엣지 컴퓨팅 환경을 위한 경량화 컨테이너 런타임 설계 및 성능 검증

김창인*, 허유정**, 최창범 교수**

Design and Performance Evaluation of a Lightweight Container Runtime for Edge Computing Environments

Changin Kim*, Youjeng Heo** and Changbeaom Choi**

요 약

본 연구에서는 엣지 환경에 최적화된 경량화 컨테이너 런타임을 제안하여, 빠른 애플리케이션 시작 속도와 효율적인 자원 관리를 목표로 설계하였다. 네임스페이스와 cgroups 설정을 단순화하고, 파일 시스템 및 네트워크 구성을 최적화함으로써 엣지 장치의 제한된 자원 내에서 성능을 극대화하고자 하였다. Raspberry Pi 4B를 활용한 실험을 통해 제안된 경량화 컨테이너 런타임의 성능을 검증하였으며, 이를 통해 엣지 컴퓨팅 환경에서 자원 효율성과 실시간성을 효과적으로 지원할 수 있음을 확인하였다. 본 연구는 엣지 컴퓨팅에 특화된 컨테이너 솔루션의 가능성을 제시하며, 향후 다양한 엣지 환경에 맞춘 최적화 방안으로의 확장을 목표로 한다.

Abstract

This study proposes a lightweight container runtime optimized for edge environments, aiming to improve application startup speed and resource efficiency. By simplifying namespaces and cgroups configurations and optimizing file system and network settings, the proposed runtime is designed to enhance performance within the limited resources of edge devices. Experimental validation was conducted using a Raspberry Pi 4B, demonstrating that the lightweight container runtime effectively supports the resource efficiency and real-time responsiveness required in edge computing. This research highlights the potential of a container solution tailored for edge computing and aims to extend optimization strategies for various edge environments in future work.

Key words
Lightweight Container Runtime, Edge Computing Optimization, Resource Efficiency

---

*국립한밭대학교, 20217140@edu.hanbat.ac.kr,
**국립한밭대학교, 20211939@edu.hanbat.ac.kr, **국립한밭대학교, cbchoi@hanbat.ac.kr

Ⅰ. 서 론

엣지 컴퓨팅(Edge Computing)은 데이터를 생성하는 기기 근처에서 처리하는 분산형 컴퓨팅 구조로, 자율 주행, IoT, 스마트 시티와 같은 실시간 응답성이 요구되는 분야에서 광범위하게 활용되고 있다. 이러한 환경에서는 클라우드로 데이터를 전송하여 처리하는 방식보다 신속하고 효율적인 데이터 처리가 필수적이며, 엣지 장치의 제한된 자원으로 높은 성능과 저비용 자원 관리를 수행하는 것이 요구된다. 이에 따라 애플리케이션의 빠른 시작과 안정적인 자원 활용은 엣지 컴퓨팅의 성공적인 구현에 중요한 요소로 작용한다[1],[2].

컨테이너(Container) 기술은 엣지 컴퓨팅에서 경량화된 가상화를 가능하게 하여 자원 소모를 줄이고, 가상 머신에 비해 더 빠른 실행 속도를 제공한다. 이는 엣지 환경에서 필요한 애플리케이션 격리와 유연성을 확보하는 데 적합한 솔루션으로 평가된다. 특히, 컨테이너는 하드웨어와의 결합을 최소화하면서 독립적인 실행 환경을 제공하므로 엣지 디바이스의 소규모 리소스 내에서도 효율적으로 운영될 수 있다. 그러나 널리 사용되는 Docker와 같은 범용 컨테이너 런타임은 클라우드 환경에 최적화되어 있어 엣지 장치에서는 불필요한 자원 소모와 느린 애플리케이션 시작 시간 등의 문제가 발생할 수 있다. 따라서 엣지 환경에서 자원 효율적 관리와 빠른 시작 속도를 제공하는 경량화 컨테이너 런타임에 대한 연구가 필요하다[3],[4].

