Distributed Meta Optimization of Reinforcement Learning Agents

Greg Heinrich, Iuri Frosio - GTC San Jose, March 2019
## Contents

- Introduction to Reinforcement Learning
- Introduction to Metaoptimization (on distributed systems) / Maglev
- Metaoptimization and Reinforcement Learning (on distributed systems)
- HyperTrick
- Results
- Conclusion
GPU-Based A3C for Deep Reinforcement Learning (RL)

keywords: GPU, A3C, RL

M. Babaeizadeh, I. Frosio, S. Tyree, J. Clemons, J. Kautz, Reinforcement Learning through Asynchronous Advantage Actor-Critic on a GPU, ICLR 2017 (available at

Open source implementation:
GPU-BASED A3C FOR DEEP REINFORCEMENT LEARNING

Learning to accomplish a task

Image from www.33rdsquare.com
GPU-BASED A3C FOR DEEP REINFORCEMENT LEARNING

Definitions

\[ S_t = \pi(S_t) \]

\[ R_t \]

\[ a_t = \pi(S_t) \]

✓ Environment
✓ Agent
✓ Observable status \( S_t \)
✓ Reward \( R_t \)
✓ Action \( a_t \)
✓ Policy \( a_t = \pi(S_t) \)
Definitions

\[ S_t, R_t \]

\[ a_t = \pi(S_t) \]

Deep RL agent

GPU-BASED A3C FOR DEEP REINFORCEMENT LEARNING
GPU-BASED A3C FOR DEEP REINFORCEMENT LEARNING

Definitions

\[ a_t = \pi(S_t) \]

\[ \Delta \pi(\cdot) \]

\[ \sim R_t \]

\[ S_t, R_t \]

\[ S_t \]

\[ R_0, R_1, R_2, R_3, R_4 \]
Objective: maximize expected discounted rewards

Value of a state

\[ v_\pi(s) = \mathbb{E}_\pi \left[ R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s \right] \]

Expected

Reward discounted

Given that state

The role of \( \gamma \): short or far-sighted agents

\( 0 < \gamma < 1 \), usually 0.99
GPU-Based A3C for Deep Reinforcement Learning (RL)

keywords: GPU, A3C, RL

GPU-BASED A3C FOR DEEP REINFORCEMENT LEARNING

GPU-Based A3C for Deep Reinforcement Learning (RL)

keywords: GPU, A3C, RL
MAPPING DEEP PROBLEMS TO A GPU

REGRESSION, CLASSIFICATION, …

100% utilization / occupancy

Pear, pear, pear, pear, …
Empty, empty, …
Fig, fig, fig, fig, fig, fig, fig, fig,
Strawberry, Strawberry, Strawberry, …

data

REINFORCEMENT LEARNING

status, reward

labels

action
A3C

Agent 1

\[ a_t = \pi(S_t) \]

Agent 2

\[ a_t = \pi(S_t) \]

Agent 16

\[ a_t = \pi(S_t) \]

\[ t_{\text{max}} \]

Master model
A3C

Agent 1

Agent 2

Agent 16

Small inference batch size (1), low occupancy

Master model
A3C

Agent 1

Agent 2

Agent 16

\[ \Delta \pi(\cdot) \]

Small training batch size (5), low occupancy
A3C

\[ \Delta \pi(\cdot) \]

\[ \pi'(\cdot) \]

Master model

Intense traffic, low utilization

Agent 1

\( a_t = \pi(S_t) \)

Agent 2

\( a_t = \pi(S_t) \)

Agent 16

\( a_t = \pi(S_t) \)

\( t_{\text{max}} \)
GA3C (INFEERENCE)

Agent 1 \( a_t \)

Agent 2 \( a_t \)

Agent N \( a_t \)

\( S_t \)

prediction queue

\( \{a_t\} \)

\( \{S_t\} \)

Master model

Large inference batch size
GA3C (TRAINING)

Large training batch size, avoid model broadcasting

Agent 1

Agent 2

Agent N

Δπ(·)

Master model

{S_t, R_t}

training queue

trainers
GA3C
CPU & GPU UTILIZATION IN GA3C

For larger DNNs - bandwidth limited, do not scale to multiple GPUs!

