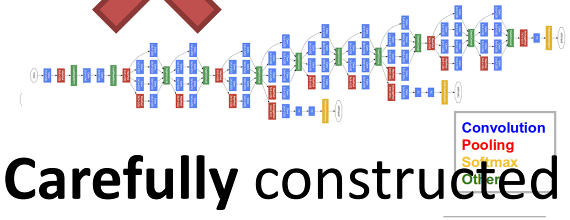
Takes weeks



No internet

Device

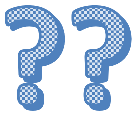
No ML expert



Carefully constructed Neural Network topology

Task itself changes

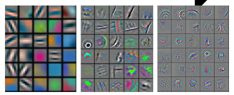
Continuous Learning Landscape



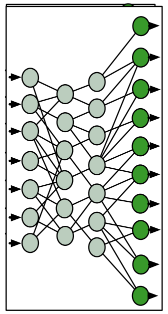
Learning Agent

Robust

Topology



Weights



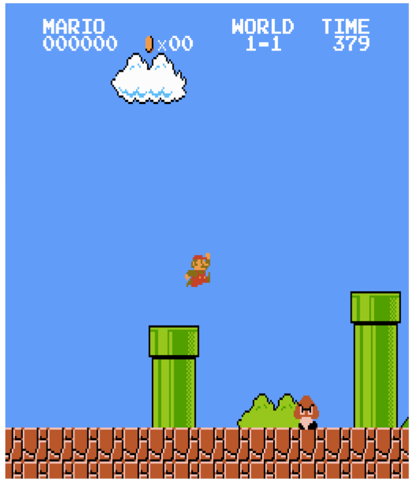
Interacting Agent

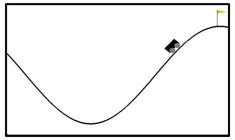
Continuous

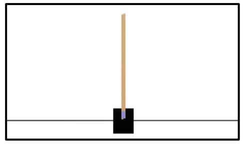
Action

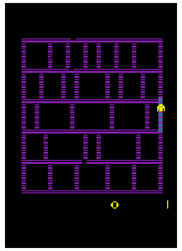
Reward

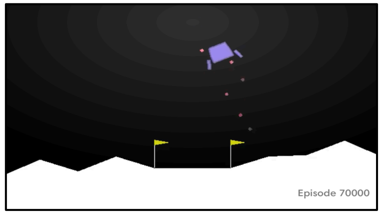
Environment

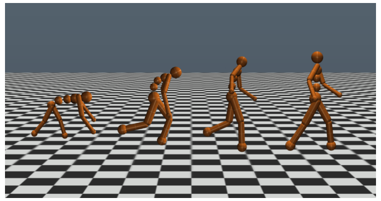












Accumulated Rewards

This is Reinforcement Learning

Learn multiple tasks

Conventional RL: Challenges

Deep NNs used internally

- ! Manual hyperparameter tuning

Each update results in **Backpropagation**

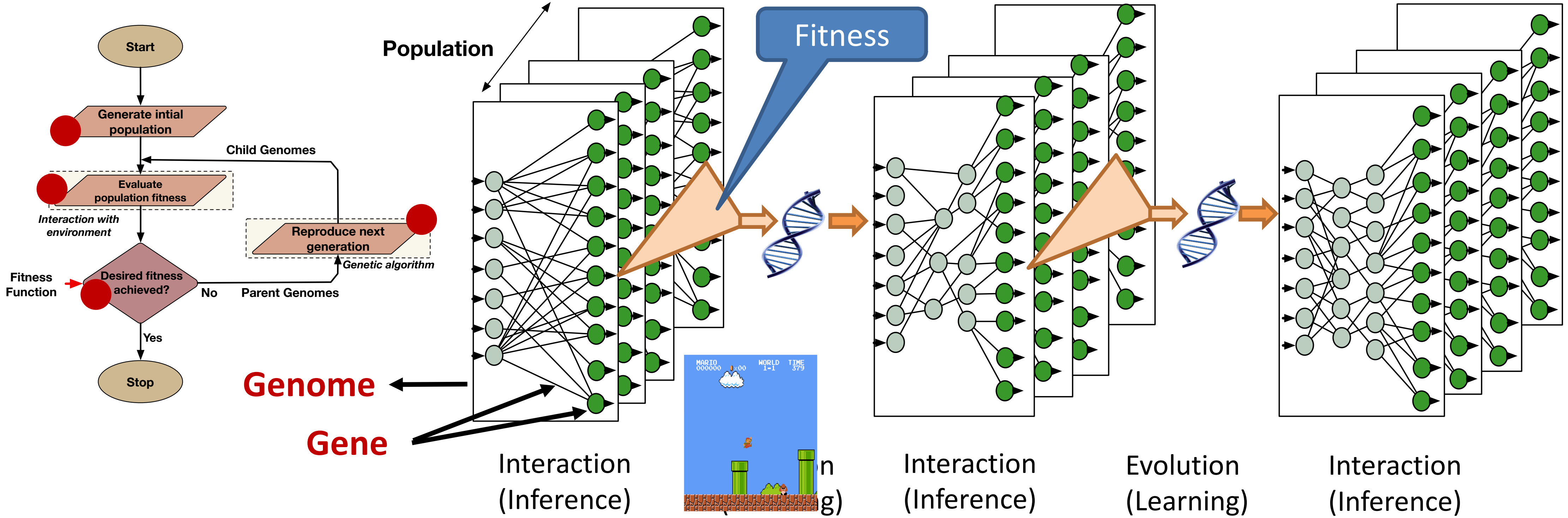
- ! High compute requirement at every update
- ! High memory overhead
- ! Not scalable

Not viable for continuous learning

Outline

- **Motivation**
- **Neuro Evolutionary Algorithm**
  - Algorithm description
  - Characterizing NEAT
- **Microarchitecture**
- **Evaluations**

Neuro-Evolutionary (NE) Algorithm



Neural Network (NN) expressed as a graph

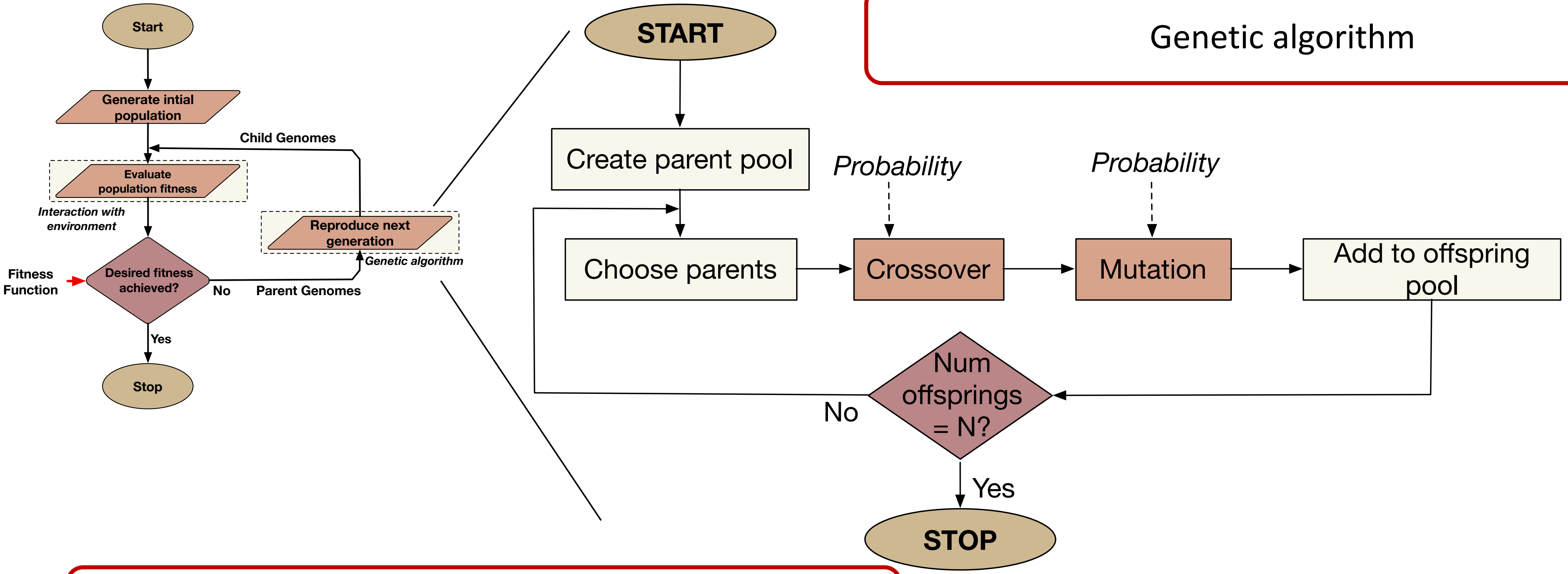
Gene: Vertex or Edge in the graph

Genome: Collection of all genes (i.e., a NN)

[1] Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), 99-127.

Neuro-Evolutionary (NE) Algorithm

Genetic algorithm



Neural Network (NN) expressed as a graph

NeuroEvolution of Augmented Topologies (NEAT) [1]

Gene: Vertex or Edge in the graph

Genome: Collection of all genes (i.e., a NN)

[1] Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), 99-127.

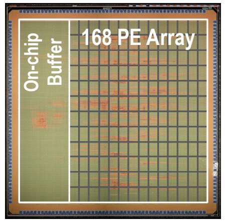
Challenges with Genetic Algorithms!

Too much compute!

Can it converge in reasonable time?

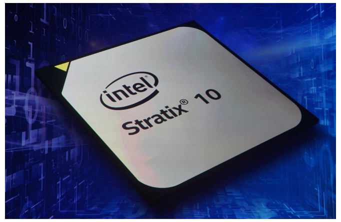
What about accuracy?

déjà vu! Looks like Deep nets in the 90s



Eyeriss


GPU



FPGA

HW solutions enabled Deep Learning

Can we do the same with EA?

Outline

- **Motivation**
- **Neuro Evolutionary Algorithm**
  – Algorithm description
  – Characterizing NEAT
- **Microarchitecture**
- **Evaluations**
  – Implementation
  – Results

