

동적 저궤도 위성 네트워크에서 실시간 라우팅을 위한 FPGA 기반 컨볼루션층 추론 병렬화 기술

김대연*, 이헌철**, 원동식***, 한명훈****

FPGA-based Inference Parallelization of Convolutional Layers for Real-Time Routing in Dynamic LEO Satellite Networks

Daeyeon Kim*, Heoncheol Lee**, Dongshik Won***, and Myoung-Hun Han****

이 연구는 2022년 정부(방위사업청)의 재원으로 국방과학연구소의 지원을 받아 수행된 연구임(UI220033VD)

요약

본 논문에서는 저궤도 위성 네트워크(LEO)의 실시간 라우팅 문제를 다룬다. 기존의 라우팅 알고리즘은 동적인 위성 네트워크 환경에 효과적으로 적응하는 데 한계가 있음을 확인하였다. 이에 따라, 본 연구는 강화학습을 기반으로 한 라우팅 방법을 제시하고, 이를 Dueling Deep Q-Network 모델로 구현하였다. 그러나 위성 상에서의 추론 과정은 제한된 연산 능력으로 인해 실시간 요구 사항을 만족시키기 어려움이 있었고, 이를 해결하기 위해, 본 연구에서는 컨볼루션층의 추론을 병렬화를 통해 추론 속도를 가속화 하는 방법을 제안하였다. 실험 결과, 제안된 방법은 기존 방법 대비 컨볼루션층 수행시간이 약 90.2%, 전체 알고리즘 수행시간은 약 29.0% 단축된 결과를 나타내었다.

Abstract

This paper addresses the real-time routing problem in Low Earth Orbit (LEO) satellite networks. Existing routing algorithms have been found to struggle to adapt effectively to dynamic satellite network environments. As such, this study proposes a reinforcement learning-based routing approach and implements it using a dueling deep Q-network model. However, the inference process on satellites faces challenges in meeting real-time requirements due to limited computational capabilities. To resolve this, we propose an approach that accelerates inference speed by parallelizing the convolutional layer's inference process. Experimental results show that our proposed method has reduced the computation time of the convolutional layer by 90.2% and the total algorithm execution time by 29.0% compared to the existing methods.

Keywords

LEO satellite network routing, deep reinforcement learning, FPGA, parallelization

* 금오공과대학교 IT융복합공학과 석사과정
- ORCID: <https://orcid.org/0009-0009-0558-6876>
** 금오공과대학교 IT융복합공학과 교수(교신저자)
- ORCID: <https://orcid.org/0000-0003-2962-3474>
*** 텔레픽스 주식회사 이사
- ORCID: <https://orcid.org/0009-0000-3521-4680>
**** 국방과학연구소 선임연구원
- ORCID: <https://orcid.org/0009-0004-2571-0153>

· Received: Jul. 11, 2023, Revised: Jul. 28, 2023, Accepted: Jul. 31, 2023
· Corresponding Author: Heoncheol Lee
Dept. of IT Convergence Engineering, School of Electronic Engineering
Kumoh National Institute of Technology, Korea
Tel.: +82-54-478-7476, Email: hlee@kumoh.ac.kr

I. 서 론

저궤도 위성 네트워크는 다양한 산업 분야에서 상당한 양의 데이터를 생산하고 분석하는 현대에 이르러 글로벌 네트워크 연결의 필요성이 점점 커짐에 따라, 낮은 지연 시간 및 높은 통신 효율을 제공함과 동시에 원격 지역에서도 성능이 우수한 연결을 제공함으로써 디지털 격차를 줄이는 것에 기여하고 있다[1]. 다만, 이러한 복잡한 네트워크와 효율적인 관리와 운영은 라우팅 최적화 문제를 포함하여 여러 어려움이 있다.

이러한 문제를 해결하기 위하여 강화학습을 기반으로 한 라우팅 알고리즘 연구들이 진행되었다. 강화학습은 다양한 분야에서 효과적으로 적용되는 기계학습의 한 방법으로[2], 에이전트가 환경과 상호 작용하며 보상이라는 개념을 통해 성능을 개선하는데, 이는 복잡한 환경에서도 보상을 최대화하는, 즉 최적의 라우팅 전략을 찾아낼 수 있게 한다.

하지만 기존의 연구는 정적인 네트워크 환경을 가정하고 있어 실제 동적인 환경에 최적화되지 않은 경우가 있다[3]. 이러한 가정은 저궤도 위성 네트워크와 같이 빠르게 변화하는 환경에서는 적합하지 않을 수 있다. 위성의 위치와 통신 용량, 지구의 트래픽 수요 등은 시간에 따라 변할 수 있기 때문이다[4].

또한, 실시간성은 라우팅에서 응답 시간 최소화, 동적 환경 대응, 신뢰성과 안정성을 보장하는 등 매우 중요한 요소이다. [5][6]는 동적인 환경에 최적화되었지만, 실시간성은 보장되지 않을 수 있다. 특히 자원 사용이 제한적인 온보드 추론 환경에서는 실시간성을 심각하게 고려해야 할 수 있는데, 병렬화를 통한 가속화를 달성할 수 있다[7].