- CPU for environment simulation
- GPU for inference / training
Role of $t_{\text{max}}$

t_{\text{max}} = 4 [Play to the end - Monte Carlo]
No variance (collected rewards are real)
High bias (we played only once)
One update every $t_{\text{max}}$ frames
Role of $t_{\text{max}}$

$t_{\text{max}} = 2$
Value network
High variance (noisy value network)
Low bias (unbiased net, many agents)
More updates per second

Value network: from here, approximately 2.5

$v_{\pi}(s) = \mathbb{E}_{\pi} \left[ R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s \right]$

$t_{\text{max}}$ affects bias, variance, computational cost (number of updates per second, batch size)
Other parameters and stability

Hyperparameter search in 2015:
The search for the optimal learning rate:

Asynchronous Methods for Deep Reinforcement Learning

Figure 2. Scatter plots of scores obtained by asynchronous advantage actor-critic on five games (Beamrider, Breakout, Pong, Qbert, Space Invaders) for 50 different learning rates and random initializations. On each game, there is a wide range of learning rates for which all random initializations achieve good scores. This shows that A3C is quite robust to learning rates and initial random weights.

GA3C on distributed systems

- RL is unstable, **metaoptimization** for optimal hyperparameters search
- E.g. learning rate may affect the **stability** and **speed of convergence**
- GA3C does not scale to multiple GPUs (bandwidth limited), but ... We can run **parallel instances** of GA3C on a **distributed system**
- The discount factor $\gamma$ affects the **final aim** (short or far-sighted agent)
- The $t_{\text{max}}$ factor affects the **computational cost** and **stability** of GA3C

AGENDA

Contents

Introduction to Reinforcement Learning

Introduction to Metaoptimization (on distributed systems) / Maglev

Metaoptimization and Reinforcement Learning (on distributed systems)

HyperTrick

Results

Conclusion
META OPTIMIZATION

It is as easy as flying a Concorde.

GA3C Agent

- Topology parameters
  - Number of layers and their width
  - Choice of activations

- Training parameters
  - Learning rate
  - Reward decay rate ($\gamma$)
  - back-propagation window size ($t_{max}$)
  - Choice of optimizer
  - Number of training episodes

- Data parameters
  - Environment model.

⇒ Exhaustive search is intractable

Source: Christian Kath
META OPTIMIZATION

How does a standard optimization algorithm fare?

Example: Tree of Parzen Estimators

- Two Parameters, one Metric to minimize.
- Optimization Trade-offs:
  - Exploitation v.s. exploration.
  - Wall time v.s. resource efficiency.
- Optimization packages start with a Random Search.
- Tens of experiments are needed before historical records can be leveraged.

→ Warm Starts are needed to cut down complexity over time.
META OPTIMIZATION
The Need for Diversity

Metric Variance

- Non-determinism makes individual experiments inconclusive.
- A change can only be considered an improvement if it works under a variety of conditions.

Meta Optimization should be part of data scientists’ daily routine.
META OPTIMIZATION
The Complexity of Evaluating Models

Complex Pipelines

• Evaluation cannot be reduced to a single Python function, or Docker container.

⇒ Meta Optimization must be independent of task scheduling.
META OPTIMIZATION
Project MagLev: Machine Learning Platform

Architecture

- Scalable Platform for Traceable Machine Learning Workflows
- Self-Documented Experiments
- Services can be used in isolation, or combined for maximum traceability.
Knowledge Base

- Experiment Data is fully connected.
- Objects are searched through their relationships with others.
- No Information silo.

**META OPTIMIZATION**

**MagLev Experiment Tracking**

<table>
<thead>
<tr>
<th>Knowledge Base</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Experiment Data is fully connected.</td>
</tr>
<tr>
<td>• Objects are searched through their relationships with others.</td>
</tr>
<tr>
<td>➡️ No Information silo.</td>
</tr>
</tbody>
</table>

---

**Model**

<table>
<thead>
<tr>
<th>id</th>
<th>6a-e7-59</th>
</tr>
</thead>
<tbody>
<tr>
<td>job_id</td>
<td>ae-45-4a</td>
</tr>
<tr>
<td>param_set_id</td>
<td>45-4a-26</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
<tr>
<td>dataset_id</td>
<td>ac-73-fc</td>
</tr>
</tbody>
</table>

**Metric**

<table>
<thead>
<tr>
<th>id</th>
<th>69-bd-4c</th>
</tr>
</thead>
<tbody>
<tr>
<td>param_set_id</td>
<td>45-4a-26</td>
</tr>
<tr>
<td>job_id</td>
<td>ae-45-4a</td>
</tr>
<tr>
<td>model_id</td>
<td>6a-e7-59</td>
</tr>
<tr>
<td>name</td>
<td>xentropy</td>
</tr>
<tr>
<td>value</td>
<td>0.01</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
<tr>
<td>dataset_id</td>
<td>ac-73-fc</td>
</tr>
</tbody>
</table>

**Job**

<table>
<thead>
<tr>
<th>id</th>
<th>ae-45-4a</th>
</tr>
</thead>
<tbody>
<tr>
<td>scm</td>
<td>SNA:3487c</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>workflow_id</td>
<td>fb-d5-7a</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
</tbody>
</table>

