LAPORAN PERANCANGAN PROSESOR REINFORCEMENT LEARNING BERBASIS SYSTEM ON CHIP

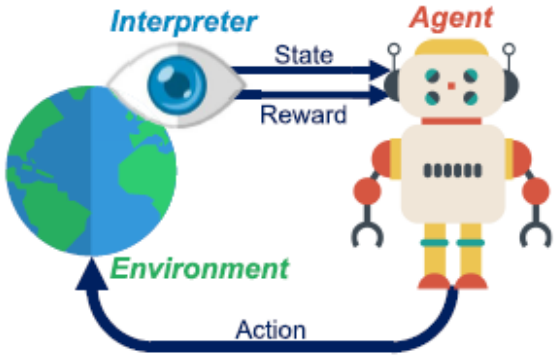
MINGGU 1

Dismas Widyanto (13218065)
Tanggal: 20/08/2021 – 03/09/2021
Merdaka Belajar Kampus Merdeka

1. AN EFFICIENT HARDWARE IMPLEMENTATION OF REINFORCEMENT LEARNING: THE Q-LEARNING ALGORITHM

1.1 INTRODUCTION

Paper ini membahas mengenai arsitektur prosesor yang lebih effisien dalam mengimplementasikan Reinforcement Learning menggunakan algoritma Q-Learning. Reinformcement Learning merupakan bagian dari Machine Learning yang melatih suatu agent untuk melakukan tugas tertentu.



Gambar 1-1 Framework Reinforcement Learning

Agent melakukan action yang berpengaruh terhadap environment. Kemudian interpreter akan mengamati perubahan pada environment dan memberikan informasi berupa state dan reward kepada agent. Reward atau reinforcement merupakan nilai kualitas dari action terakhir yang dilakukan oleh agent. Reward ini berupa angka positif maupun negative. Proses tersebut dilakukan secara berulang-ulang dengan kondisi tertentu agar agent dapat melakukan tugas yang diberikan.

Proses iterasi yang besar menjadi hambatan tersendiri bagi implementasi Reinforcement Learning. Diperlukan prosesor yang mampu mengolah data dalam jumlah besar dengan waktu singkat dan daya yang rendah. Melalui paper ini, permasalahan tersebut diatasi dengan melakukan optimasi berbagai blok yang digunakan.

Algoritma Q-Learning memanfaatkan Quality Matriks untuk menyimpan nilai. Matriks Q terdiri dari state N dan action Z dengan ukuran Z × N. Proses belajar dilakukan dengan menginisialisasi matriks dengan angka sembarang atau nol, kemudian dilakukan proses iterasi menggunakan persamaan

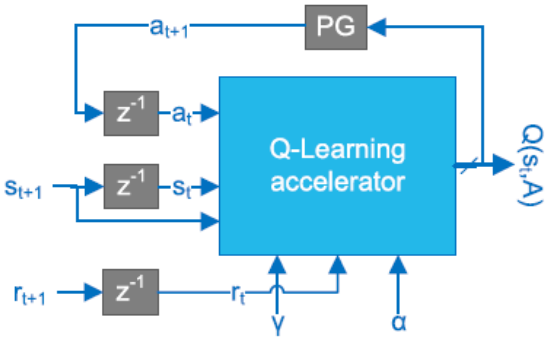
$$Q_{new}(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha\left(r_t + \gamma\, max_a Q(s_{t+1}, a_t)\right)$$

Dengan

- s_t dan s_{t+1} adalah state saat ini dan berikutnya
- a_t dan a_{t+1} adalah action saat ini dan berikutnya
- γ adalah discount factor
- α adalah learning rate
- r_t adalah reward

1.2 ARCHITECTURE

Arsitektur yang diajukan oleh paper ini terdiri dari dua buah blok utama yaitu policy generator (PG) dan Q-Learning accelerator.