일반적으로 병렬화를 통한 알고리즘 가속화는 표 1에서 정리된 것과 같이 GPU(Graphic Processing Unit) 기반[8][9] 또는 FPGA(Field Programmable Gate Array)를 사용하게 되는데[10][11], 독립적이고 반복적인 연산을 식별하여 병렬화 또는 하드웨어 친화적으로 알고리즘을 수정하는 알고리즘 리팩토링을 통해 이루어진다. 또한, FPGA 기반 가속화의 경우 전력 효율이 GPU 대비 뛰어나다는 장점이 있다[12].

표 1. 알고리즘 가속화 관련 연구

Table 1. Research on algorithm acceleration

Ref.	Platform	Application	Accelerated part
[8]	GPU	Particle filter for ballistic target	Inference
[9]	GPU	Model predictive control (MPC)	Solve linear system
[10]	FPGA	Convolution layer	Convolution operation
[11]	FPGA	Rigid body dynamics	Calculate gradient

따라서 라우팅 알고리즘의 반복성과 독립성을 프로그래밍하여, 저궤도 위성 환경에 탑재되기 적합한 FPGA를 기반으로 라우팅 알고리즘을 가속화 하는 것이 적합하다고 할 수 있다.

본 논문의 주요 기여점은 다음과 같다. 우선, 강화학습 기반 알고리즘의 추론 가속화를 위한 FPGA 구현을 기반 추론 방법을 제안한다. 하드웨어에 친화적으로 재구성된 컨볼루션층은 데이터 종속이 작은 연산을 병렬로 처리하도록 구현하여, 실시간성을 보장하게 한다. 또한, 동시 설계된 시스템의 시뮬레이션을 수행한 결과를 제시하여 제안된 방법의 가속화 효과를 입증하였다.

다음 장에서 저궤도 위성 네트워크의 라우팅을 위한 강화학습 기반 추론 문제가 제시되고, 3장에서는 제안하는 기법을 요소별로 자세하게 기술하였다. 그리고 4장에서는 구현된 시스템의 시뮬레이션 결과와 이에 대한 분석이 이루어진다.

II. 기존 연구의 문제점

저궤도 위성 네트워크에서의 라우팅은 위성의 동적인 움직임, 변동하는 연결성, 제한된 통신 연결 등으로 인해 복잡성이 증가하는 어려운 문제다. 이러한 문제를 해결하기 위해 그림 1과 같이 그리드 환경에서의 MDP(Markov Decision Process)를 이용하는 방안이 제시되었지만[13], 여전히 다수의 문제점이 존재한다.

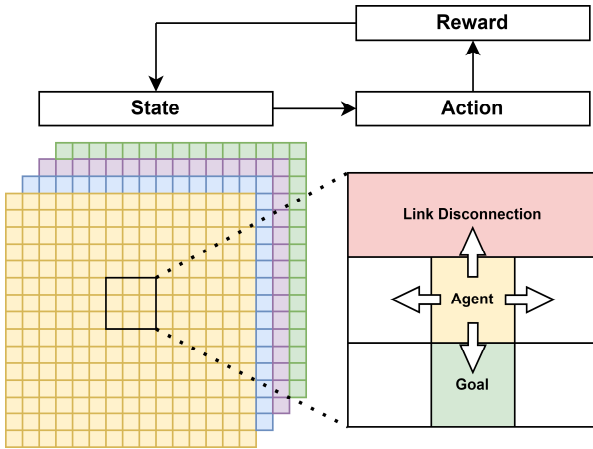


그림 1. 강화학습을 위한 그리드 MDP
Fig. 1. Grid MDP for reinforcement learning

Dueling deep Q-network는 강화학습의 한 형태로 [14], 그리드 환경에서의 MDP의 풀이에 사용될 수 있다. 표 2는 dueling deep Q-network의 의사코드를 나타내며, 행동층과 가치층을 각각의 단계로 분리하여 연산을 수행함으로써 학습 과정을 보다 효율적으로 만들 수 있다.

그러나 dueling deep Q-network 기반의 학습 모델은 복잡한 상태 공간과 행동 공간으로 인해 컴퓨팅 자원의 부족과 긴 학습 시간이 필요하다는 문제가 있다. 또한, 보상 함수의 설계가 어렵고, 이는 성능 향상에 중요한 요소임에도 불구하고 복잡성이 증가하면 학습 과정이 어려워지고 느려질 수 있다.

또 다른 문제점은 온보드 환경에서의 추론에 있다. 위성에 탑재된 시스템의 컴퓨팅 자원은 제한적이기 때문에, 실시간으로 복잡한 알고리즘을 실행하는 것은 매우 어려운 일이다[15]. 특히, dueling deep Q-Network는 컨볼루션 층에서 많은 시간이 소요되는데, 이는 실시간성이 저하되는 것으로 귀결된다. 이러한 문제를 해결하지 않으면, 지연 시간이 증가하고, 패킷 전송률이 감소하는 등의 네트워킹 성능 저하가 발생할 수 있다.