**Experiment**

<table>
<thead>
<tr>
<th>id</th>
<th>03-ad-24</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>yml</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>project</td>
<td>DormRoom</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
</tbody>
</table>

**Workflow**

<table>
<thead>
<tr>
<th>id</th>
<th>fb-d5-7a</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desc</td>
<td>my_v0.1.1</td>
</tr>
<tr>
<td>spec</td>
<td>...</td>
</tr>
<tr>
<td>project</td>
<td>DormRoom</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
</tbody>
</table>

**Dataset**

<table>
<thead>
<tr>
<th>id</th>
<th>ac-73-fc</th>
</tr>
</thead>
<tbody>
<tr>
<td>vdisk</td>
<td>zzz://...</td>
</tr>
<tr>
<td>project</td>
<td>DormRoom</td>
</tr>
<tr>
<td>creator</td>
<td>bob</td>
</tr>
<tr>
<td>created</td>
<td>2018-09-28</td>
</tr>
</tbody>
</table>
META OPTIMIZATION

Typical Setup

Main SDK Features

• All common parameter types.
• Early-termination methods.
• Standard + custom parameter picking methods.
## Contents

- Introduction to Reinforcement Learning
- Introduction to Metaoptimization (on distributed systems) / Maglev
- Metaoptimization and Reinforcement Learning (on distributed systems)
- HyperTrick
- Results
- Conclusion
META OPTIMIZATION + GA3C

Hyper Parameters and Preview of results.

- Learning Rate: log uniform distribution over \([1e^{-5}, 1e^{-2}]\) interval.
- \(t_{\text{max}}\): quantized (q=1) log uniform distribution over \([2, 100]\) interval.
- \(\gamma\): one of \(\{0.9, 0.95, 0.99, 0.995, 0.999, 0.9995, 0.9999\}\)

<table>
<thead>
<tr>
<th>Game</th>
<th>Episodes per Phase</th>
<th>(N_p)</th>
<th>(r)</th>
<th>(\alpha) (min[(\alpha)], E[(\alpha)])</th>
<th>Score (GA3C)</th>
<th>Score (HyperTrick)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxing</td>
<td>2500</td>
<td>10</td>
<td>25%</td>
<td>48.2% (18.87%, 37.75%)</td>
<td>92</td>
<td>98</td>
</tr>
<tr>
<td>Centipede</td>
<td>2500</td>
<td>10</td>
<td>25%</td>
<td>52.2% (18.87%, 37.75%)</td>
<td>7386</td>
<td>8707</td>
</tr>
<tr>
<td>Ms Pacman</td>
<td>2500</td>
<td>10</td>
<td>25%</td>
<td>46.1% (18.87%, 37.75%)</td>
<td>1978</td>
<td>2112</td>
</tr>
<tr>
<td>Pong</td>
<td>2500</td>
<td>5</td>
<td>25%</td>
<td>59.1% (30.51%, 61.02%)</td>
<td>18</td>
<td>18</td>
</tr>
</tbody>
</table>
# Contents

- Introduction to Reinforcement Learning
- Introduction to Metaoptimization (on distributed systems) / Maglev
- Metaoptimization and Reinforcement Learning (on distributed systems)
- HyperTrick
- Results
- Conclusion
HYPERTRICK
Early Termination Without Compromise

Successive Halving (SH) \cite{1}

Terminate $\frac{1}{2}$ of workers every $N \cdot 2^P$ units of work ($P$ is a phase index).

- Requires synchronization between workers.
- Assumes relative perf over time is constant.

\cite{1} https://arxiv.org/abs/1502.07943
## HYPERTICK

**Early Termination Without Compromise**

<table>
<thead>
<tr>
<th>Successive Halving (SH) [1]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminate $\frac{1}{2}$ of workers every $N \times 2^P$ units of work ($P$ is a phase index).</td>
</tr>
<tr>
<td>- Requires synchronization between workers.</td>
</tr>
<tr>
<td>- Assumes relative perf over time is constant.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HyperBand [2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run several instances of SH in parallel with different values of $N$.</td>
</tr>
<tr>
<td>- Does not assume constant relative perf but still requires synchronization.</td>
</tr>
</tbody>
</table>

# Early Termination Without Compromise

## Successive Halving (SH) [1]

- Terminate $\frac{1}{2}$ of workers every $N \times 2^P$ units of work ($P$ is a phase index).
- Requires synchronization between workers.
- Assumes relative perf over time is constant.