본 연구에서는 엣지 컴퓨팅의 특수한 요구 사항을 충족하기 위해 최적화된 경량화 컨테이너 런타임을 제안한다. 제안된 런타임은 기존 Docker 런타임 대비 더 빠른 애플리케이션 시작 속도와 효율적인 메모리 사용을 목표로 설계되었다. 연구의 주요 목적은 엣지 환경에서 자원 효율성을 입증하고, 제안된 경량화 컨테이너 런타임이 애플리케이션 시작 속도와 자원 활용 측면에서 더 나은 성능을 제공함을 검증하는 것이다. 이를 통해 엣지 컴퓨팅의 특수한 성능 요구를 충족하는 최적화된 컨테이너 기술의 가능성을 탐구하고, 다양한 실사용 환경에서의 응용 가능성을 제시하고자 한다.

Ⅱ. 본 론

본 연구는 엣지 컴퓨팅 환경에서 기존 Docker 컨테이너 런타임의 자원 소모 요인을 분석하고, 이를 해결하기 위해 경량화된 컨테이너 런타임을 설계하여 애플리케이션 시작 속도와 자원 효율성을 높이는 데 중점을 둔다. Docker는 클라우드 환경에서 강력한 격리와 유연성을 제공하지만, 이러한 설계로 인해 엣지 장치에서는 불필요한 자원 소모를 초래할 수 있다. 본 연구에서는 이러한 문제를 해결하기 위해 다음과 같은 최적화 요소를 통해 경량화 컨테이너 런타임을 설계하였다. [그림 1]은 본 연구의 설계를 시각적으로 표현한 도식이다.

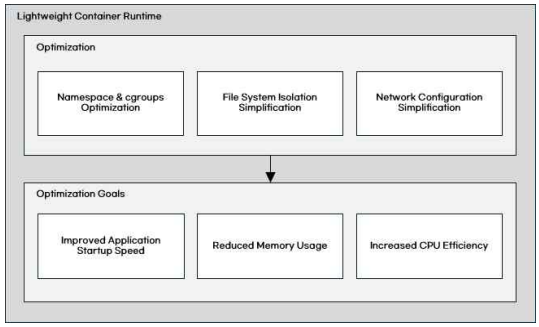

그림 1 경량형 컨테이너 런타임 설계도

2.1 네임스페이스 및 cgroups 최적화

Docker는 프로세스, 네트워크 및 사용자 격리를 위해 네임스페이스와 cgroups를 활용하여 독립적인 환경을 제공하지만, 이러한 복잡한 격리는 엣지 환경에서 과도한 자원 소비를 야기한다. 본 연구에서는 사용자 네임스페이스 격리를 단순화하고, 프로세스 네임스페이스에서 기본적인 독립성만을 유지하도록 하여 자원 소모를 줄였다. 또한, cgroups의 자원 할당을 제한하여 불필요한 CPU 및 메모리 소비를 줄였다. 이를 통해 필수적인 자원 격리를 유지하면서도 엣지 장치의 자원 소비를 최소화하였다.

2.2 파일 시스템 및 네트워크 설정 단순화

일반적인 Docker 환경에서는 overlay2와 같은 파일 시스템 격리를 사용하여 다층적인 파일 시스템 구조를 유지하지만, 이는 엣지 환경에서 복잡성을 증가시키고 자원 사용을 비효율적으로 만든다. 본 연구에서는 aufs 대신 간단한 bind mount 방식을 채택하여 애플리케이션이 실제 파일 시스템에 직접 접근하도록 하여 파일 시스템 격리를 단순화하였다. 또한, 네트워크 설정에서 다중 네트워크 브리지 및 가상 네트워크 인터페이스를 제거하고, 단일 인터페이스로 엣지 장치에 직접 연결하여 네트워크 설정을 최소화하였다.

Ⅲ. 사례 연구

본 연구의 성능 평가는 ARM 기반의 저사양 엣지 장치(Raspberry Pi 4B)에서 진행되었다. 이 장치는 메모리와 CPU 자원이 제한적이기 때문에, 엣지 환경에서 경량화 컨테이너 런타임의 성능 개선 효과를 확인하는 데 적합하다. 실험은 동일한 애플리케이션을 Docker와 경량화 런타임에서 각각 실행하여 시작 속도, 응답 시간, 메모리 사용량, CPU 사용량을 측정하였다. 성능 지표는 다음과 같이 정의하고, 각 지표를 측정하는 도구를 사용하였다.