따라서, 저궤도 위성 네트워크에서 효과적인 라우팅을 위해서는 그리드 환경에서의 MDP와 dueling deep Q-network 기반의 학습 모델의 문제점을 극복하고, 그림 2에 나타난 것과 같이 하드웨어 가속화를 통한 온보드 환경에서의 실시간성을 보장하는 방법이 필요하다.

표 2. Dueling deep Q-network의 의사코드
Table 2. Pseudocode for dueling deep Q-network

Algorithm 1: Dueling deep Q-network	
Input:	state
Output:	value
1:	$x \leftarrow conv_1(state)$
2:	$x \leftarrow ReLU(x)$
3:	$x \leftarrow conv_2(x)$
4:	$x \leftarrow ReLU(x)$
5:	$x \leftarrow conv_3(x)$
6:	$x \leftarrow ReLU(x)$
7:	$x \leftarrow conv_4(x)$
8:	$x \leftarrow ReLU(x)$
9:	$x \leftarrow affine_1(x)$
10:	$x \leftarrow ReLU(x)$
11:	$x \leftarrow affine_2(x)$
12:	$x \leftarrow ReLU(x)$
13:	$x_{adv} \leftarrow affine_{adv1}(x), x_{val} \leftarrow affine_{val1}(x)$
14:	$x_{adv} \leftarrow ReLU(x_{adv}), x_{val} \leftarrow ReLU(x_{val})$
15:	$x_{adv} \leftarrow affine_{adv2}(x_{adv}),$ $x_{val} \leftarrow affine_{val2}(x_{val})$
16:	$x_{adv} \leftarrow mean(x_{adv}), x_{val} \leftarrow expand(x_{val})$
17:	$value \leftarrow x_{adv} + x_{val}$
return value	

III. 제안하는 기법

3.1 추론을 위한 컨볼루션층 병렬화 전체 구조

이기종 시스템, 즉 CPU와 FPGA가 동시에 존재하는 시스템의 경우 각 플랫폼 사이에는 데이터 교환으로 인한 오버헤드가 존재한다[16]. 따라서 병렬 연산으로 인한 가속화 효과와 오버헤드는 Trade-Off 관계가 있다고 볼 수 있다. 또한, 알고리즘에서 앞선 연산의 데이터와 이후의 데이터가 종속 관계일 경우 병렬 처리 또는 파이프라인 설계에 어려움이 있다[17].

그러므로 하드웨어와 소프트웨어의 동시 설계를 통해 가속화 효과를 기대하기 위해서는 알고리즘의 시공간 복잡도와 데이터 의존성을 신중하게 고려하여 구성하여야 한다. 이러한 관계를 고려하지 않은 설계의 경우 데이터 교환에 과도하게 시간이 소모되어 가속화에 실패할 우려가 있다.

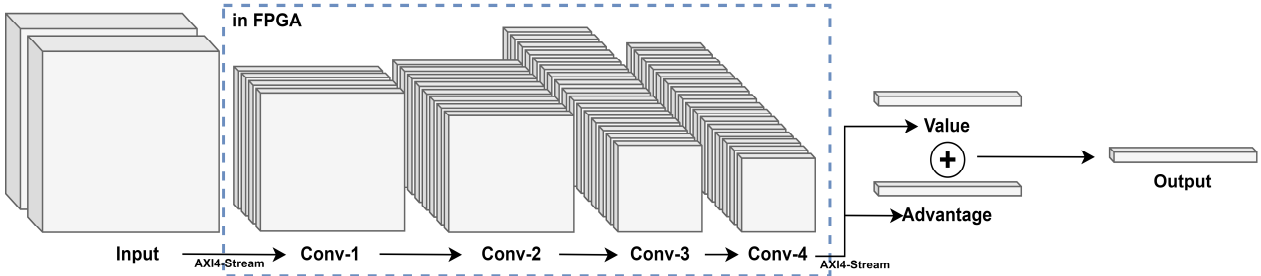


그림 2. Dueling deep Q-network의 도식화
Fig. 2. Schematic of dueling deep Q-network

Dueling deep Q-network는 크게 컨볼루션층[18]과 선형층으로 나뉘는데, 컨볼루션층의 경우 한 커널 연산 안에서 데이터들이 독립적이므로 병렬 연산을 통해 가속 효과를 기대할 수 있다. 다만, 선형층은 모든 신경이 연결된 관계이며 모든 관계에 대해 가중치가 존재한다. 이때 병렬 연산을 통해 기대되는 가속화 효과보다 가중치에 대한 데이터 교환으로 인해 발생하는 오버헤드가 크게 되므로 본 논문에서는 그림 3과 같이 컨볼루션층에 대한 병렬화를 실시하였다. 다시 말해, 제안하는 시스템에서 전체 dueling deep Q-network의 컨볼루션층은 FPGA에서 연산하고, 선형층은 CPU에서 연산한다. 하드웨어 IP의 구현에는 HLS(High-Level Synthesis) 도구[19]가 사용되었다.