## HyperTrick [3]

### Parameters:
- $N$: total number of workers.
- $r$: eviction rate per phase $P$.

### Worker:
```
while maglev.should_continue():
    work()
    maglev.report_metrics()
```

### MagLev:
- Let earliest $N(1 - \sqrt{r})(1 - r^P)$ run.
- Others are terminated, if in bottom $\sqrt{r}$ quantile.
- Expected number of workers at end of phase $P$ is $N(1 - r)^P$

## HyperBand [2]

- Run several instances of SH in parallel with different values of $N$.
- Does not assume constant relative perf but still requires synchronization.

---

HYPERTRECK v.s. SUCCESSIVE HALVING

16 workers on 6 nodes running up to 4 iterations

Successive Halving

HyperTrick

Must support preemption

No context switches, shorter wall time
Contents

Introduction to Reinforcement Learning
Introduction to Metaoptimization (on distributed systems) / Maglev
Metaoptimization and Reinforcement Learning (on distributed systems)
HyperTrick
Results
Conclusion
PONG
Videos of Trained Agents

$\gamma = 0.9$ (short-sighted)  

$\gamma = 0.995$ (far-sighted)
HYPERTROICK
Terminate Underperformers

Boxing

Centipede

Pacman

Pong

unpainted area = saved resources
HYPERTRICK
Comparison Against HyperBand

Cluster occupancy (Pong)

Timelines (Pong)

Best Score v.s. Wall Time (Pong)
## META OPTIMIZATION

Experimental Comparison of HyperTrick v.s. HyperBand

<table>
<thead>
<tr>
<th>Game</th>
<th>Method</th>
<th>Best Score</th>
<th>Total Wall Time</th>
<th>Time To Best Score</th>
<th>Best Config</th>
<th>$T_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boxing</td>
<td>HyperBand</td>
<td>96</td>
<td>51h</td>
<td>29h</td>
<td>$3.3e^{-4}$</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>HyperTrick</td>
<td>95</td>
<td>38h</td>
<td>13h</td>
<td>$3.3e^{-4}$</td>
<td>13</td>
</tr>
<tr>
<td>Centipede</td>
<td>HyperBand</td>
<td>8521</td>
<td>42h</td>
<td>2h</td>
<td>$5.4e^{-3}$</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td>HyperTrick</td>
<td>8667</td>
<td>38h</td>
<td>29h</td>
<td>$1.2e^{-4}$</td>
<td>33</td>
</tr>
<tr>
<td>Pacman</td>
<td>HyperBand</td>
<td>2456</td>
<td>31h</td>
<td>26h</td>
<td>$1.6e^{-4}$</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>HyperTrick</td>
<td>2243</td>
<td>27h</td>
<td>16h</td>
<td>$1.6e^{-4}$</td>
<td>73</td>
</tr>
<tr>
<td>Pong</td>
<td>HyperBand</td>
<td>17.5</td>
<td>48h</td>
<td>47h</td>
<td>$2.0e^{-3}$</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>HyperTrick</td>
<td>17.8</td>
<td>39h</td>
<td>22h</td>
<td>$5.9e^{-4}$</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 3: HyperBand vs HyperTrick results on four Atari games.
## Take Away

RL is unstable => meta optimization is useful.

Hypertrick, a new algorithm for metaoptimization.

GA3C + HyperTrick + Maglev is effective.


GA3C: [https://github.com/NVlabs/GA3C](https://github.com/NVlabs/GA3C)

MagLev info: Yehia Khoja (ykhoja@nvidia.com)

## Related Talks

<table>
<thead>
<tr>
<th>Session</th>
<th>Date</th>
<th>Time</th>
<th>Title</th>
<th>Speaker</th>
</tr>
</thead>
<tbody>
<tr>
<td>S9649</td>
<td>Wed</td>
<td>9:00am</td>
<td>NVIDIA's AI Infrastructure for Self-Driving Cars</td>
<td>Clement Farabet</td>
</tr>
<tr>
<td>S9613</td>
<td>Wed</td>
<td>10:00am</td>
<td>Deep Active Learning</td>
<td>Adam Lesnikowski</td>
</tr>
<tr>
<td>S9911</td>
<td>Wed</td>
<td>2:00pm</td>
<td>Determinism In Deep Learning</td>
<td>Duncan Riach</td>
</tr>
<tr>
<td>S9630</td>
<td>Thu</td>
<td>2:00pm</td>
<td>Scaling Up DL for Autonomous Driving</td>
<td>Jose Alvarez</td>
</tr>
<tr>
<td>S9987</td>
<td>Thu</td>
<td>9:00am</td>
<td>MagLev: production-grade AI platform...</td>
<td>Divya Vavili</td>
</tr>
</tbody>
</table>