Documentation Index

Fetch the complete documentation index at: /docs/llms.txt

Use this file to discover all available pages before exploring further.

Before you Begin

- IO Overview
  - Company Origins
  - IO Coin
- Get Started
- Latest Updates

Block Rewards

- Overview
- Monitor Block Rewards

IO Architecture

- Architectural Layers
- IO Tunnels
- IO Network
- Glossary
- FAQ

On this page

Moore's Law is Dead
Domain-Specific Hardware is Not Enough

Archived copy · cryp2nova.com

Training
Tuning
Why we need distributed computing for AI
References:

IO Overview

Company Origins

The origins of io.net.

Prior to June 2022, io.net was exclusively devoted to the development of institutional-grade quantitative trading systems for both the United States stock market and the cryptocurrency market. Our primary challenge was constructing the infrastructure necessary to accommodate our complex needs, which included a robust backend trading system with significant computational power. Our trading strategies, bordering on high-frequency trading (HFT), necessitated real-time monitoring of the tick data of over 1,000 stocks and 150 cryptocurrencies. HFT is a method of trading that uses powerful computer programs to transact a large number of orders in fractions of a second. It uses complex algorithms to analyze multiple markets and execute orders based on market conditions. Furthermore, our system had to dynamically backtest and adjust algorithm parameters for each asset in real-time, while also being optimized to facilitate trading for more than 30,000 individual clients across ETrade.com, Alpaca.markets, and Binance.com, maintaining a latency below 200 milliseconds from market events to system reaction on client account for order execution. Achieving such an infrastructure would typically require a dedicated team of MLOps and DevOps professionals. However, our discovery of Ray.io, an open-source library used by OpenAI to distribute GPT-3/4 training across over 300,000 CPUs and GPUs (source) revolutionized our approach and streamlined our infrastructure management. and increased our speed to build this backend from +6 months to less than 60 days . After integrating Ray into our backend and preparing to deploy the application on a cluster of GPU & CPU workers to handle our substantial compute power, we faced the wall of price for running such system due to overpriced GPU on-demand cloud providers. For instance, an NVIDIA A100 card was priced at over $80 USD/day per card. We needed more than 50 of these cards to run on average 25 days/month, amounting to $80 x 50 card x 25 day = 100K USD/month. This posed a serious challenge for us as also for other self-funded startups in the AI/ML industry. Even with such high prices, compute requirements for AI apps have been doubling every 3 months, 10x every 18 months

CRYP2NOVA

Archived copy · cryp2nova.com

Distributed applications have been around for over five decades, starting with the emergence of computer networks like ARPANET. Over the years, developers have utilized distributed systems to scale applications and services, including large-scale simulations, web serving, and big data processing. Nonetheless, distributed applications have generally been the exception rather than the rule. Even today, most undergraduate students complete only a handful of projects involving distributed applications, if any. This landscape is quickly changing as distributed applications are on track to become the norm. Two primary trends drive this transformation: the end of Moore's Law and the skyrocketing computational demands of new machine learning applications. Consequently, a rapidly widening gap between application demands and single-node performance is leaving us no choice but to distribute these applications.

Moore's Law is Dead

For the past 40 years, Moore's Law has driven the unprecedented growth of the computer industry. According to this law, processor performance doubles every 18 months. However, performance growth has slowed to a meagre 10-20% over the same period. Although Moore's Law may have ended, the demand for increased computing power has increased. In response, computer architects have shifted their focus to developing domain-specific processors that prioritize performance over generality.