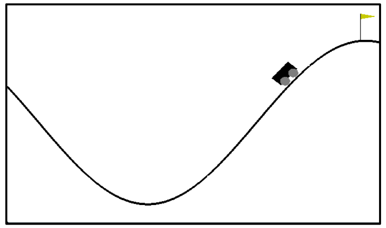
Characterization of NEAT

NEAT - Python

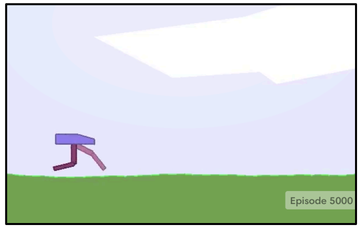
Codebase

OpenAI Gym

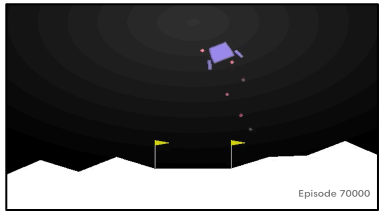
Environments



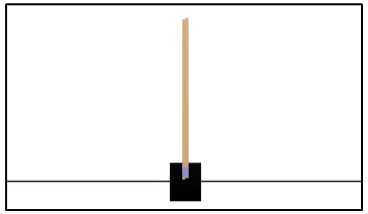
Mountain car



Bipedal



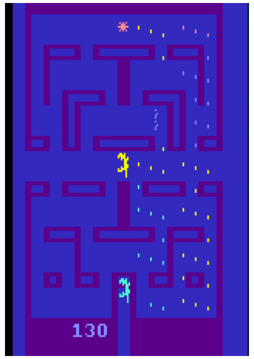
Lunar Lander



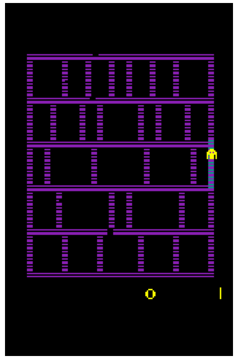
Cart pole

Ran each environment till convergence, multiple times

Airraid-RAM



Alien-RAM

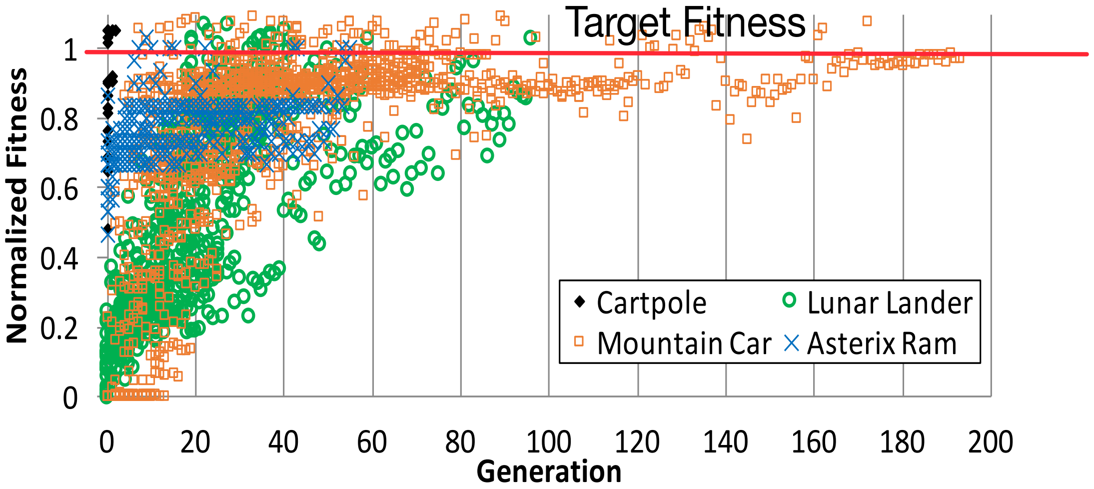


Amidar-RAM

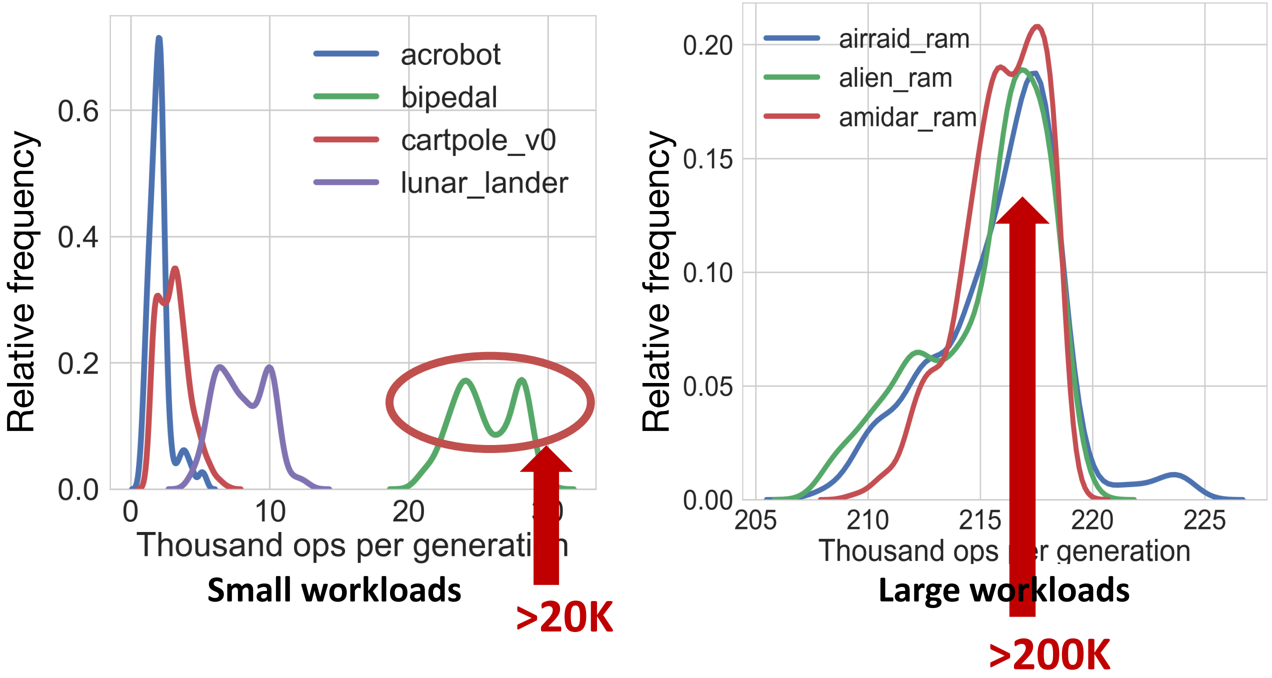
Only changed fitness function between workloads