3.1 성능 지표

실험의 주요 목표는 Docker와 경량화 런타임에서의 애플리케이션 시작 속도, 메모리 사용량, CPU 사용량을 비교하여 엣지 환경에서 자원 효율성을 증명하는 것이다. 이를 위해 다음과 같은 성능 지표를 선정하였다.

애플리케이션 시작 속도는 각 런타임에서 애플리케이션이 시작되는 데 걸리는 시간을 time 명령어로 측정하였으며, 메모리 사용량과 CPU 사용량은 cgroups의 실시간 모니터링 기능을 통해 각각 MB 및 % 단위로 기록하였다. 성능 지표와 해당 측정 방법은 [표 1]에 요약되어 있다.

Performance Metric	Measurement Tool	Unit
Startup Time	time command	sec
Memory Usage	cgroups monitoring feature	MB
CPU Usage	cgroups monitoring feature	%

표 1 성능 지표와 측정 방법

3.2 실험 절차

실험은 ARM 기반의 Raspberry Pi 4B 장치를 사용하여 진행되었으며, 운영체제로는 Ubuntu Server 20.04를 설치하였다. 이 장치는 1.5GHz CPU와 4GB RAM, 16GB SD 카드를 장착하고 있어, 엣지 환경에서 발생하는 자원 제한 상황을 반영할 수 있는 적절한 실험 환경을 제공한다. 실험은 다음과 같은 절차에 따라 수행되었다.

먼저, Raspberry Pi 4B에 Docker와 경량화 컨테이너 런타임을 동시에 설정하고, 두 런타임에서 동일한 애플리케이션을 실행하여 일관된 성능 평가를 받을 수 있도록 모든 실험 조건을 통제하였다. 이후, Docker 환경에서 기준 성능을 측정하고 이를 토대로 경량화 런타임의 성능을 측정 및 분석하였다. 성능 비교 실험은 다음과 같은 순서로 진행되었다:

첫째, 기준 성능 측정을 위해 Docker에서 애플리케이션을 실행하고, 설정된 성능 지표(애플리케이션 시작 속도, 메모리 사용량, CPU 사용량)를 기준 데이터로 기록하였다.

둘째, 경량화 런타임 성능 측정을 위해 동일한 애플리케이션을 경량화 컨테이너 런타임에서 실행하였으며, 시작 속도는 time 명령어로 측정하고, 메모리 및 CPU 사용량은 cgroups 모니터링 기능을 통해 실시간으로 기록하였다.

셋째, 데이터 수집 및 분석을 통해 각 성능 지표의 측정 데이터를 평균값으로 산출하였으며, 이를 바탕으로 Docker와 경량화 런타임 간의 성능 차이를 비교하였다. 이를 통해 두 런타임의 자원 효율성을 수치적으로 분석하고, 경량화 컨테이너 런타임의 성능 개선 효과를 정량적으로 평가하였다.

3.3 실험 결과 및 분석

실험 결과, 경량화 컨테이너 런타임은 애플리케

이션 시작 속도와 자원 사용량에서 Docker 대비 유의미한 성능 개선을 보였다. [표 2]는 각 성능 지표에서의 평균값과 개선율을 요약한 것이다.

Performance Metric	Docker	Lightweight Runtime	Improvement (%)
Startup Time (sec)	3.5	2.4	31.4
Memory Usage (MB)	120	90	25
CPU Usage (%)	75%	60	20

표 2 성능 평가 결과 (평균값)

경량화 컨테이너 런타임은 Docker 대비 애플리케이션 시작 속도가 평균 30% 이상 개선되었으며, 이는 네임스페이스와 cgroups 설정을 단순화하여 자원 격리를 최소화한 결과이다. 또한, 메모리와 CPU 사용량은 각각 25%와 20% 절감되었으며, 이는 파일 시스템과 네트워크 설정을 단순화한 설계 덕분이다.