3.2 커널 연산 병렬화

그림 4에서 확인할 수 있듯이, 컨볼루션층에서 이루어지는 2차원 합성곱 연산은 커널이 한 칸씩 이동하며 그 위치에 해당하는 입력과 가중치의 곱을 누적하는 형태로 이루어진다. 여기서 표 3의 7과 같이 각각 독립적이고 반복적인 행렬은 표 4에서 제시한 방법대로 병렬 연산하기에 적합하다. 다만, 병렬 연산의 규모가 커질수록 많은 FPGA 자원을 필요로 하며, 이는 상황에 따라 상당히 제한적이므로 이를 고려하여 파이프라인 시작 간격을 신중하게 결정하여야 한다.

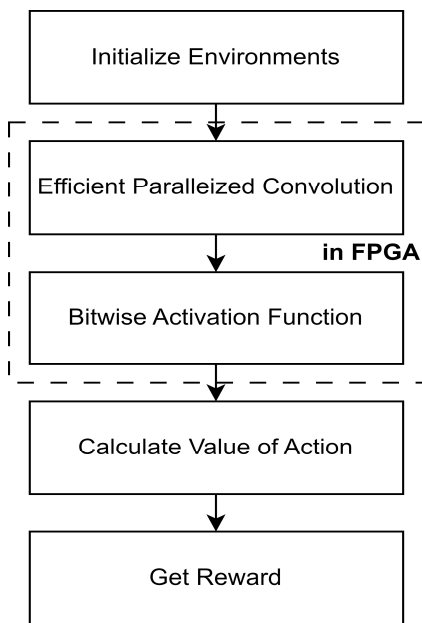
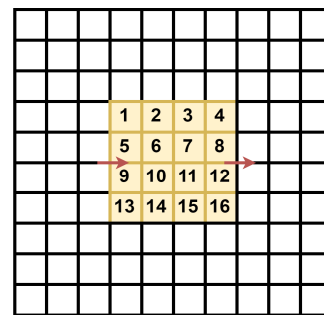


그림 3. 제안된 FPGA기반 컨볼루션층 추론 병렬화 방법
Fig. 3. Proposed FPGA-based parallelization method for convolutional layer inference



— Baseline →

$$1 \times \text{Weight}_1, \dots, 16 \times \text{Weight}_{16}$$

— Proposed →

$$1 \times \text{Weight}_1, \dots, 16 \times \text{Weight}_{16}$$

in Parallel

그림 4. 커널 병렬 연산 예시
Fig. 4. Example of parallel kernel computation

표 3. 리팩토링 전의 컨볼루션층

Table 3. Convolutional layer before refactoring

Algorithm 2: Baseline convolution	
Input:	in, weight, bias
Output:	output
1:	for n : do:
2:	for i : do:
3:	for j : do:
4:	for c : do:
5:	for p : do:
6:	for q : do:
7:	output[n][i][j] += weight[n][c][q][p] × in[c][i+p][j+q]
8:	end
9:	end
10:	output[n][i][j] += bias[n]
11:	end
12:	end
13:	end
14:	end
	return output

표 4. 제안된 HW 기반 컨볼루션층

Table 4. Proposed HW-based convolutional layer

Algorithm 3: Hardware-based convolution	
Input:	in, weight, bias
Output:	output
1:	for n : do:
2:	for i : do:
3:	for j : do:
4:	for c : do:
	local[c] =
5:	$\sum_{k=0}^{S_c} \sum_{t=0}^{S_c} (\text{weight}[n][c][t][k] \times \text{in}[c][i+k][j+t])$
6:	end
7:	output[n][i][j] = $\sum_{k=0}^e \text{local}[k] + \text{bias}[n]$
8:	end
9:	end
10:	end
	return output

3.3 부호비트 검사 기반 ReLU

컨볼루션층에서 입력은 2차원 합성곱 연산 후에 활성화 함수인 ReLU를 통과하게 된다.

$$\text{ReLU}(x) = \begin{cases} x & (x > 0) \\ 0 & (x \leq 0) \end{cases} \quad (1)$$

ReLU는 입력값이 0보다 작으면 0을 반환하고, 0보다 크면 입력값을 반환한다.

한편, 일반적으로 소수점 데이터의 처리에는 64비트 부동 소수점 표현이 사용되나, 하드웨어를 구현하는 상황에서는 제한된 자원을 효율적으로 활용할 필요가 있다. 그러므로 소수점 표현의 정밀도와 연산 속도 및 하드웨어 자원 가용량의 Trade-Off 관계를 고려하여 본 논문에서는 FPGA에서 소수점 데이터를 그림 5와 같이 IEEE-754[20] 기반 16비트 부동 소수점 표현으로 처리하도록 설계되었다. 이때 첫 번째 비트는 부호를 나타내므로 표 5에 나타난 알고리즘과 같이 부호를 나타내는 비트만 검사함으로써 불필요한 FPGA 자원 소모를 방지하고, 하드웨어 차원에서 낮은 지연 시간의 활성화 함수를 구현할 수 있다.