Characterization of NEAT

Computations



Distribution of Operations/Generation



Population level parallelism

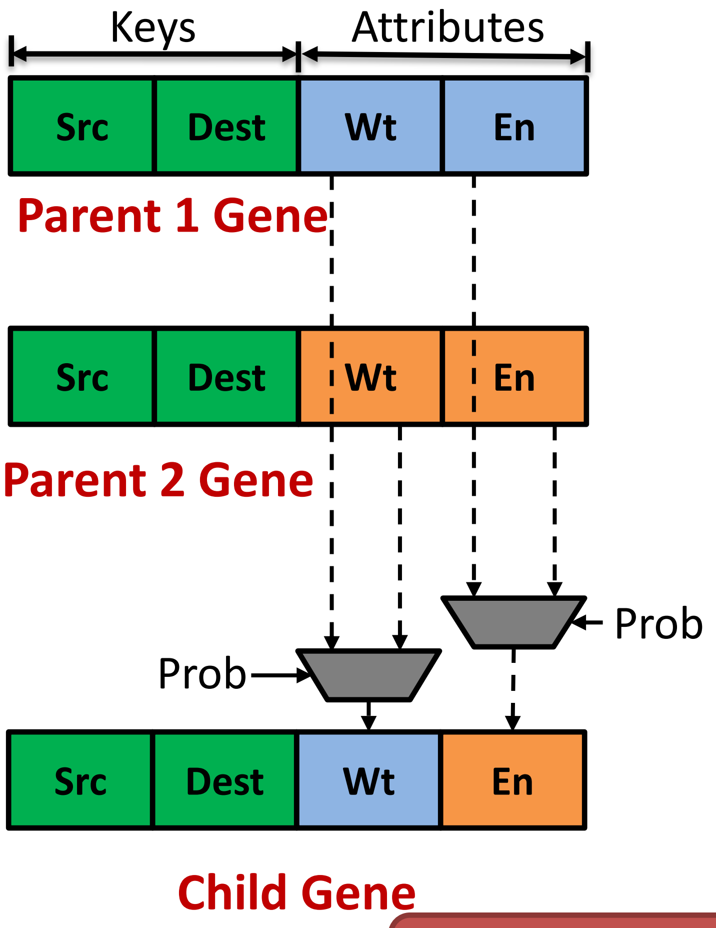
Gene level parallelism

All operations are independent

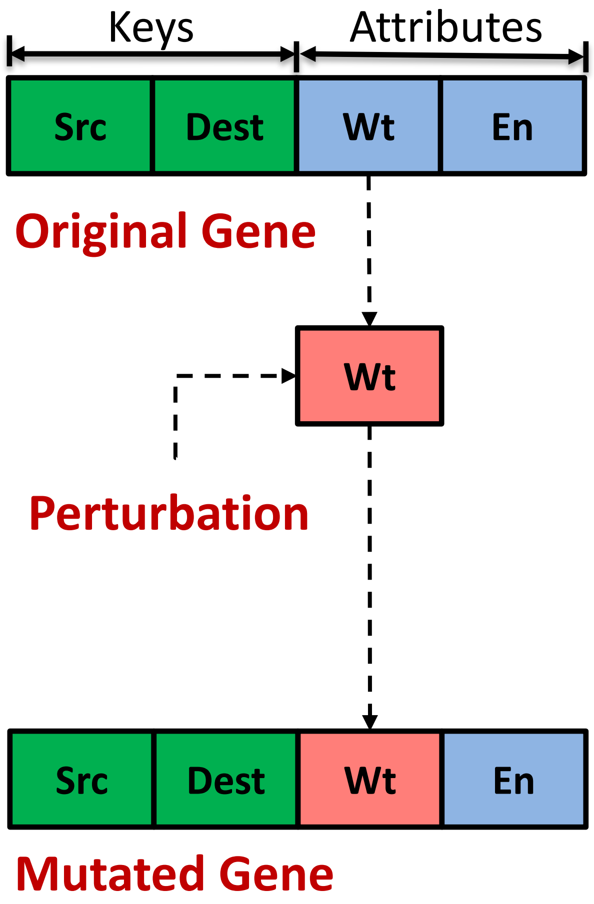
Large operation level Parallelism

Operations in NEAT

Crossover



Mutation



Addition mutation

- *Add new node*
- *Add new connection*

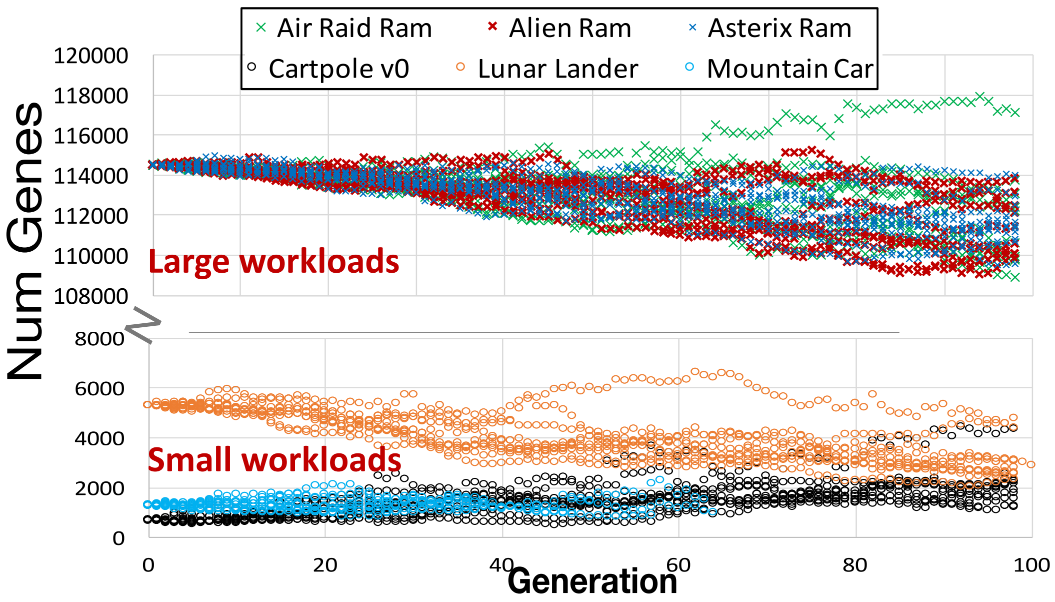
Deletion mutation

- *Delete connection*
- *Delete node*

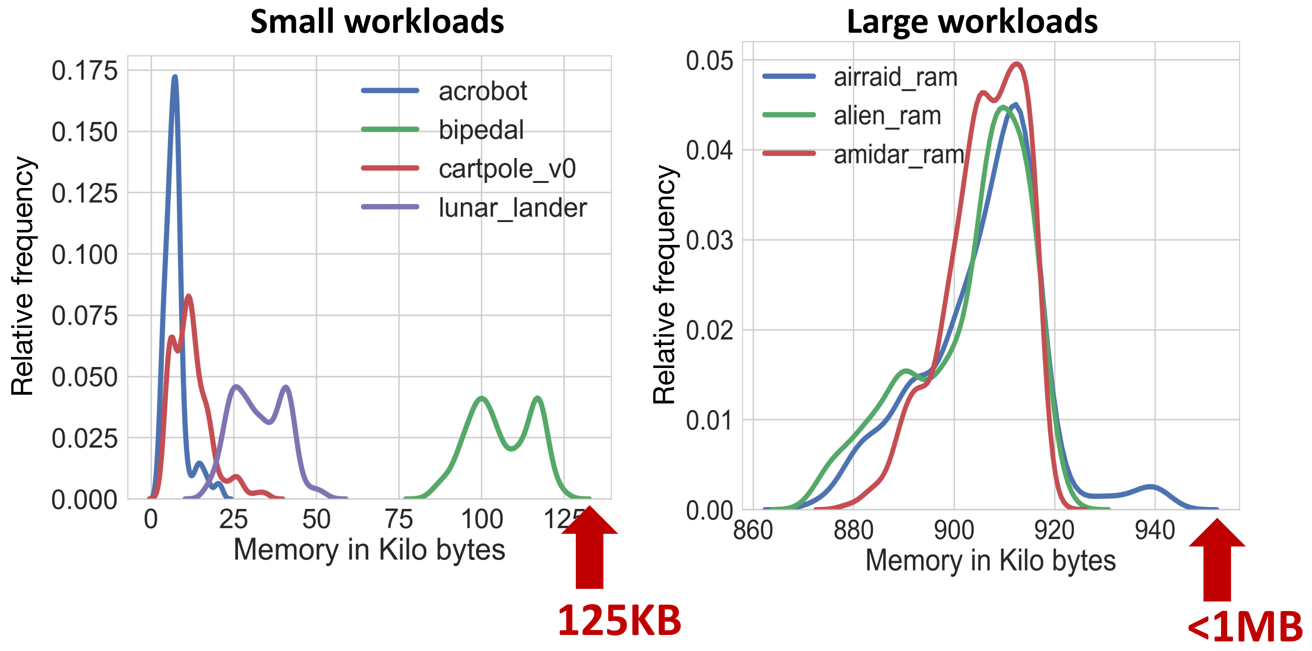
Simple operations

Characterization of NEAT

Memory



Distribution of Memory footprint/Generation



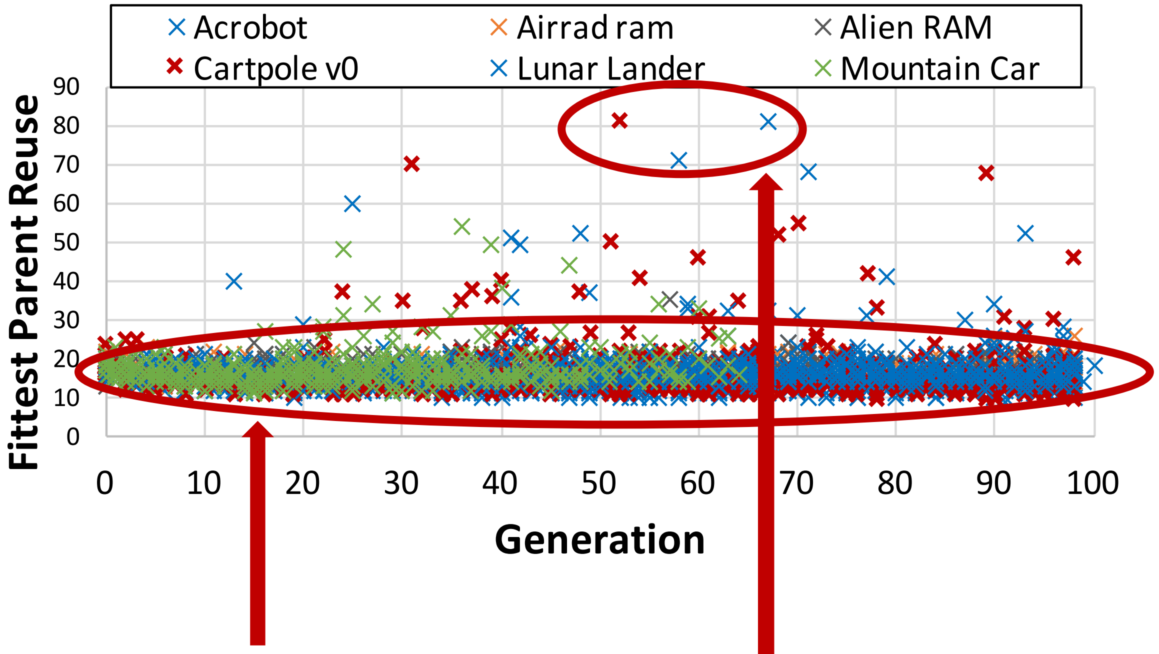
Entire population can fit on-chip

Only need to store the weights and node info

Characterization of NEAT

Memory

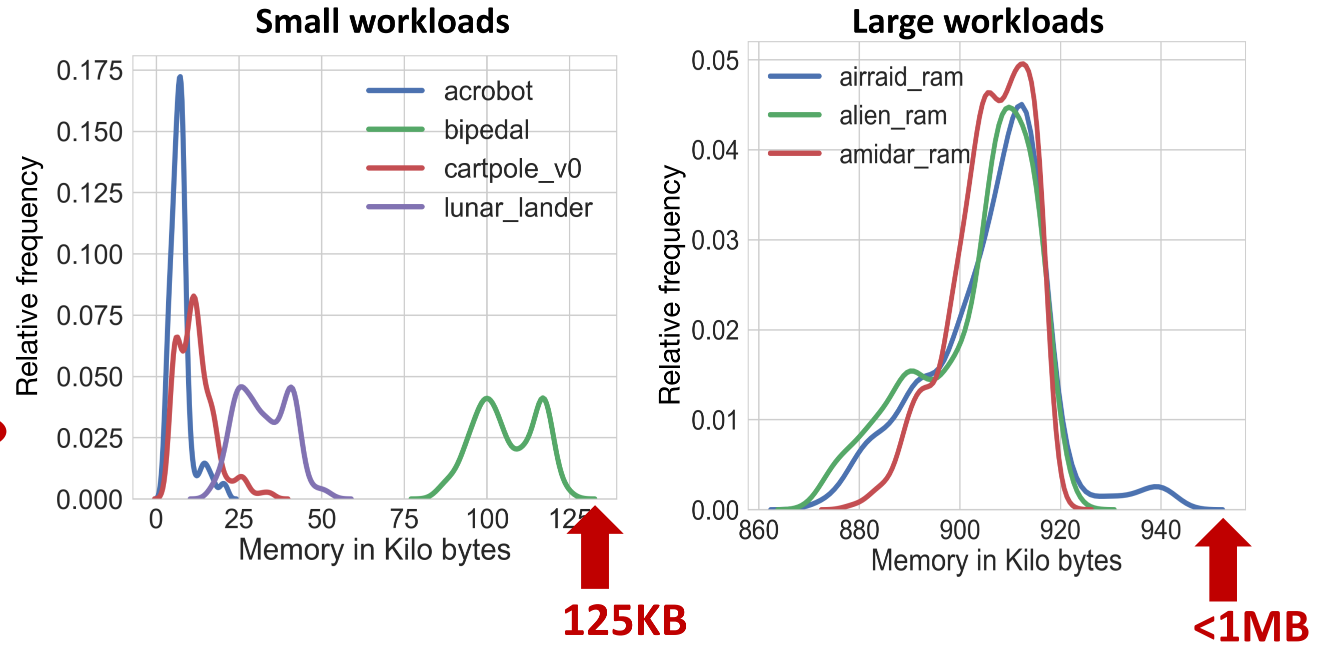
Opportunity for Reuse



Fittest parent genome is used about ~10-20 times each generation

Even higher in certain cases

Distribution of Memory footprint/Generation



Entire population can fit on-chip

Only need to store the weights and node info

Motivating Hardware Solution

Massive parallelism → **Scalability** → **Faster convergence**

Gene and Population level parallelism

Power efficiency → **More deployable compute** → **Target complex problems**

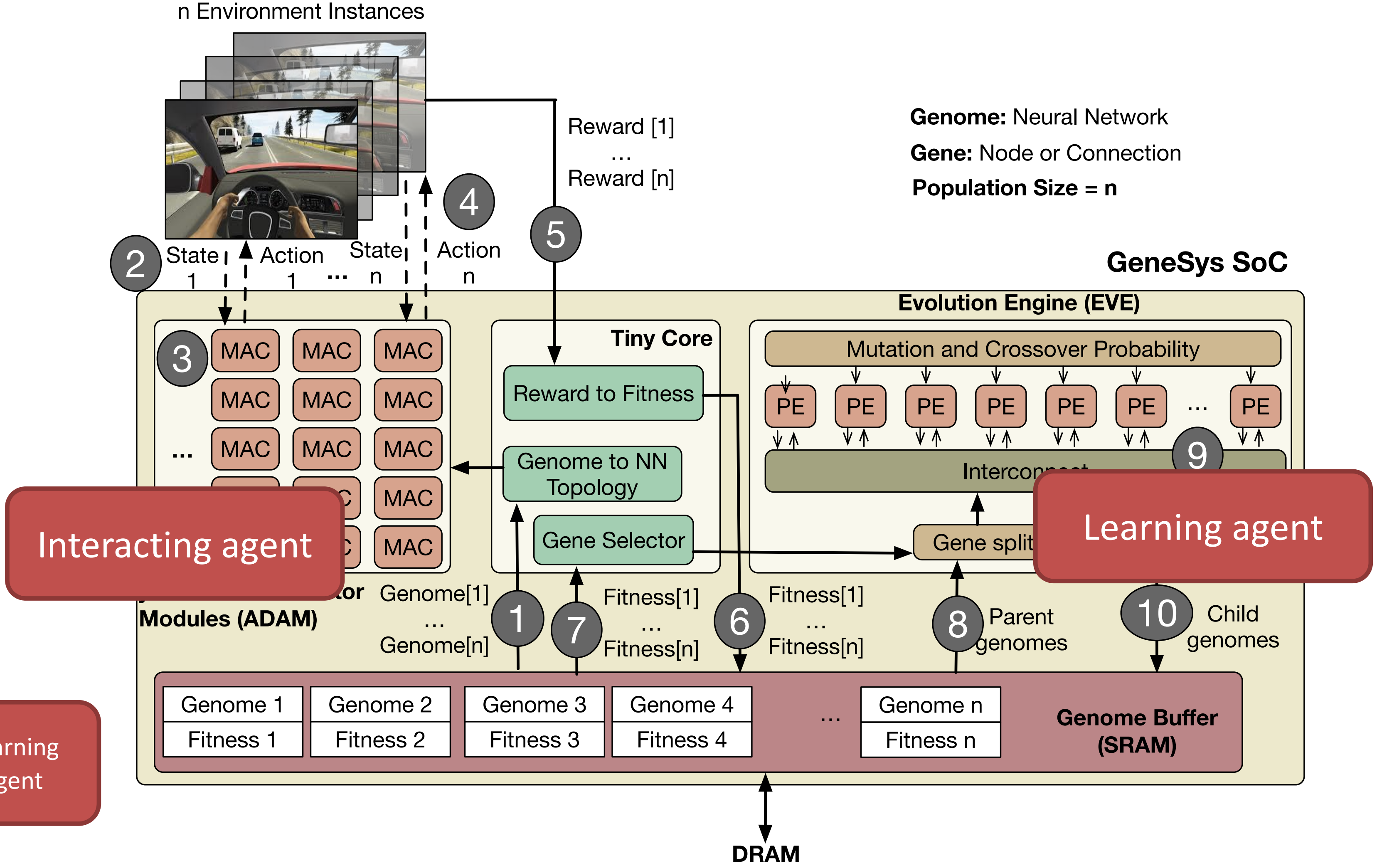
Simple HW friendly operations

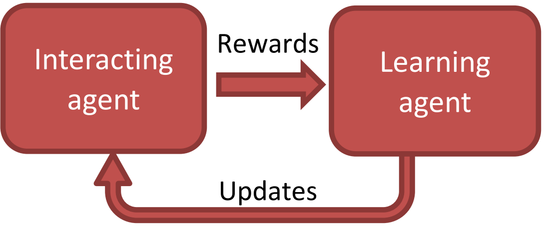
Hardware-Software codesign of NE makes them viable for continuous learning