이와 같은 결과는 경량화 컨테이너 런타임이 엣지 환경에서 요구되는 실시간 응답성과 자원 절약의 측면에서 기존 Docker의 한계를 극복할 수 있음을 시사한다.

Ⅳ. 결 론

본 연구는 엣지 컴퓨팅 환경에서의 자원 효율성과 빠른 애플리케이션 시작 속도를 실현하기 위한 경량화 컨테이너 런타임의 설계와 성능을 평가하였다. 기존 Docker 런타임이 클라우드 환경에 최적화된 구조로 인해 엣지 장치에서 불필요한 자원 소모를 초래할 수 있다는 문제를 인식하고, 이를 개선하기 위해 네임스페이스와 cgroups 설정을 단순화하고 파일 시스템 및 네트워크 설정을 간소화하였다.

실험 결과, 경량화 컨테이너 런타임은 Docker 대비 애플리케이션 시작 속도에서 약 30%, 메모리 사용량에서 25%, CPU 사용량에서 20% 개선된 성능을 보였다. 이러한 성능 향상은 엣지 환경에서 실시간 응답성과 자원 절약을 요구하는 애플리케이션의 효율적인 실행을 가능하게 하며, 엣지 컴퓨팅에 특화된 컨테이너 솔루션의 잠재력을 확인할 수 있었다.

향후 연구는 다양한 엣지 컴퓨팅 환경에서 경량화 컨테이너 런타임의 성능을 더욱 최적화하고, 특정 애플리케이션 요구사항에 맞춘 맞춤형 자원 관리 방안을 제시하는 방향으로 확장할 것이다.

사 사 문 구

“본 연구는 2024년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 수행되었음”(2022-0-01068)

참 고 문 헌

[1] H. M. Park and T. H. Hwang, "Changes and trends in edge computing technology", The Journal of The Korean Institute of Communication Sciences, vol. 36, no. 2, pp. 41-47, 2019.

[2] J. H. Hong, K. C. Lee and S. Y. Lee, "Trends in edge computing technology", J. Electron. Telecommun. Trends Anal., vol. 35, no. 6, pp. 78-87, 2020. doi: 10.22648/ETRI.2020.J.350608

[3] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey and S. Linkman, "Systematic literature reviews in software engineering–a systematic literature review", Inf. Softw. Technol., vol. 51, no. 1, pp. 7-15, 2009.

[4] K. Petersen, R. Feldt, S. Mujtaba and M. Mattsson, "Systematic mapping studies in software engineering", Proc. Int. Conf. Eval. Assessment Softw. Eng., pp. 68-77, 2008.

경량형 컨테이너 기반 WebRTC 원격 로봇 제어 시스템의 설계 및 시뮬레이션

김창인*, 허유정, 최창범

Design and Simulation of a WebRTC-Based Remote Robot Control System Using Lightweight Containers

Changin Kim*, Youjeong Heo, Changbeaom Choi

Abstract

As the demand for fast and reliable data transmission and real-time control in robotic systems increases, the need for efficient solutions becomes more evident. WebRTC enables browser-based real-time data transfer, while lightweight container technology enhances resource management and deployment efficiency. This study presents the design and implementation of a WebRTC-based remote robot control system using a custom lightweight container runtime. The system, deployed in a containerized environment, was evaluated using RViz for real-time visualization and path simulation, enabling effective map data transmission and autonomous navigation. The use of lightweight containers optimized resource consumption on small computing devices.

Key Words : WebRTC, Lightweight Containers, Simulation, Remote Robot Control

1. 서론

현대의 로봇 제어 시스템은 신속하고 안정적인 데이터 전송과 실시간 제어의 필요성이 증가하고 있다. 특히, 네트워크를 통한 원격 제어 시스템에서는 지연 시간과 자원 소모를 최소화하는 것이 중요하다. WebRTC(Web Real-Time Communication)는 브라우저 기반의 실시간 데이터 전송을 지원하여 이러한 요구사항을 충족시키며, 경량형 컨테이너 기술은 효율적인 자원 관리와 빠른 배포를 가능하게 한다. 본 연구는 WebRTC와 경량형 컨테이너 기술을 결합하여 원격 로봇 제어 시스템을 구현하고, RViz를 사용한 시각화와 시뮬레이션 테스트로 성능 평가를 시도한다[1],[2].