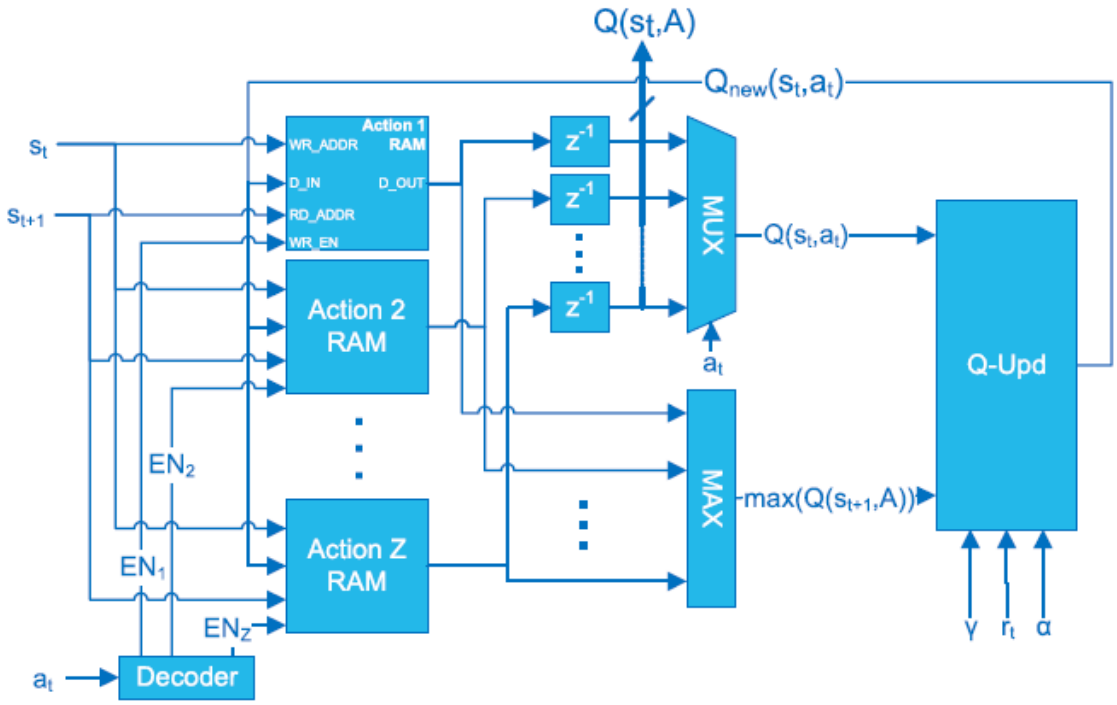


Gambar 1-2 Arsitektur Prosesor yang Diajukan

Prosesor akan menerima masukan berupa reward dan state. Kemudian diberikan delay untuk menghasilkan nilai yang sesuai. Blok PG merupakan blok yang bergantung dari aplikasi

Reinforcement Learning sehingga tidak akan dibahas lebih lanjut dalam paper ini.

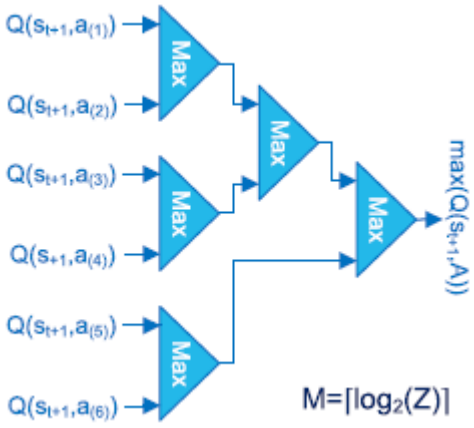
Blok accelerator terdiri dari RAM, MAX, Q-Updater, dan Multiplier.



Gambar 1-3 Arsitektur Blok Q-Learning Accelerator

Blok RAM disusun secara parallel dengan masing-masing blok merepresentasikan setiap action yang mungkin. Masing-masing blok RAM juga akan menyimpan seluruh kolom dari matriks Q.

Limitasi pada accelerator ini terdapat pada blok MAX. Blok ini akan memiliki propagation delay yang berbanding lurus dengan banyaknya action. Salah satu cara untuk mengatasi masalah tersebut adalah menggunakan binary comparator, namun dengan kompensasi area yang diperlukan lebih luas.