Outline

- **Motivation**
- **Neuro Evolutionary Algorithm**
  - Algorithm Description
  - Characterizing NEAT
- **Microarchitecture**
- **Evaluations**

GeneSys SoC

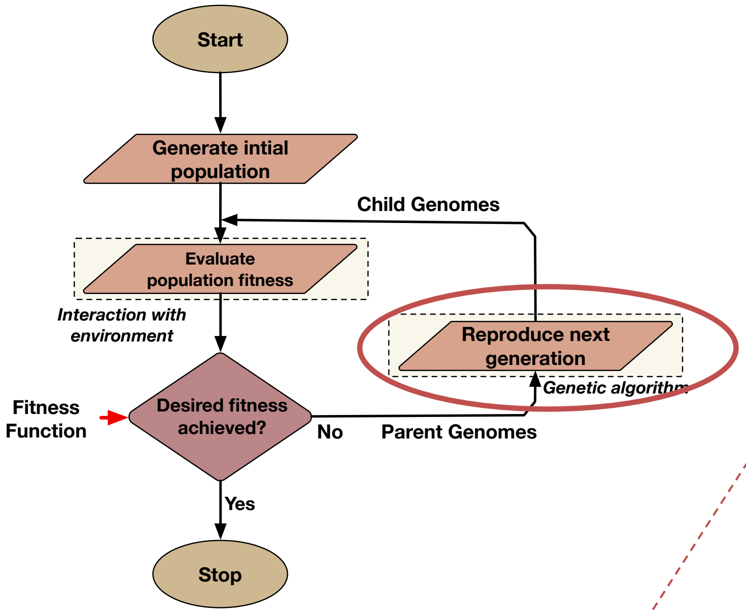


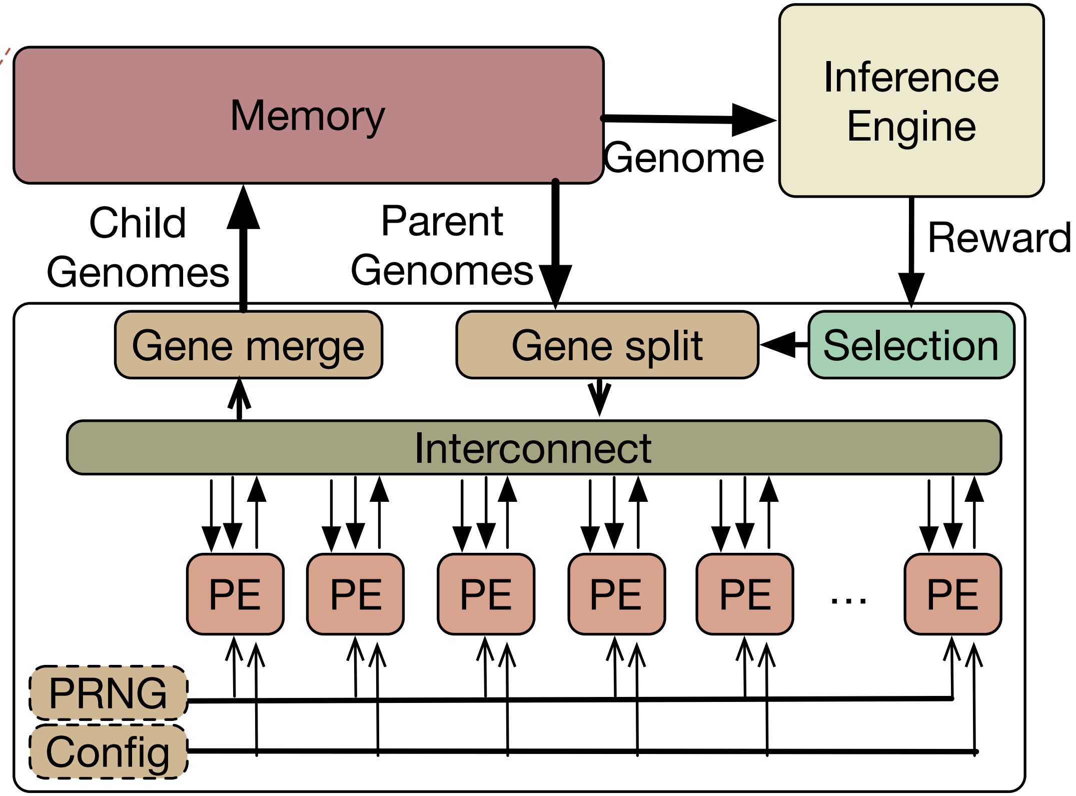


GeneSys

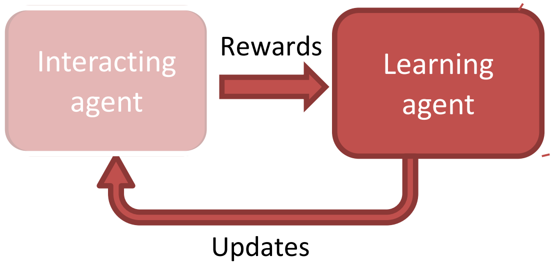
Evolution Engine: EvE Microarchitecture

Large number of PE to exploit parallelism



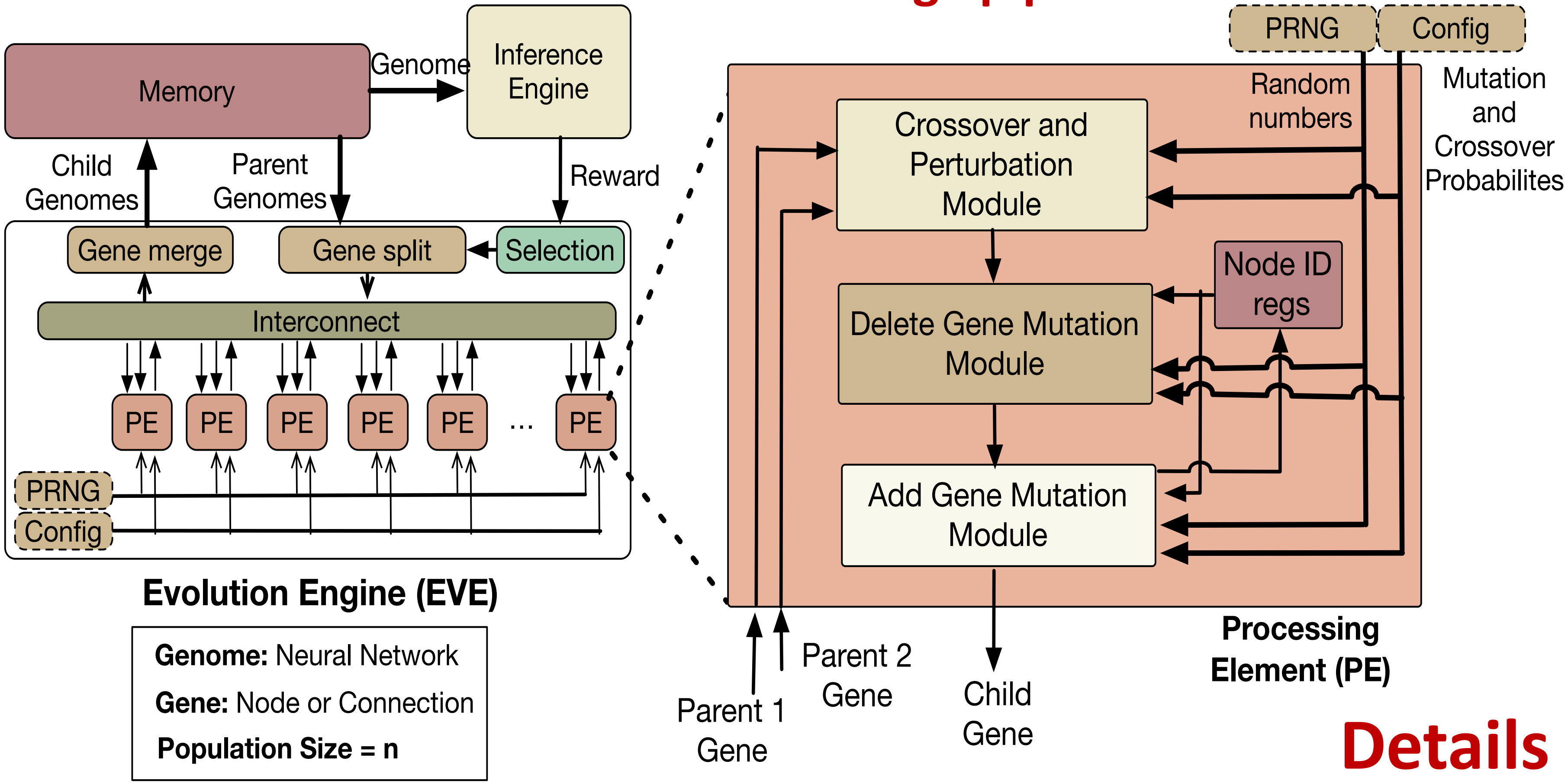


Evolution Engine (EVE)