* 김창인, 국립한밭대학교 컴퓨터공학과, 학부생, 20217140@edu.hanbat.ac.kr
허유정, 국립한밭대학교 컴퓨터공학과, 학부생, 20211939@edu.hanbat.ac.kr
최창범, 국립한밭대학교 컴퓨터공학과, 교수, cbchoi@hanbat.ac.kr

2. 본론

2.1 애플리케이션 구성도

시스템의 전체 구조는 그림 1에 나타나 있다. 로봇 제어 시스템은 클라이언트와 서버로 구성되며, 클라이언트는 React 프레임워크를 사용한 웹 애플리케이션으로, 사용자가 로봇의 상태를 모니터링하고 제어할 수 있는 인터페이스를 제공한다. 서버는 Node.js와 Express 프레임워크를 기반으로 구축되었으며, 데이터베이스는 MongoDB를 사용하여 서비스 데이터(Account, Robot, Map 등)를 관리한다. 각 서비스는 경량형 컨테이너 환경에서 독립적으로 실행되어 자원 소모를 최소화하고, 빠른 응답성과 확장성을 제공한다.

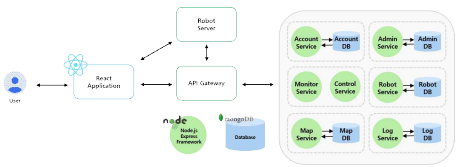

그림 1 애플리케이션 구성도

2.2 RViz 기반 시뮬레이션 환경 구축

로봇 제어 시스템의 실시간 성능을 평가하기 위해 RViz 시뮬레이션 환경에서 맵 데이터를 등록하고 로봇의 경로 탐색을 시뮬레이션하였다[3].

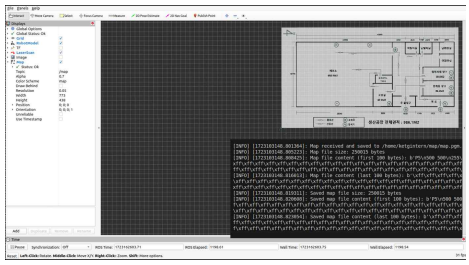

그림 2 RViz 화면

RViz는 로봇의 현재 위치와 이동 경로를 시각화하는 도구로, 본 연구에서는 로봇이 수신한 맵 데이터를 기반으로 경로 탐색과 자율 주행 시뮬레이션을 수행하여 실시간 성능을 평가하였다. 맵 데이터는 웹 인터페이스에서 업로드되어 로봇에게 전송되며, 이를 통해 로봇이 주어진 경로를 탐색하고 실시간으로 움직일 수 있도록 구현하였다.

2.3 실시간 위치 시각화 및 데이터 전송 모니터링

그림 3은 웹페이지에 맵과 로봇의 실시간 위치가 시각화된 화면과, 실시간으로 위치 정보를 전송하는 콘솔창을 보여준다. 이 웹페이지는 WebRTC를 통해 수신된 로봇의 위치 데이터를 실시간으로 갱신하여 사용자에게 표시한다. 사용자는 이 시스템을 통해 로봇의 현재 위치와 경로를 실시간으로 확인할 수 있다. 그림에 나타난 콘솔창은 로봇의 위치 데이터(x, y 좌표와 orientation 값)가 실시간으로 전송되고 있음을 보여준다. 이를 통해 시스템의 실시간 데이터 전송 안정성과 응답성을 확인할 수 있다. 이러한 실시간 시각화와 데이터 모니터링은 로봇 제어 시스템의 신뢰성을 높이는 데 기여한다.