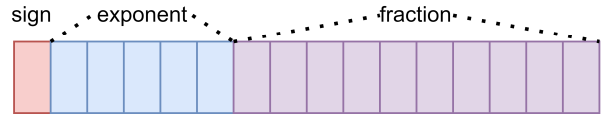


그림 5. IEEE-754 반정밀도 부동 소수점
Fig. 5. IEEE-754 half-precision floating point

표 5. 제안된 HW 기반 활성화 함수

Table 5. Proposed HW-based activation function

Algorithm 4: Hardware-based ReLU	
Input:	x
Output:	x
1:	if(x & 0x8000): do:
2:	x ← 0
	return x

3.4 데이터 의존성 및 메모리 액세스 개선

2차원 합성곱 연산에 있어 공간 복잡도에 가장 큰 영향을 미치는 요소는 가중치 데이터이다. 이때 가중치는 4차원 배열로 구현되는데, 앞의 합성곱 연산의 흐름을 볼 때, 이를 메모리에 한 번에 모두 불러오는 것은 비효율적이라고 볼 수 있다. 이는 파이프라인 설계에 있어 데이터 종속을 일으키며, 수행 시간 단축에 좋지 않은 영향을 미칠 수 있다.

따라서 그림 6에 나타난 것과 같이 가중치 배열을 적절히 나눔으로써 데이터 의존성 및 메모리 액세스를 개선할 수 있다. 또한, 2차원 합성곱 연산은 곱 연산 결과를 계속해서 누적하는데, 이는 읽기 및 쓰기가 빈번하게 일어나는 것으로 귀결된다. 이러한 패턴을 개선하기 위해 표 4의 5, 7과 같이 지역 변수에 곱 연산 결과를 저장하고 후에 이를 모두 더하여 효율적인 파이프라인 설계가 가능하게 하였다.

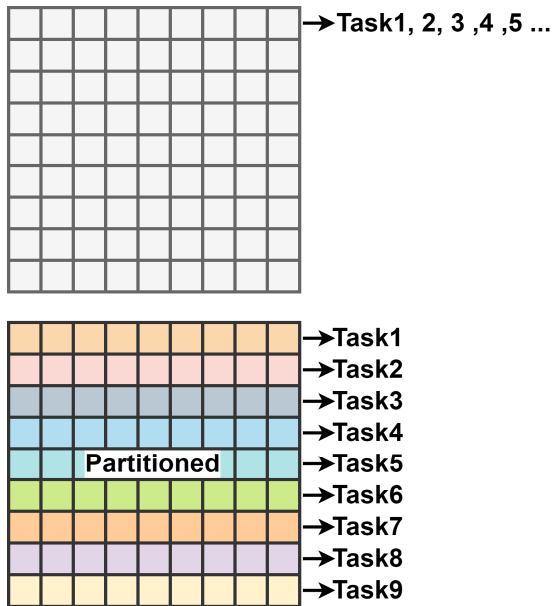


그림 6. 배열 분할 도식화
Fig. 6. Schematic of array partitioning

IV. 결 과

4.1 환경 셋업

강화학습을 위한 환경은 앞에서 밝혔듯이 15×15 그리드 환경에서 이루어진다. 추론에는 사전 훈련된 dueling deep Q-network 모델이 사용되었으며, 현재 상태를 기반으로 에이전트가 가질 수 있는 행동 actions 중 최상의 가치를 가지는 행동을 취한 후 환경으로부터 보상 rewards에 따라 보상을 누적한다. 여기서 양의 보상은 오직 목적지에 도착했을 때만 얻을 수 있도록 보상 함수를 설정하였다.

$$\{up, down, left, right, wait\} \in actions \quad (2)$$

$$rewards = -a_{obstacle} - a_{wait} - a_{normal} + a_{end} \quad (3)$$

제안된 라우팅 알고리즘의 시뮬레이션은 Xilinx사의 UltraScale+ MPSoC ZCU104 Evaluation Kit 보드에서 시행되었다. 해당 보드는 UltraScale+ FPGA와 ARM Cortex-A53 프로세서를 포함한다. CPU와 FPGA 간의 데이터 통신에는 AXI4-Stream 인터페이스[21], Pytorch 딥러닝 프레임워크 및 Python 환경에서의 시뮬레이션을 위해 Petalinux 기반의 PYNQ OS가 사용되었다.

4.2 결과 및 분석

구현된 하드웨어의 자원 사용량은 표 6에 정리되어 있으며, LUT(Look-Up Table)의 경우 사용된 보드의 한도 대비 약 42.7%를, DSP(Digital Signal Processor)는 약 8.3%를 사용하였다. 그림 7은 알고리즘의 추론 환경을 나타내는데, 각 타임 스텝마다 링크 단절 구간은 왼쪽으로 한 칸씩 이동하며, 노란색으로 표시된 시작 지점 [2, 2]에서 초록색으로 표현된 지점 [10, 10]으로 라우팅을 수행해야 한다. 이때 분홍색으로 표시된 링크 단절 구간을 피하면서 목표 지점에 도달해야 한다.