PE Microarchitecture

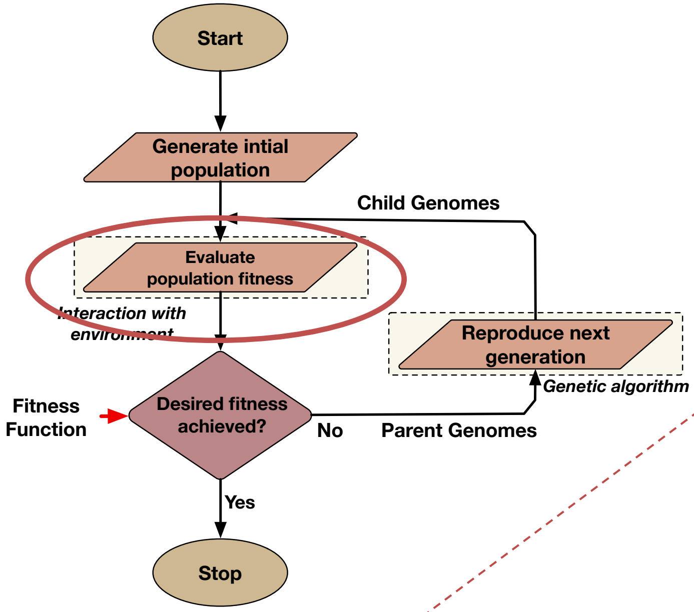
4 stage pipeline

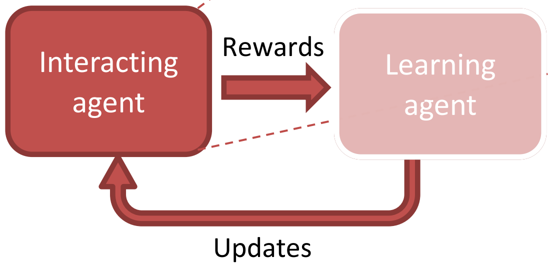


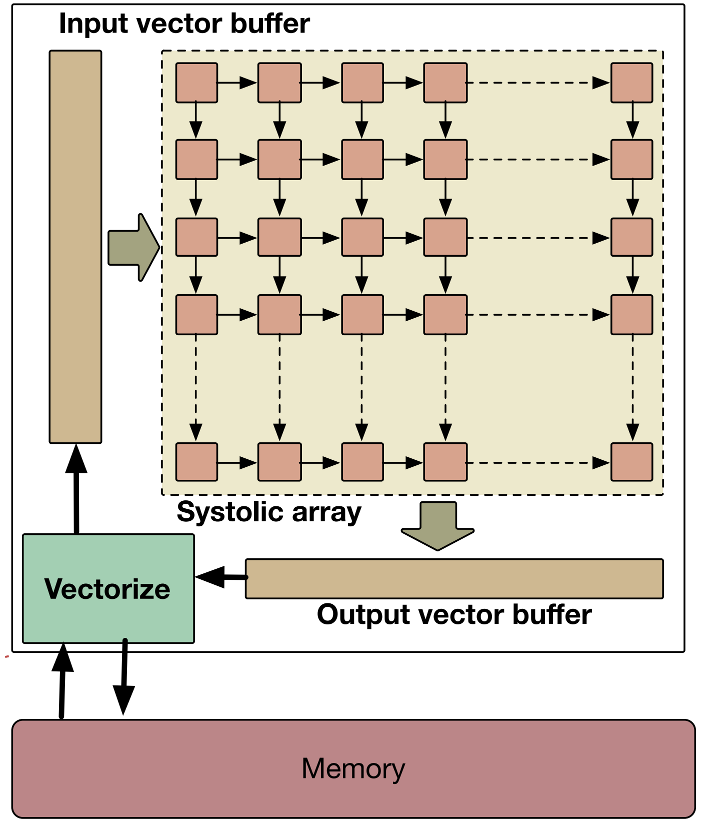
- **One child per PE**
- **One child gene processed per cycle**

Details of pipeline stages in the paper

Inference Engine: ADAM Microarchitecture







Networks generated by NEAT are irregular (thus sparse)

Inference is similar to graph processing

Pack input vectors for dense compute

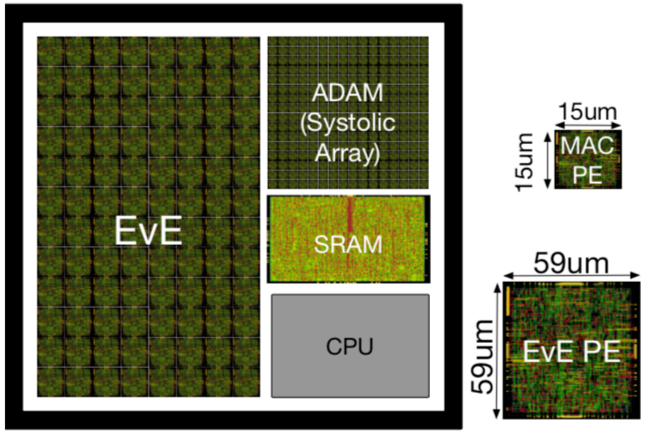
Exploit Population Level Parallelism

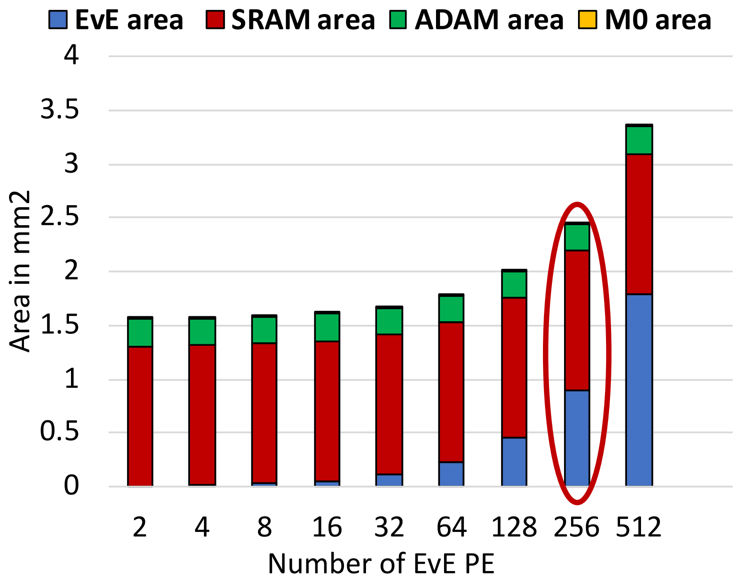
Outline

- **Motivation**
- **Neuro Evolutionary Algorithm**
  - Algorithm Description
  - Characterizing NEAT
- **Microarchitecture**
- **Evaluations**

Implementation

GeneSys Parameters	
Tech node	15nm
Num EvE PE	256
Num ADAM PE	1024
EvE Area	0.89 mm2
ADAM Area	0.25 mm2
GeneSys Area	2.45 mm2
Power	947.5 mW
Frequency	200 MHz
Voltage	1.0 V
SRAM banks	48
SRAM depth	4096



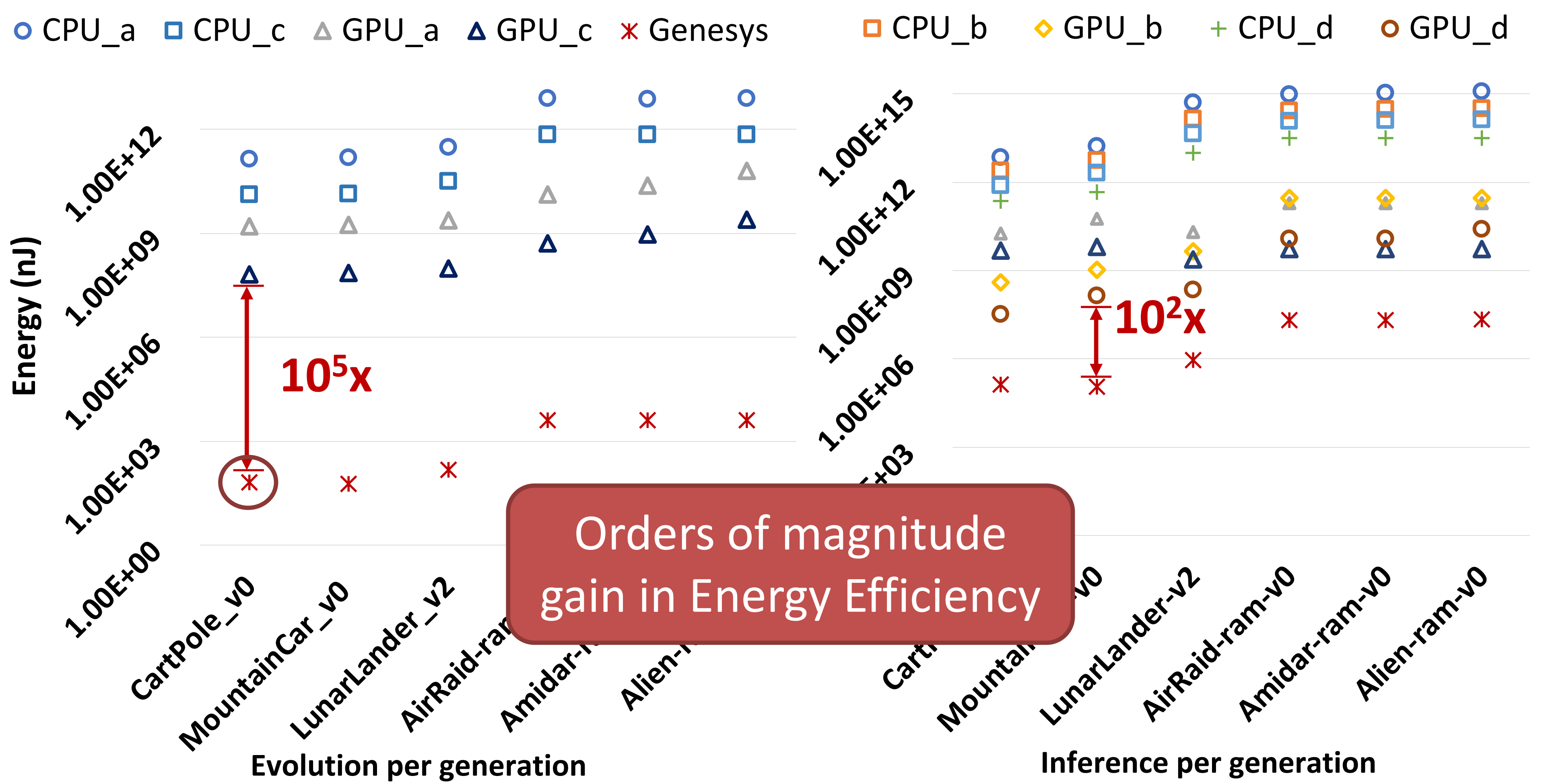


Evaluations

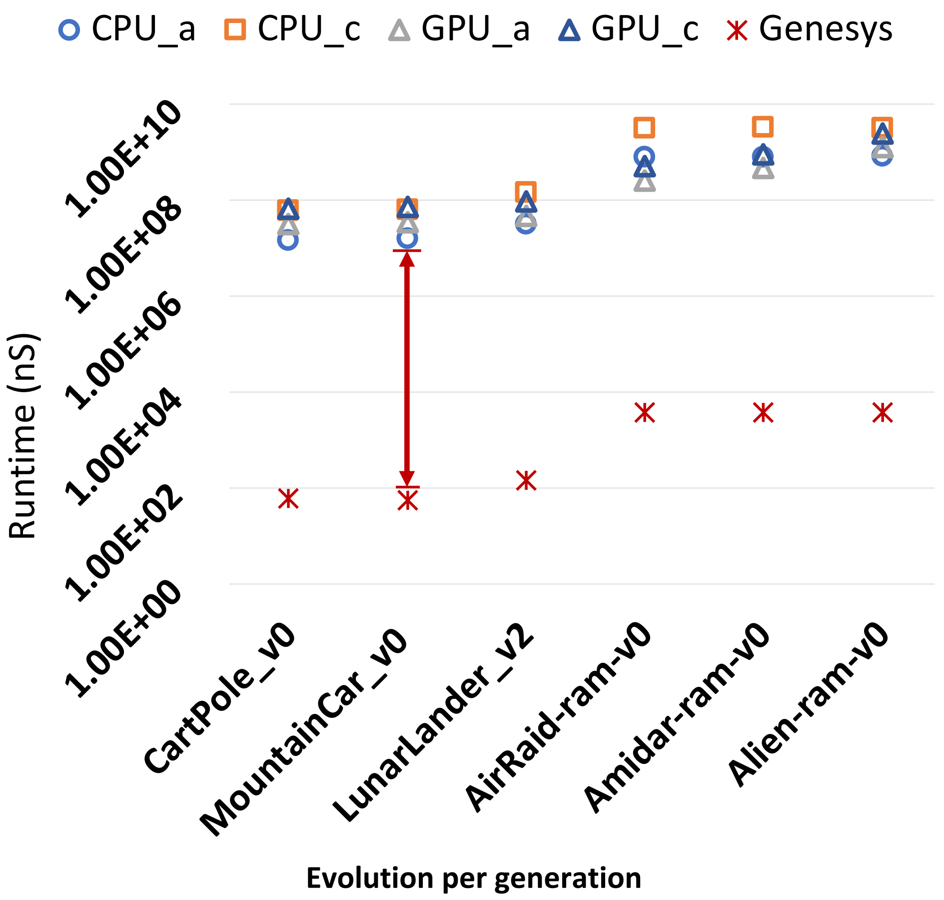
Legend	Inference	Evolution	Platform
CPU_a	Serial	Serial	6th gen i7
CPU_b	PLP	Serial	6th gen i7
GPU_a	BSP	PLP	Nvidia GTX 1080
GPU_b	BSP + PLP	PLP	Nvidia GTX 1080
CPU_c	Serial	Serial	ARM Cortex A57
CPU_d	PLP	Serial	ARM Cortex A57
GPU_c	BSP	PLP	Nvidia Tegra
GPU_d	BSP + PLP	PLP	Nvidia Tegra
GENESYS	PLP	PLP + GLP	GENESYS