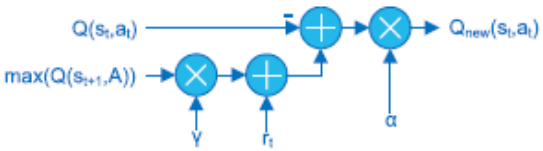


Gambar 1-4 Binary Comparator

Pada blok Q-Updater akan dilakukan perhitungan nilai Q menggunakan persamaan di atas. Persamaan di atas bisa dioptimasi menjadi

$$Q_{new}(s_t, a_t) = Q(s_t, a_t) + \alpha \left(r_t + \gamma \, max_a Q(s_{t+1}, a_t) - Q(s_t, a_t)\right)$$

Persamaan tersebut akan menjadi lebih sederhana ketika diimplementasikan karena hanya memerlukan 2 buah multiplier.



Gambar 1-5 Q-Updater

Blok ini pun masih bisa dioptimasi dengan menggunakan approximated multiplier. Metode ini digunakan karena multiplier memiliki delay propagasi yang cukup besar, sehingga dengan menggunakan pendekatan diharapkan delay bisa makin kecil dengan hasil yang masih sesuai.

1.3 HASIL

Hasil percobaan menunjukkan bahwa arsitektur yang diajukan memberikan performa yang proporsional dengan banyaknya state dan action serta lebar data Q value. Semakin banyak state, data, dan lebar data maka kebutuhan LUT dan dayanya makin besar sedangkan clock yang dihasilkan akan makin kecil.

Penggunaan approximated multiplier pun dapat menurunkan jumlah LUT secara drastis dan mengurangi kebutuhan daya dibandingkan dengan full multiplier.

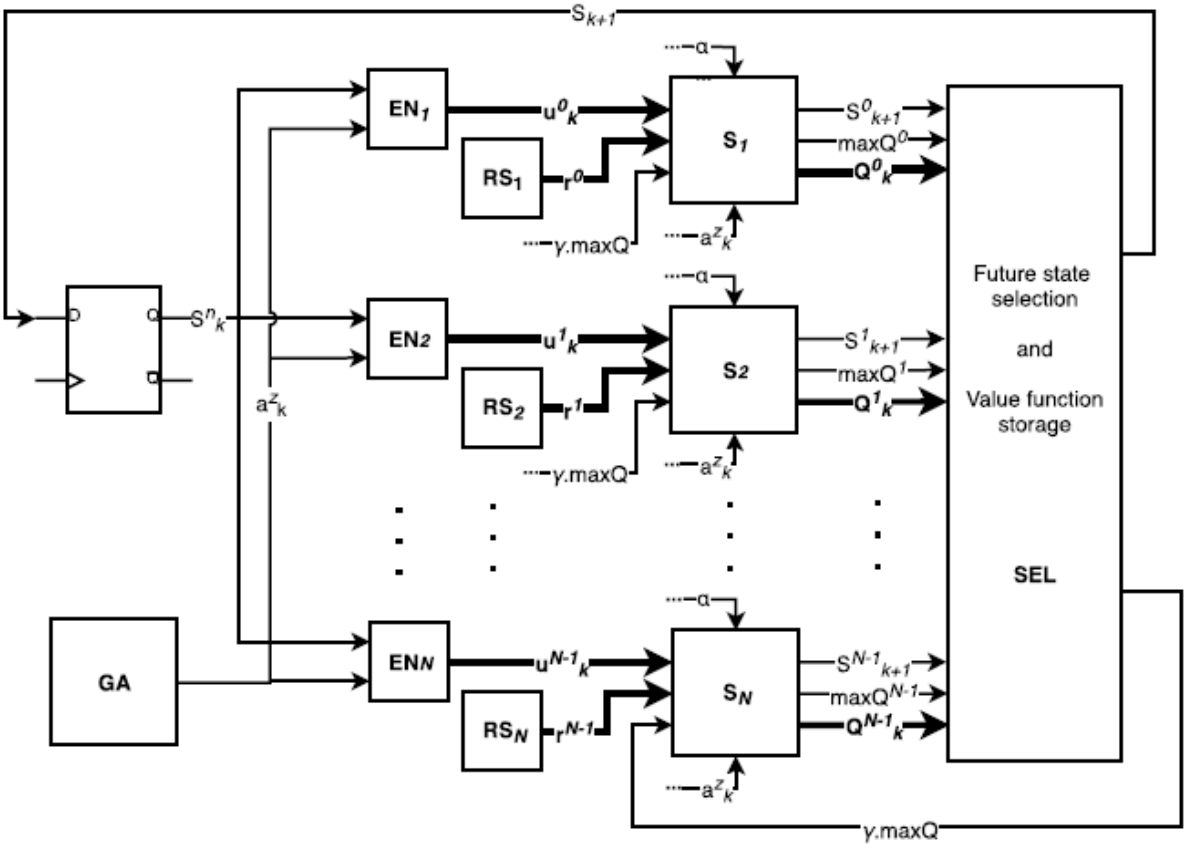
2. PARALLEL IMPLEMENTATION OF REINFORCEMENT LEARNING Q-LEARNING TECHNIQUE FOR FPGA