표 6. FPGA 자원 사용량
Table 6. FPGA resource utilization

Resource	Utilization
LUT	196,739
LUTRAM	66,182
FF	110,418
BRAM	5
DSP	144
BUFG	6

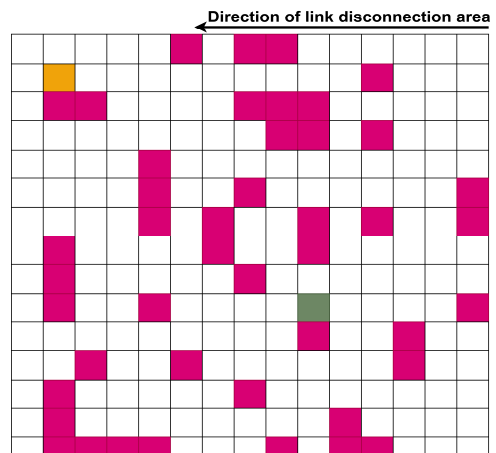


그림 7. 라우팅 환경
Fig. 7. Routing environment

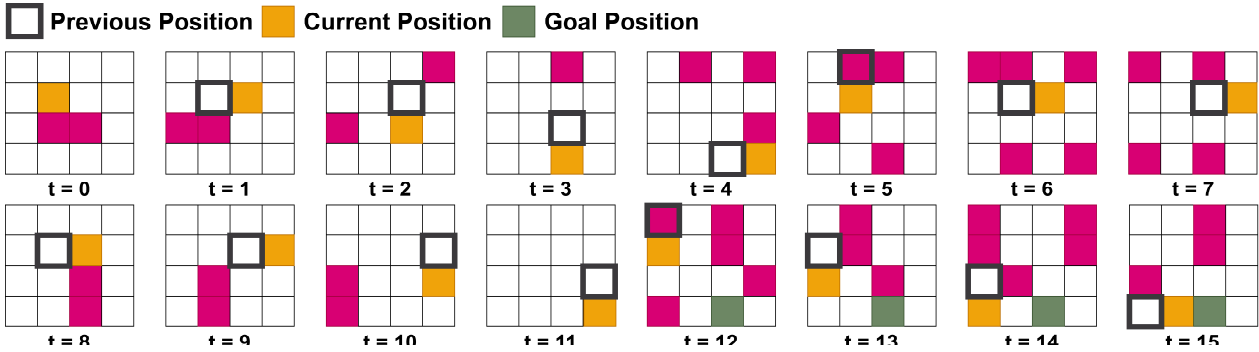


그림 8. 각 타임 스텝에서의 라우팅 과정
Fig. 8. Routing process at each time step

결론적으로, 시작 지점에서 최적의 행동을 반복한 결과 16번의 반복 끝에 목표 지점에 도달하였고, 결과는 각 시스템에서 그림 8의 경로와 같은 결과를 출력했는데, 이는 반 정밀도 부동 소수점 기반 연산의 정밀도가 추론에 영향을 크게 주지 않으면서 연산 속도를 높이고 자원을 절약할 수 있음을 나타낸다. 이때 컨볼루션층의 연산에 대한 수행시간은 그림 9와 같았다.

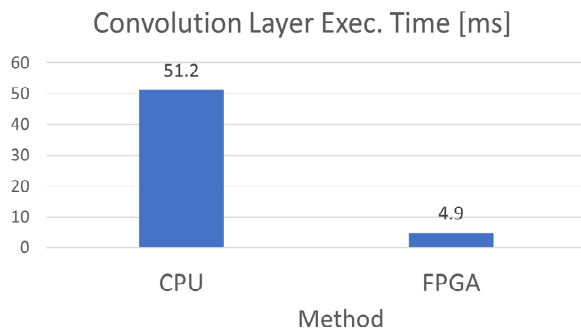


그림 9. 컨볼루션층 연산 수행시간 측정 결과
Fig. 9. Evaluation results of convolutional layer computation time

그림 10은 CPU만 사용했을 때와 제안된 알고리즘을 적용하여 FPGA를 기반으로 컨볼루션층을 병렬화하여 학습했을 때의 추론에 걸린 전체 수행시간을 나타낸다. 결론적으로 제안된 알고리즘의 전체 수행시간이 약 29.0% 감소했음을 확인할 수 있다. 순수 하드웨어에서의 컨볼루션층 연산 수행시간이 약 90.2% 감소했지만, 데이터 교환 과정에서 발생한 입출력 오버헤드가 수반되기 때문에 이러한 결과가 도출되었다.