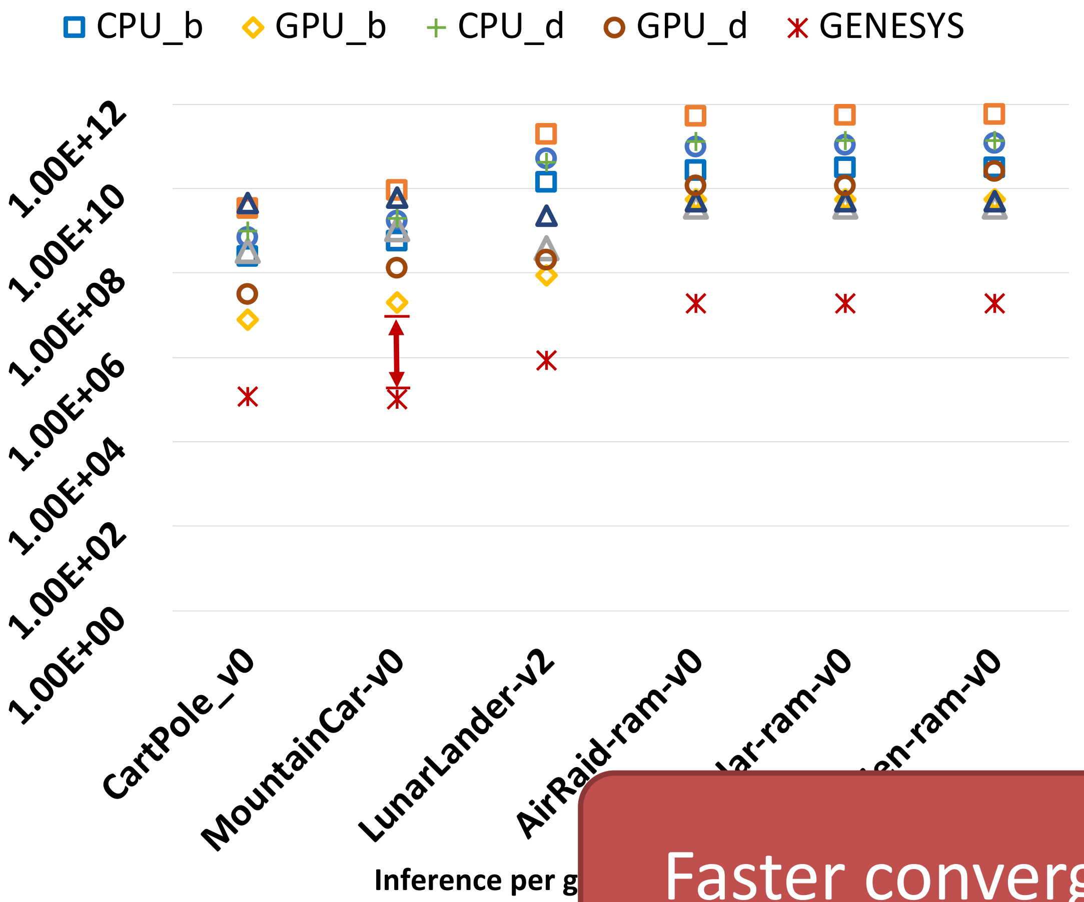
PLP (GLP) - Population (Gene) Level Parallelism
BSP - Bulk Synchronous Parallelism (GPU)

Evaluations: Energy



Evaluations: Runtime

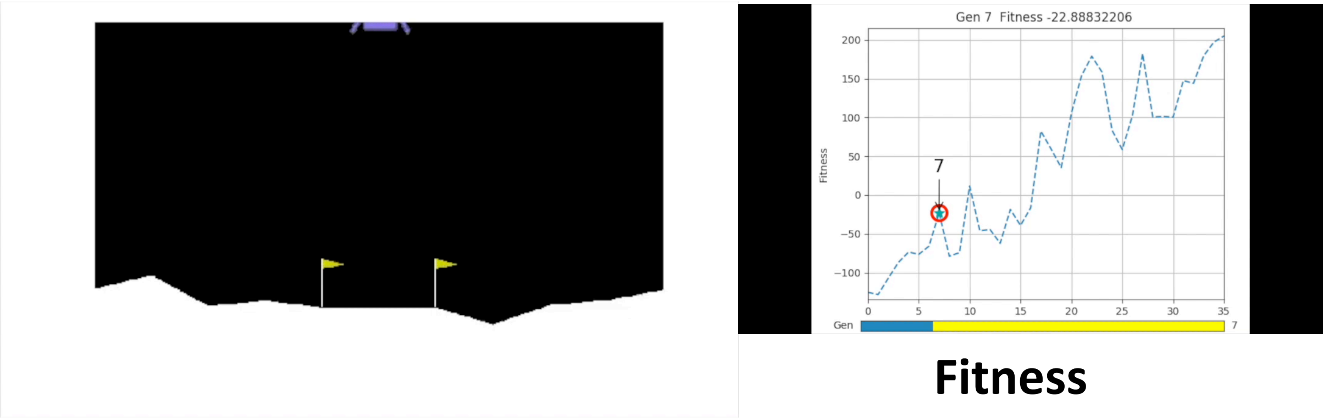




Faster convergence

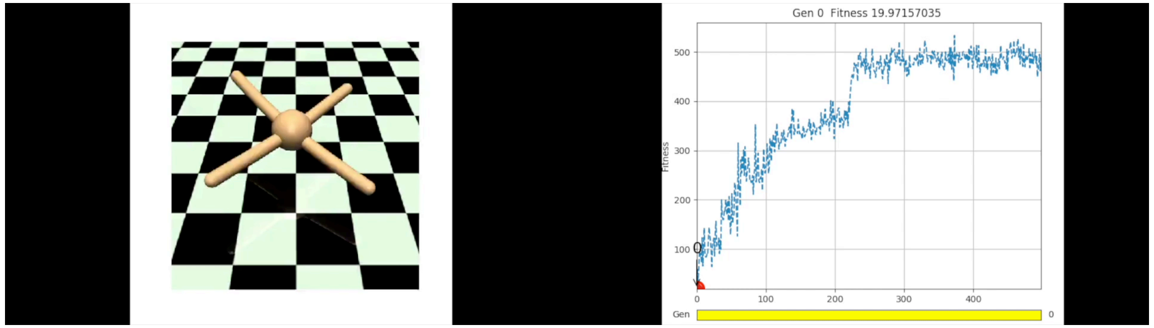
Conclusions

Thank You!



Fitness

Change fitness function



Fitness

Robust, **Scalable** and **Energy** efficient solutions needed for continuous learning

Look **beyond DL and RL**

NEs offer promise

Parallelism **HW friendly**

GeneSys

100x – 100000x energy efficiency and performance

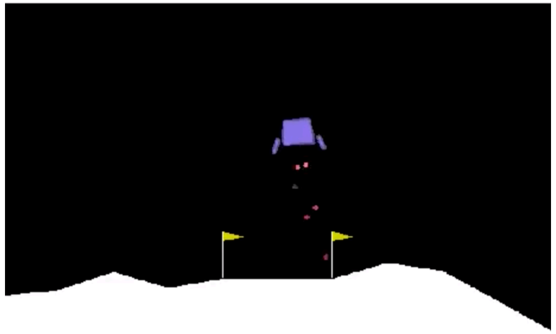
Enables AI solutions for a large gamut of problems

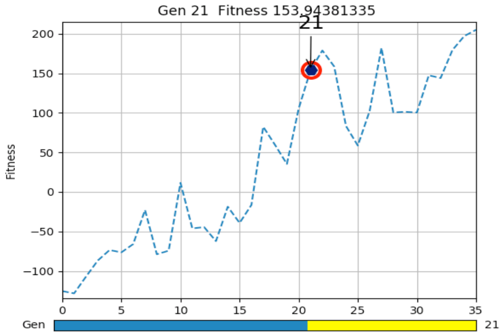
Thank You!

Backup

Conclusions

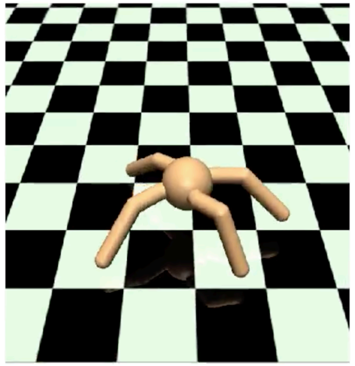
Thank You!

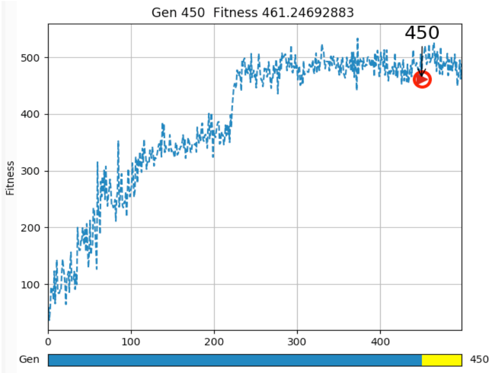




Fitness

Change fitness function





Fitness

Robust, **Scalable** and **Energy** efficient solutions needed for continuous learning

Look **beyond DL and RL**

NEs offer promise

Parallelism **HW friendly**

GeneSys

100x – 100000x energy efficiency and performance

Enables AI solutions for a large gamut of problems

Deep Learning Landscape

POPULAR SCIENCE

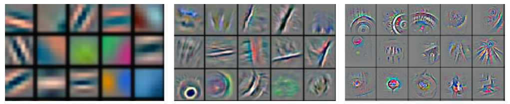
WANT MORE?

SCIENCE | TECH | DIY | GOODS | VIDEO | ROLL THE DICE | SUBSCRIBE

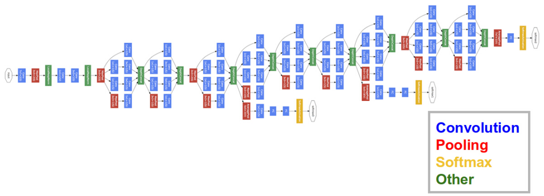
TECHNOLOGY

There are two kinds of AI, and the difference is important

Most of today's AI is designed to solve specific problems



High performanc
cluster



Carefully constructed topology

Wise up, deep learning may never create a general purpose AI

AI and deep learning have been subject to a huge amount of hype. In a new paper, Gary Marcus argues there's been an "irrational exuberance" around deep learning

POPULAR SCIENCE

WANT MORE?