2.1 INTRODUCTION

Paper ini membahas pengembangan prosesor untuk reinforcement learning yang mampu mengolah data dalam jumlah besar dengan waktu yang cepat dan memiliki daya yang rendah. Aplikasi reinforcement learning yang makin luas menyebabkan kebutuhan prosesor yang lebih andal. Paper ini akan menunjukkan prosesor dengan arsitektur paralel yang diimplementasikan pada FPGA.

2.2 ARCHITECTURE

Arsitektur yang digunakan pada paper ini adalah sebagai berikut



Gambar 2-1 Arsitektur yang Diajukan

Arsitektur tersebut terdiri dari blok GA sebagai blok untuk memilih action, EN sebagai blok untuk menyimpan setiap nilai action pada state tertentu, RS merupakan blok penyimpan nilai reward, S sebagai penghitung nilai Q, dan SEL sebagai blok untuk memilih state selanjutnya.

2.3 HASIL

Hasil dari percobaan ini yaitu komponen yang digunakan, throughput, dan daya yang digunakan akan berbanding secara proporsional dengan banyaknya state, action, dan lebar data yang digunakan. Makin besar state, action, dan lebar datanya maka jumlah komponen, besar throughput, dan jumlah daya akan makin tinggi.

Hasil perbandingan dengan metode lain juga menunjukkan bahwa arsitektur ini mampu meningkatkan throughput yang cukup signifikan. Hasil perbandingan dapat dilihat pada gambar berikut

Reference	Hardware	**Q** matrix size ($N \times Z$)	Reference throughput	Obtained throughput	Speedup
[25]	CPU - 1 processor	528	7400 Sps	12.23 MSps	1655×
	CPU - 2 processors		9411 Sps		1302×
	CPU - 4 processors		172971 Sps		708×
	CPU - 8 processors		12075 Sps		1015×
[19], [22]	FPGA	240	2.34 MSps	15.53 MSps	6.63×
[21]	FPGA	160	25000 Sps	16.74 MSps	669.6×

Gambar 2-2 Hasil Perbandingan dengan Metode Lain

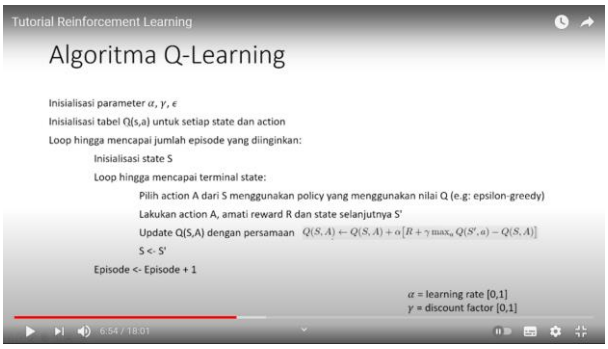
3. TUTORIAL SINGKAT TENTANG REINFORCEMENT LEARNING

Gambar 3-1 Dasar Reinforcement Learning

Reinforcement learning merupakan salah satu bagian dari Machine Learning yang bekerja dengan cara melatih agent untuk menyelesaikan suatu tugas. Agent akan memberikan action yang berpengaruh ke lingkungan sedangkan lingkungan akan memberikan respon dengan cara memberik informasi state dan reward kepada agent berkaitan dengan action yang dilakukan sebelumnya.