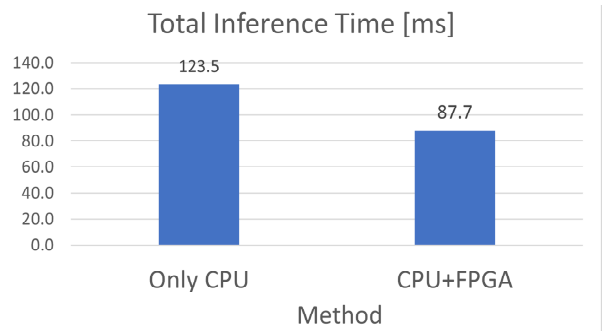


그림 10. 전체 추론 시간 측정 결과
Fig. 10. Evaluation results of total inference time

V. 결론

본 논문에서는 저궤도 위성 네트워크를 위한 강화학습 기반 라우팅 최적화 기법과 실시간성 향상을 위한 병렬화 기반의 추론 기법을 제안하였다. 라우팅 문제를 그리드 환경에서의 MDP로 정의하고, 이를 효율적으로 해결하기 위해 dueling deep Q-network를 적용하였다. 또한 저궤도 위성 네트워크 환경에서의 실시간성을 보장하기 위하여 컨볼루션층을 병렬화하는 하드웨어와 소프트웨어의 상호설계 시스템을 설계하였다. 커널 연산의 병렬화, 부호비트 검사 기반의 ReLU 구현과 데이터 의존성 및 메모리 액세스 개선을 통해 컨볼루션층 연산의 수행시간이 기존 대비 약 90.2% 단축되었으며, 전체 수행시간의 경우 기존의 방식과 같은 라우팅 결과를 약 29.0% 짧은 시간 안에 도출함으로써 기존 방식보다 높은 실시간성을 보장한다. 추후 실시간 라우팅 알고리즘 외의, 저궤도 위성에 탑재될 다른 애플리케이션들과의 통합이 이루어지는 시점에서 전체 시스템에 관한 가속화 연구를 진행하고자 한다.

References

- [1] Y.-E. Lee and K.-I. Kim, "Cross-Point Based Routing Protocol in Low Earth Orbit Communication Networks", Proc. of the Korea Information Processing Society, pp. 72-74, Nov. 2021. <https://doi.org/10.3745/PKIPS.y2021m11a.72>.
- [2] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey", Journal of Artificial Intelligence Research, Vol. 4, pp. 237-285, May 1996. <https://doi.org/10.1613/jair.301>.
- [3] B.-S. Roh, M.-H. Han, D.-W. Kum, and K.-S. Jeon, "A Study on the Reinforcement Learning Routing for LEO Satellite Network", Proc. of the Korean Institute of Communication Sciences Conference, pp. 537-538, 2022.
- [4] L. He, X.-L. Liu, Y.-W. Chen, L.-N. Xing, and K. Liu, "Hierarchical scheduling for real-time agile satellite task scheduling in a dynamic environment", Advances in Space Research, Vol. 63, No. 2, pp. 897-912, Jan. 2019. <https://doi.org/10.1016/j.asr.2018.10.007>.
- [5] X. Wang, Z. Dai, and Z. Xu, "LEO Satellite Network Routing Algorithm Based on Reinforcement Learning", 2021 IEEE 4th International Conference on Electronics Technology (ICET), Chengdu, China, pp. 1105-1109, May 2021. <https://doi.org/10.1109/ICET51757.2021.9451072>.
- [6] P. Zuo, C. Wang, Z. Yao, S. Hou, and H. Jiang, "An Intelligent Routing Algorithm for LEO Satellites Based on Deep Reinforcement Learning", 2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall), Norman, OK, USA, pp. 1-5, Sep. 2021. <https://doi.org/10.1109/VTC2021-Fall52928.2021.9625325>.
- [7] N. Hou, X. Yan, and F. He, "A survey on partitioning models, solution algorithms and algorithm parallelization for hardware/software co-design", Design Automation for Embedded Systems, Vol. 23, pp. 57-77, Apr. 2019. <https://doi.org/10.1007/s10617-019-09220-7>.
- [8] D. Kim, Y. Han, H. Lee, Y. Kim, H.-H. Kwon, C. Kim, and W. Choi, "Accelerated Particle Filter with GPU for Real-Time Ballistic Target Tracking", IEEE Access, Vol. 11, pp. 12139-12149, Jan. 2023. <https://doi.org/10.1109/ACCESS.2023.3238873>.
- [9] S. Lee, H. Lee, Y. Kim, J. Kim, and W. Choi, "GPU-Accelerated PD-IPM for Real-Time Model Predictive Control in Integrated Missile Guidance and Control Systems", Sensors, Vol. 22, No. 12, pp. 1-14, Jun. 2022. <https://doi.org/10.3390/s22124512>.
- [10] J. Wang, W. Tong, and X. Zhi, "Model Parallelism Optimization for CNN FPGA Accelerator", Algorithms, Vol. 16, No. 2, pp. 1-13, Feb. 2023. <https://doi.org/10.3390/a16020110>.
- [11] L. Gong, C. Wang, X. Li, and X. Zhou, "Improving HW/SW Adaptability for Accelerating CNNs on FPGAs Through A Dynamic/Static Co-Reconfiguration Approach", IEEE Transactions on Parallel and Distributed Systems, Vol. 32, No. 7, pp. 1854-1865, Jul. 2021. <https://doi.org/10.1109/TPDS.2020.3046762>.
- [12] M. Qasaimeh, K. Denolf, J. Lo, K. Vissers, J. Zambreno, and P. H. Jones, "Comparing Energy Efficiency of CPU, GPU and FPGA Implementations for Vision Kernels", 2019 IEEE International Conference on Embedded Software and Systems (ICCESS), Las Vegas, NV, USA, pp. 1-8, Jun. 2019. <https://doi.org/10.1109/ICCESS.2019.8782524>.
- [13] P. Yuan, Y. Wang, M. Su, Z. Yang, and Q. Zhang, "Markov decision process-based routing algorithm in hybrid Satellites/UAVs disruption-tolerant sensing networks", IET Communications, Vol. 13, No. 10, pp. 1415-1424, Jun. 2019. <https://doi.org/10.1049/iet-com.2018.5932>.
- [14] M. Sewak, "Deep Q Network (DQN), Double DQN, and Dueling DQN", Deep Reinforcement Learning, pp. 95-108, Jun. 2019. https://doi.org/10.1007/978-981-13-8285-7_8.