SCIENCE | TECH | DIY | GOODS | VIDEO | ROLL THE DICE | SUBSCRIBE

TECHNOLOGY

There are two kinds of AI, and the difference is important

Most of today's AI is designed to solve specific problems

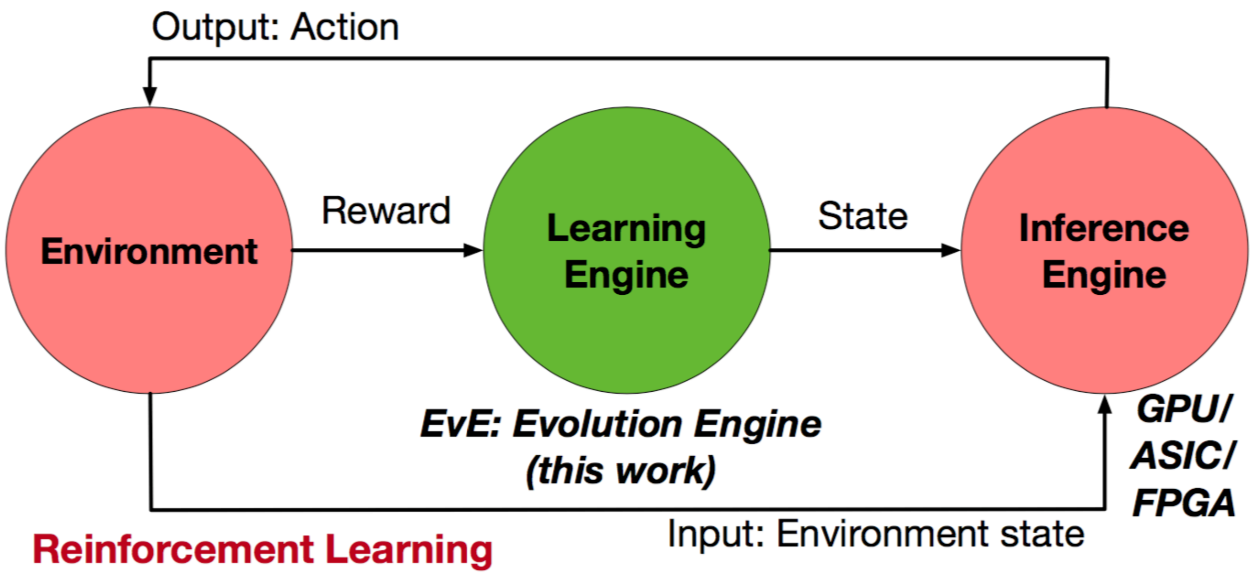
Wise up, deep learning may never create a general purpose AI

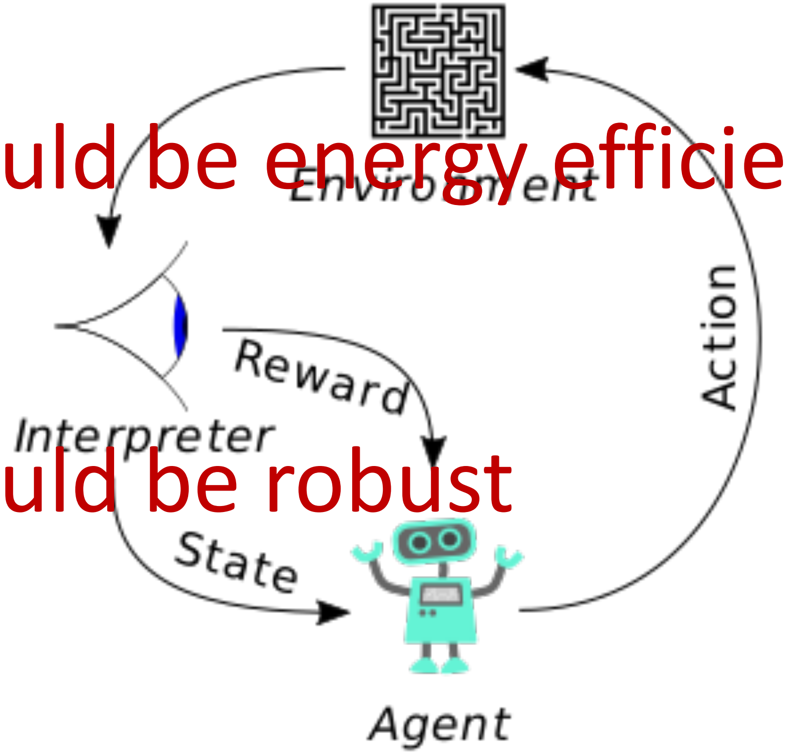
AI and deep learning have been subject to a huge amount of hype. In a new paper, Gary Marcus argues there's been an "irrational exuberance" around deep learning

The next step

What happens when…

Large compute resources are not available?
No labelled dataset?
The problem changes with time?





Should be energy efficient

Should be robust

Reinforcement learning

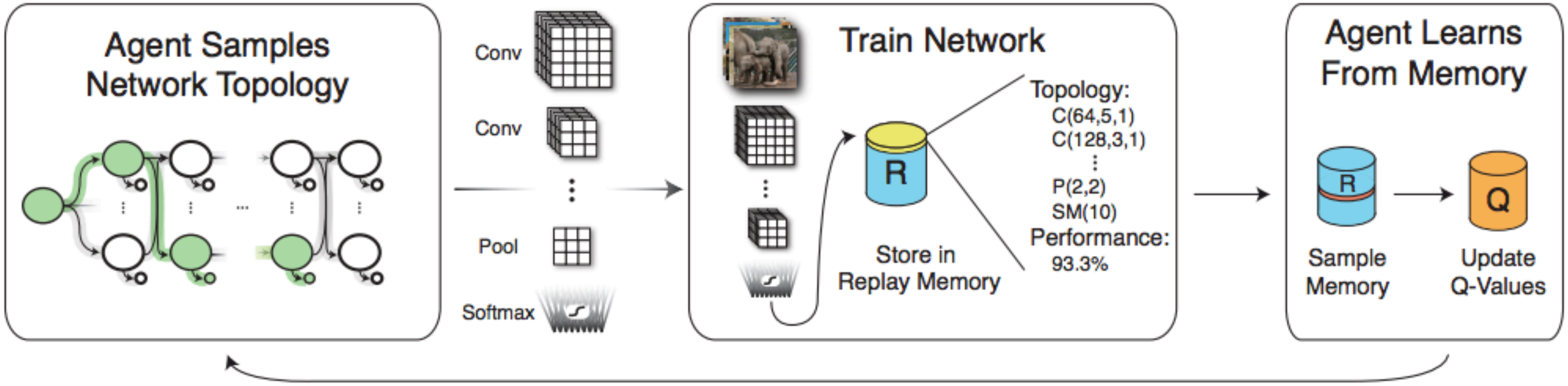
Reinforcement Learning for Topology Generation

DESIGNING NEURAL NETWORK ARCHITECTURES USING REINFORCEMENT LEARNING

Bowen Baker, Otkrist Gupta, Nikhil Naik & Ramesh Raskar
Media Laboratory
Massachusetts Institute of Technology
Cambridge MA 02139, USA
{bowen, otkrist, naik, raskar}@mit.edu

Key Points

- Uses a Q learning agent to learn the optimal policies
- States are different convolution layer types, and policy is the task of selecting next layer
- Child topologies are trained for a few epochs before inference is performed to get reward values.



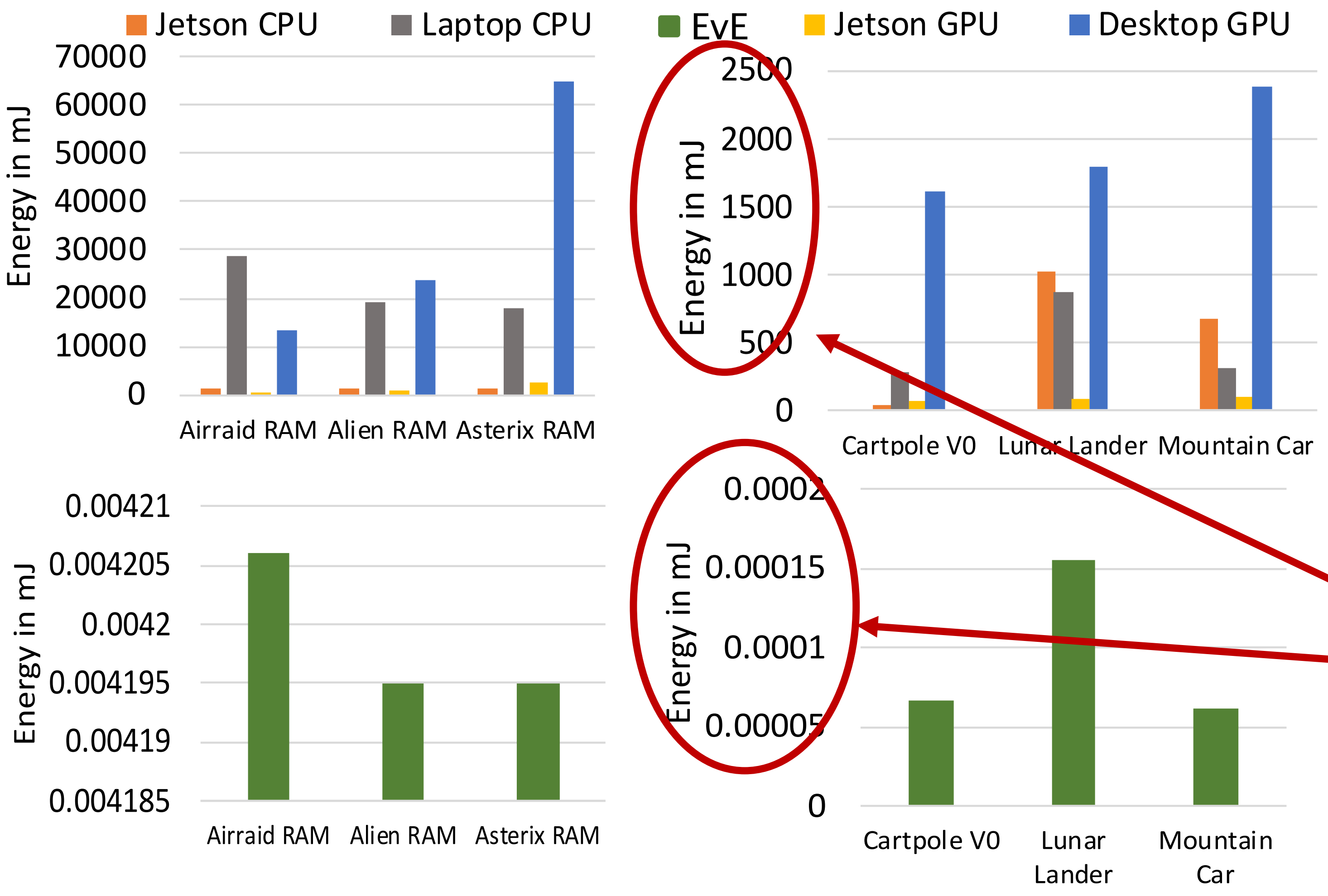
Conventional RL: Challenges

- Deep neural networks estimate the environment
  - Deep Q network (DQN): Generates Q values
  - Policy gradient: Predicts policies

- Each update results in a **backpropagation**
  - **Lots of compute, lot of hyper parameter tuning**
  - Lots of gradient calculation **Not Scalable**
  - Store activations or recalculate **Huge memory footprint**

Not energy efficient

Evaluations



Comparison points

1) Intel i7- 6th gen
2) ARM cortex A57 on nVidia Jetson TX-2
3) nVidia GTX 1080 – Pascal
4) nVidia Tegra on Jetson TX-2 - Pascal