Terdapat beberapa elemen pada reinforcement learning yaitu

- Policy: aturan bagi agent dalam mengambil action
- Reward: tujuan agent yaitu memaksimalkan reward
- Value: jumlah reward yang bisa didapat oleh suatu state dalam jangka panjang
- Model environment: model untuk prediksi reward dan state yang diperoleh dari suatu state dan action

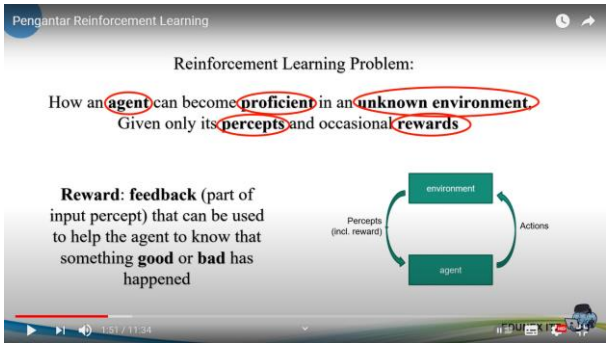


Gambar 3-2 Proses Iteratif Perhitungan Value

Proses perhitungan value secara iteratif mengikuti pseudocode di atas. Parameter akan ditentukan terlebih dahulu, kemudian matriks Q diinisialisasi. Selanjutnya dilakukan iterasi sebanyak episode yang diinginkan. Dalam iterasi tersebut akan dilakukan inisialisasi state lalu iterasi hingga state akhir. Selama proses iterasi state tersebut, nilai dari matrik Q akan selalu diupdate.

Selama proses iterasi, akan dilakukan pemilihan action yang sesuai. Salah satu metode yang bisa digunakan adalah epsilon-greedy. Metode ini akan memilih action dengan value tertinggi dengan probabilitas tertentu.

4. TUTORIAL PENGANTAR REINFORCEMENT LEARNING



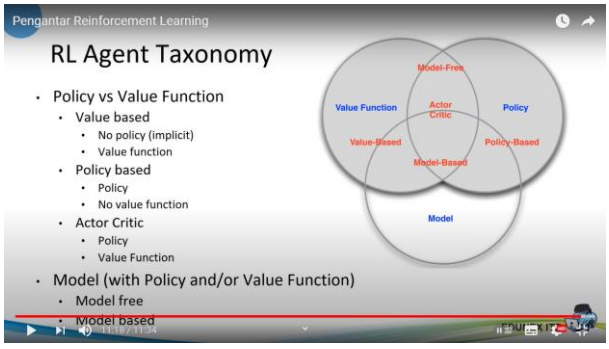
Gambar 4-1 Dasar Reinforcement Learning

Menurut tutorial ini, reinforcement learning merupakan bagaimana sebuah agent dapat menjadi andal pada suatu environment yang belum dikenali hanya dengan diberikan persepsi dan reward.

Algoritma reinforcement learning berbeda dengan machine learning lainnya. Algoritma ini memiliki paradigma

- Feedback hanya berupa reward signal
- Feedback memiliki delay
- Urutan atau waktu sangat penting
- Action yang dilakukan akan berpengaruh ke data yang dilihat

Terdapat beberapa taksonomi agent reinforcement learning, gambar di bawah ini menunjukkan taksonomi tersebut.



Gambar 4-2 Taksonomi Reinforcement Learning

DAFTAR PUSTAKA

[1] Sergio Spanò, Gian Carlo Cardarilli, Luca Di Nunzio, Rocco Fazzolari, Daniele Giardino, Marco Matta, Alberto Nannarelli, Marco Re, "An Efficient Hardware Implementation of Reinforcement Learning: The Q-Learning Algorithm", *Access IEEE*, vol. 7, pp. 186340-186351, 2019.

[2] L. M. D. Da Silva, M. F. Torquato and M. A. C. Fernandes, "Parallel implementation of reinforcement learning Q-learning technique for FPGA", *IEEE Access*, vol. 7, pp. 2782-2798, 2018.