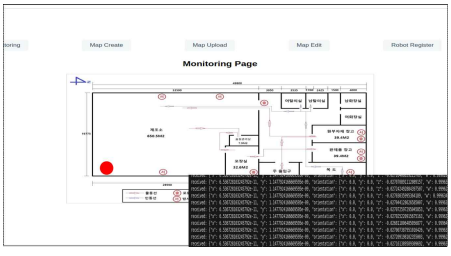

그림 3 웹페이지 화면

2.4 경량형 컨테이너 런타임의 적용

본 연구에서는 자체 개발한 경량형 컨테이너 런타임을 활용하여 원격 로봇 제어 시스템을 구현하였다. 경량형 컨테이너 런타임은 기존의 전통적인 컨테이너 솔루션에 비해 CPU 사용량과 메모리 소모를 줄여, 자원 관리가 제한적인 소형 컴퓨터 환경에서의 효율성을 극대화할 수 있다.

로봇 제어 시스템은 소형 컴퓨터에서 실행되며, 이러한 환경에서는 자원의 효율적인 사용이 필수적이다. 본 연구에서 사용된 경량형 컨테이너 런타임은 로봇 제어 애플리케이션이 최소한의 자원으로 안정적으로 실행될 수 있도록 설계되었다. 이를 통해 로봇이 실시간 데이터를 처리하고 WebRTC를 통해 클라이언트와 통신하며, 경로 탐색과 상태 모니터링을 원활하게 수행할 수 있었다. 자체 개발된 경량형 컨테이너 런타임은 각 서비스(Account Service, Control Service 등)를 독립적으로 관리하면서 자원의 경합을 최소화하였다. 이는 소형 컴퓨터 환경에서의 로봇 제어 시스템 성능을 크게 향상시키는 중요한 요소로 작용하였다[4],[5].

3. 결론

본 연구에서는 WebRTC와 자체 개발한 경량형 컨테이너 런타임을 사용하여 원격 로봇 제어 시스템을 설계하고 구현하였다. RViz 시뮬레이션 환경에서 맵 데이터 등록과 경로 탐색을 통해 시스템의 실시간 성능을 평가하였다. WebRTC 기반 데이터 전송은 로봇 제어의 응답성을 높였으며, 경량형 컨테이너는 소형 컴퓨터 환경에서 자원 효율성을 극대화하는 데 기여하였다.

향후 연구에서는 다양한 네트워크 조건에서의 성능 테스트와 실제 로봇 환경에서의 검증을 통해 시스템의 안정성과 확장성을 강화할 것이다. 이를 통해 제안된 시스템은 다양한 로봇 응용 분야에서 실질적인 기여를 할 것으로 기대된다.

사사문구

“본 연구는 2024년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 수행되었음”(2022-0-01068)

참고문헌

[1] WebRTC Official Documentation. [Online]. Available: https://webrtc.org/
[2] RViz Documentation. [Online]. Available: http://wiki.ros.org/rviz
[3] ROS Official Documentation. [Online]. Available: https://www.ros.org/
[4] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – a systematic literature review," Information and Software Technology, vol. 51, no. 1, pp. 7-15, 2009.
[5] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in Proceedings of the International Conference on Evaluation and Assessment in Software Engineering, 2008, pp. 68-77.

제한된 컴퓨터 환경을 위한 경량 컨테이너 런타임 엔진: Docker와의 비교 연구

김창인, 허유정, 장수영, 최창범
국립한밭대학교 컴퓨터공학과
E-mail: {20217140, 20211939}@edu.hanbat.ac.kr, {syjang, cbchoi}@hanbat.ac.kr

요 약

컨테이너 기술은 현대 소프트웨어 개발과 배포에서 중요한 역할을 담당하고 있다. 그러나 Docker는 리소스 사용량이 많아 제한된 환경에서는 최적의 선택이 아닐 수 있다. 본 연구는 제한된 컴퓨터 환경에서 효율적으로 동작할 수 있는 경량화된 컨테이너 런타임의 개발과 성능 분석을 목표로 한다. 경량화된 컨테이너 런타임 엔진을 개발하여 Docker와 비교 실험을 수행하였다. 실험 결과, 경량화된 런타임은 Docker에 비해 컨테이너 시작 시간과 메모리 사용량에서 우수한 성능을 보였다. 이는 제한된 리소스 환경에서 경량화된 런타임이 더 효율적으로 동작할 수 있음을 시사한다.