Post synthesis on 15nm PDK for EvE

Orders of magnitude difference

Terminology

Neural Network expressed as a ***graph***

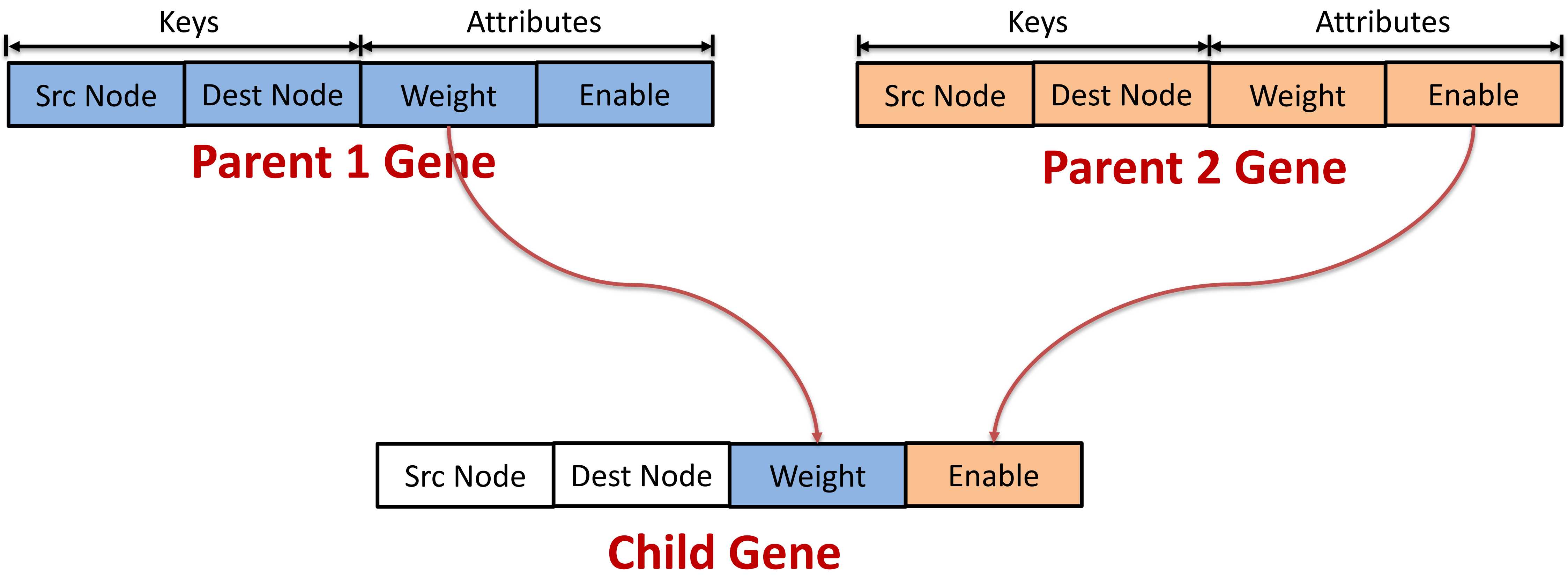
Gene

Data structure representing a vertex (node) or an edge (connection) in the graph

Connection	Src Node	Dest Node	Weight	Enable
Node	Node ID	Activation	Bias	Enable

Operations in NEAT

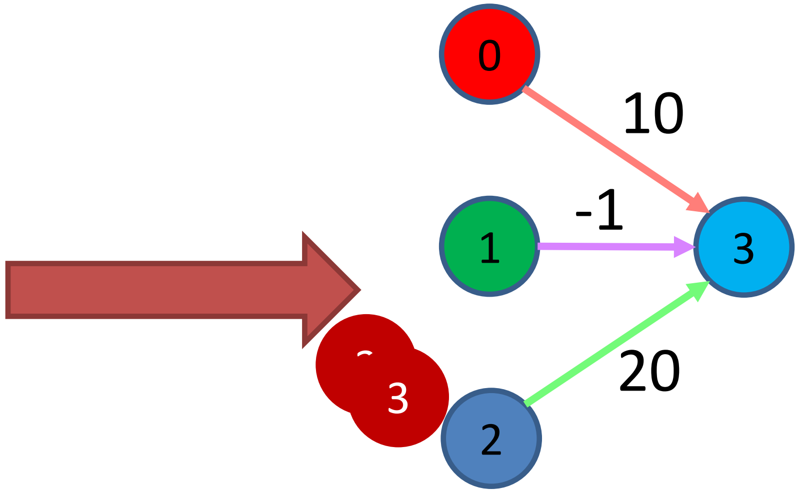
Crossover



Terminology

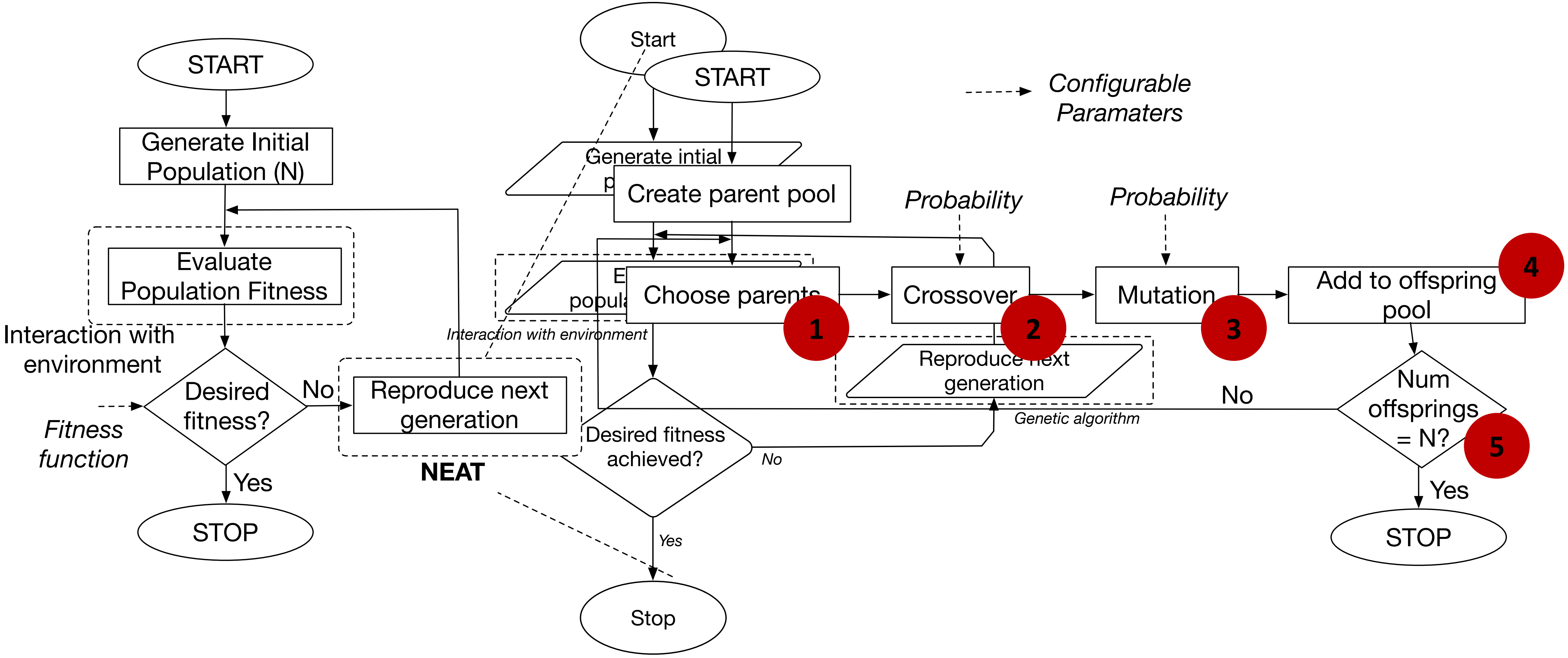
Genome Collection of genes representing the entire neural network

0	Relu	0	Yes
1	Relu	0	Yes
2	Relu	0	Yes
3	Relu	0	Yes
0	3	10	Yes
1	3	-1	Yes
2	3	20	Yes



Each genome represents one neural network

Evolution of Neural Networks

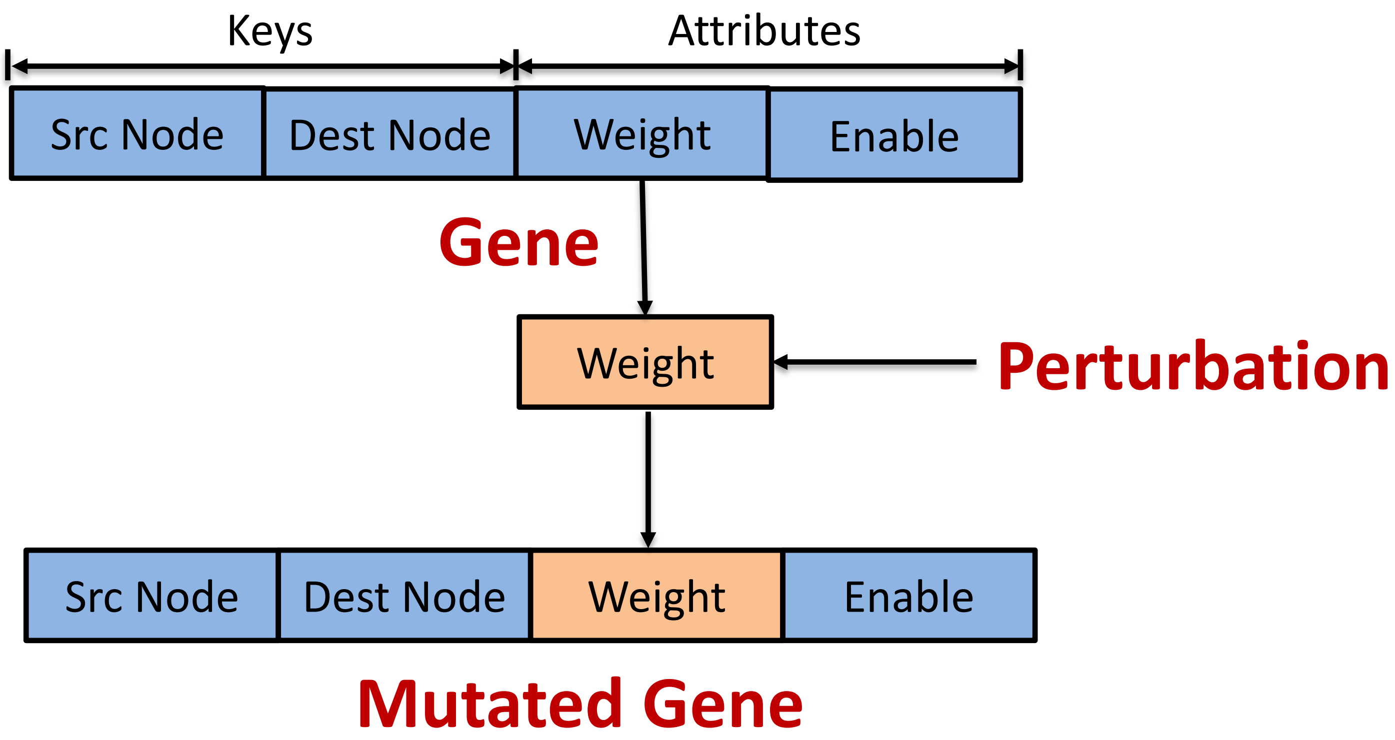


Outline

- **Motivation**
- **Evolutionary Algorithm**
  - NEAT Algorithm
  - Characterizing NEAT
- **Microarchitecture**
- **Evaluations**
  - Implementation
  - Results

Operations in NEAT

Mutation



Addition mutation

Add new node

- Break an existing connection and insert node
- Creates 3 new genes and replaces one existing

Add new connection

- Select valid source and destination and create new gene with default weight

Deletion mutation

Delete connection

- Similar to disabling weight but entry is obliterated

Delete node

- Should also delete dependent connections

Interconnect