Domain-Specific Hardware is Not Enough

As the name suggests, domain-specific processors are optimized for specific workloads, sacrificing generality for performance. Deep learning is a prime example of such a workload, revolutionizing various application domains, including financial services, industrial control, medical diagnosis, manufacturing, system optimization, etc. Companies have raced to create specialized processors, like Nvidia's GPUs and Google's TPUs, to support deep learning workloads. While accelerators such as GPUs and TPUs increase computational power, they only extend Moore's Law further into the future, rather than fundamentally increasing the rate of improvement. The Triple Whammy of Deep Learning Application Demand: Machine learning applications' demands are growing at an astonishing pace. Here are three key workloads as examples:

Training

According to a renowned OpenAI blog post, the computation required to achieve state-of-the-art machine learning results has roughly doubled every 3.4 months since 2012. This equates to an increase of almost 40x every 18 months, which is 20x more than Moore's Law! Thus, even without the end of Moore's Law, it would fall significantly short of meeting the demands of these applications. This explosive growth isn't exclusive to niche machine-learning applications like AlphaGo. Similar trends are evident in mainstream applications like computer vision and natural language processing. For example, comparing the computational resources required by the seq2seq model from 2014 to a pretraining approach on tens of billions of sentence pairs from 2019 reveals a ratio of over 5,000x. This corresponds to an annual increase of 5.5x. These figures overshadow Moore's Law, which suggests an increase of only 1.6x per year.

Tuning

Archived copy · cryp2nova.com

The situation is further exacerbated by the fact that models are not trained just once. The quality of a model often depends on various hyperparameters, such as the number of layers, hidden units, and batch size. To find the best model, developers must search through different hyperparameter settings. This process, called hyperparameter tuning, can be resource-intensive. For instance, RoBERTa, a robust technique for pretraining NLP models, uses at least 17 hyperparameters. Assuming a minimal two values per hyperparameter, the search space consists of over 130K configurations. Even partially exploring this space requires vast computational resources. Another example of a hyperparameter tuning task is neural architecture search, which automates the design of artificial neural networks by testing different architectures and selecting the best-performing one. Researchers report that designing even a simple neural network can take hundreds of thousands of GPU computing days. Simulations While deep neural network models can typically leverage advances in specialized hardware, not all ML algorithms can. In particular, reinforcement learning algorithms involve numerous simulations. Due to their complex logic, these simulations are best executed on general-purpose CPUs (with GPUs only used for rendering), meaning they don't benefit from recent advances in hardware accelerators. For example, in a recent blog post, OpenAI reported using 128,000 CPU cores and just 256 GPUs (i.e., 500x more CPUs than GPUs) to train a model capable of defeating amateurs at Dota 2. While Dota 2 is just a game, we're witnessing a surge in the use of simulations for decision-making applications, with startups like Pathmind, Prowler, and Hash.ai emerging in this area. As simulators strive for increasingly accurate environmental modelling, their complexity rises, adding another multiplicative factor to the computational complexity of reinforcement learning.

Why we need distributed computing for AI

Big data and AI are rapidly transforming our world. While technological revolutions bring risks, we see immense potential for this revolution to enhance our lives in ways we couldn't have imagined just a decade ago. However, to realize this promise, we must overcome the massive challenges posed by the rapidly growing gap between application demands and our hardware capabilities. To bridge this gap, distributing applications appears to be the only viable solution. This necessitates new software tools, frameworks, and curricula to train and enable developers to build such applications, marking the beginning of a thrilling new era in computing. At io.net, we develop innovative tools and distributed systems like Ray to guide application developers into this new era.

References:

- [1] Avg market price: https://www.paperspace.com/pricing
- [2] https://arxiv.org/pdf/2202.05924.pdf
- [3] https://businessolution.org/gpt-3-statistics/
- [4] https://research.ark-invest.com/hubfs/1_Download_Files_ARK-Invest/White_Papers/ARK_BigIdeas2022.pdf?hsCtaTracking=217bbc93-a71a-4c2b-9959-0842b6fe301c%7C2653a4d0-af35-42f0-853a-c5f90f002abb

Was this page helpful?

Yes No

CRYP2NOVA

Archived copy · cryp2nova.com

Archived copy · cryp2nova.com

Archived copy · cryp2nova.com

Ask a question...

⌘I

Powered by



This documentation is built and hosted on Mintlify, a developer documentation platform

CRYP2NOVA

Archived copy · cryp2nova.com