[15] H. Yan, Q. Zhang, and Y. Sun, "A Novel Routing Scheme for LEO Satellite Networks Based on Link State Routing", 2014 IEEE 17th International Conference on Computational Science and Engineering, Chengdu, China, pp. 876-880, Dec. 2014. <https://doi.org/10.1109/CSE.2014.178>.

[16] B. Plancher, et al., "Accelerating Robot Dynamics Gradients on a CPU, GPU, and FPGA", IEEE Robotics and Automation Letters, Vol. 6, No. 2, pp. 2335-2342, Apr. 2021. <https://doi.org/10.1109/LRA.2021.3057845>.

[17] Emma and Davidson, "Characterization of Branch and Data Dependencies in Programs for Evaluating Pipeline Performance", IEEE Transactions on Computers, Vol. C-36, No. 7, pp. 859-875, Jul. 1987. <https://doi.org/10.1109/TC.1987.1676981>.

[18] K. O'Shea and R. Nash, "An introduction to convolutional neural networks", arXiv:1511.08458, Nov. 2015. <https://doi.org/10.48550/arXiv.1511.08458>.

[19] S. Lahti, P. Sjövall, J. Vanne, and T. D. Hämmäläinen, "Are We There Yet? A Study on the State of High-Level Synthesis", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 38, No. 5, pp. 898-911, May 2019. <https://doi.org/10.1109/TCAD.2018.2834439>.

[20] IEEE, "IEEE Standard for Floating-Point Arithmetic", IEEE Std 754-2019 (Revision of IEEE 754-2008), pp. 1-84, Jul. 2019. <https://doi.org/10.1109/IEEESTD.2019.8766229>.

[21] AMBA and ARM, "Axi4-stream protocol specification", Vol. IHI B 51, pp. 1-56, Apr. 2021.

저자소개

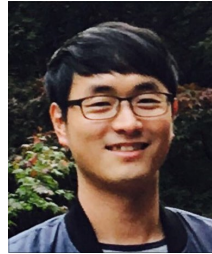
김 대 연 (Daeyeon Kim)



2023년 2월 : 금오공과대학교
전자공학부(공학사)
2023년 3월 ~ 현재 :
금오공과대학교 전자공학부
IT융복합공학과 석사과정
관심분야 : Deep learning,

real-time embedded systems, and algorithm
acceleration with GPU and FPGA

이 현 철 (Heoncheol Lee)



2006년 8월 : 경북대학교
전자전기컴퓨터학부(공학사)
2008년 8월 : 서울대학교
전기컴퓨터공학과(공학석사)
2013년 8월 : 서울대학교
전기컴퓨터공학과(공학박사)
2013년 9월 ~ 2019년 2월 :

국방과학연구소 선임연구원
2019년 3월 ~ 현재 : 금오공과대학교 전자공학부
IT융복합공학과 조교수
관심분야 : SLAM, 자율주행, 인공지능, 알고리즘 가속화

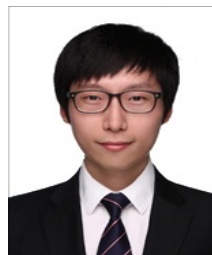
원 동 식 (Dongshik Won)



2011년 12월 : 미네소타대학교
전기공학부(전기공학사)
2014년 2월 : 한국과학기술원
전기및전자공학부(공학석사)
2014년 2월 ~ 2021년 1월 :
국방과학연구소 선임연구원
2021년 1월 ~ 2023년 3월 :

글래스도코리아 기술이사
2019년 2월 ~ 현재 : 한국과학기술원 박사과정
2023년 4월 ~ 현재 : 텔레픽스 주식회사 이사
관심분야 : 위성 미션 해석 및 설계, 전자광학 시스템,
추적및항법, 우주미션시스템을 위한 인공지능/기계학습

한 명 훈 (Myoung-Hun Han)



2007년 2월 : 중앙대학교
컴퓨터공학(공학사)
2009년 8월 : 중앙대학교
컴퓨터공학(공학석사)
2021년 8월 : 중앙대학교
컴퓨터공학(공학박사)
2014년 10월 ~ 현재 :

국방과학연구소 선임연구원
관심분야 : 네트워크, M&S, LEO, 인공지능