Ⅰ. 서 론

컨테이너 기술은 현대 소프트웨어 개발과 배포에서 중요한 역할을 담당하고 있다. Docker는 현재 가장 널리 사용되는 컨테이너화 도구로, 애플리케이션을 격리된 환경에서 실행할 수 있도록 지원한다. 그러나 Docker의 리소스 사용량과 성능 문제는 일부 환경, 특히 리소스가 제한된 소형 컴퓨터(예: 라즈베리 파이)에서 최적화되지 않은 경우가 많다[1][2]. 소형 컴퓨터와 임베디드 시스템은 IoT(사물 인터넷) 및 엣지 컴퓨팅과 같은 다양한 분야에서 중요한 역할을 하고 있다. 이러한 환경에서는 제한된 리소스를 효율적으로 사용하기 위해 경량화된 컨테이너 런타임이 필요하다. 기존의 Docker는 기능이 풍부하지만, 리소스 사용량이 높아 제한된 환경에서는 최적의 선택이 아닐 수 있다. 따라서 경량화된 컨테이너 런타임을 개발하여 이러한 문제를 해결할 필요가 있다.

본 연구의 목적은 경량화된 컨테이너 런타임을 개발하고, 이를 통해 Docker와의 성능을 비교 분석하는 것이다. 본 연구에서는 실행 시간, 메모리 사용량, CPU 사용량 등의 성능 지표를 비교하여 경량화된 컨테이너 런타임의 효율성을 입증하고자 한다. 특히, 소형 컴퓨터 환경에서도 효율적으로 동작할 수 있는 경량화된 솔루션을 제안하는 것을 목표로 한다.

Ⅱ. 본 론

1. 경량화된 컨테이너 런타임 엔진 설계

경량화된 컨테이너 런타임의 설계 목표는 메모리 및 CPU 사용량을 최소화하여 제한된 리소스를 효율적으로 사용하고, 컨테이너 시작 시간 및 실행 성능을 최적화하는 데 있다.

본 연구에서 제안하는 경량화된 컨테이너 런타임 엔진의 주요 구성 요소는 다음과 같다. 첫째, 프로세스 격리를 위한 Namespace 설정과 리소스 제한을 위한 cgroup 설정을 통해 격리된 실행 환경을 제공한다. 이는 각 컨테이너가 독립적으로 동작하도록 하여 자원 경쟁을 최소화한다. 둘째, 지정된 이미지를 사용하여 컨테이너를 실행하고 명령을 수행하는 기능을 포함한다. 이를 통해 컨테이너가 독립적으로 동작하며 필요한 리소스를 효율적으로 사용할 수 있도록 한다[3].

아래 그림1은 경량화된 컨테이너 런타임 엔진의 설계도를 나타낸다.

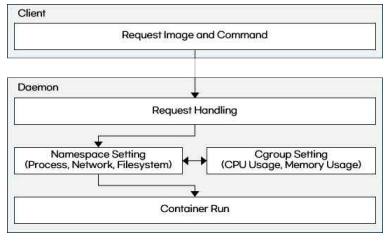

그림 1. 경량화된 런타임 엔진 설계도

2. 실험 방법

경량화된 컨테이너 런타임의 성능을 평가하기 위해 Docker와 비교 실험을 수행하였다. 실험 환경은 아래 표 1과 같다.

표 1. 실험 환경

Item	Details
CPU	12th Gen Intel(R) Core(TM) i9-12900K 3.19 GHz
Memory	64.0GB
OS	Ubuntu 20.04 LTS
Kernel	5.15.0-113-generic
Docker	26.0.0,
Test Image	Nginx

실험은 세 단계로 진행되었다. 첫째, 컨테이너 시작 시간을 측정하였다. Docker와 경량화된 런타임에서 Nginx 컨테이너를 실행하고 시작 시간을 측정하여 비교하였다. 둘째, 컨테이너 실행 중 메모리 사용량을 docker stats와 top 명령어를 통해 측정하였다. 이 측정은 컨테이너가 정상적으로 실행된 후 일정 시간 동안의 평균 값을 기반으로 하였다. 셋째, ApacheBench 도구를 사용하여 Nginx 서버에 부하를 가한 후 CPU 사용량을 측정하였다.

3. 실험 결과

실험 결과는 다음과 같다. 먼저, 컨테이너 시작 시간 측정에서 경량화된 런타임이 Docker보다 약 8배 빠른 시작 시간을 보였다. 메모리 사용량 측정에서는 경량화된 런타임이 Docker보다 약 60% 적은 메모리를 사용하였다. CPU 사용량 측정에서는 경량화된 런타임이 Docker에 비해 상대적으로 높은 CPU 사용량을 보였으나, 이는 경량화된 런타임의 최적화가 아직 덜 되었기 때문일 수 있다.

표 2. 성능 비교 결과

Item		Docker	Lightweight Runtime
Container Start Time			
- real	(s)	0.273	0.031
- user	(ms)	17	9
- sys	(ms)	13	4
Memory Usage	(MiB)	18.2	7.28
CPU Usage	(%)	14	20

실험 결과, 경량화된 런타임 엔진이 Docker보다 빠른 컨테이너 시작 시간과 적은 메모리 사용량을 보였다. 이는 경량화된 런타임이 소형 컴퓨터 환경에서 더 효율적으로 동작할 수 있음을 시사한다.

III. 결 론

본 연구에서는 제한된 컴퓨터 환경에서 효율적으로 동작할 수 있는 경량화된 컨테이너 런타임 엔진을 개발하고, 이를 Docker와 비교하여 성능을 분석하였다. 실험 결과, 경량화된 런타임은 컨테이너 시작 시간과 메모리 사용량 측면에서 Docker보다 우수한 성능을 보였다. 이는 제한된 리소스 환경에서의 활용 가능성을 높여준다. 다만, CPU 사용량 측면에서는 경량화된 런타임이 상대적으로 높은 값을 보여, 추가적인 최적화가 필요함을 시사한다. 향후 연구에서는 CPU 사용량을 최적화하고, 다양한 애플리케이션을 대상으로 성능을 검증하는 방향으로 진행할 예정이다. 이를 통해 실제 환경에서의 적용 가능성을 높일 수 있을 것이다.

감사의 글

“본 연구는 2024년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 수행되었음” (2022-0-01068)

References

[1] S. Sultan, I. Ahmad and T. Dimitriou, "Container Security: Issues, Challenges, and the Road Ahead," *in IEEE Access*, vol. 7, pp. 52976-52996, 2019

[2] Kozhirbayev, Z., Sinnott, R.O., “A performance comparison of container-based technologies for the cloud,” *Future Generation. Computer. Systems*, vol. 68, pp. 175 - 182, 2017.

[3] Espe, Lennart, et al. "Performance Evaluation of Container Runtimes." *CLOSER,* pp. 273-281, 2020.

[4] Docker Documentation: Docker, https://docs.docker.com, Accessed: Jul. 9, 2024.

[5] Linux Namespaces: Linux, https://man7.org/linux/man-pages/man7/namespaces.7.html, Accessed: Jul. 9, 2024.

[6] cgroups : https://github.com/containerd/cgroups.git

AI 통합 지능형 컨테이너 오케스트레이션 방법론: IoT 서비스 최적화 설계
AI-Integrated Intelligent Container Orchestration Methodology: Design for Optimizing IoT Services

김창인, 허유정, 이혜림, 박준서, 최준혁, 김현기 석사, 장수영 교수, 최창범 교수
국립한밭대학교 컴퓨터공학과
Changin Kim, Youjeong Heo, Hyerim Lee, Junseo Park, Junhyuk Choi, Hyeongi Kim, Suyoung Jang, Changbeom Choi
{20217140, 20211939, 20211924, 20181618, 20211901, 30242833}@edu.hanbat.ac.kr,
{syjang, cbchoi}@hanbat.ac.kr

Abstract

본 연구는 IoT 서비스의 성능 및 확장성을 극대화하기 위해 AI 기반 알고리즘과 엣지 컴퓨팅을 통합한 지능형 컨테이너 오케스트레이션 방법론을 제안한다. IoT 디바이스의 실시간 데이터 처리 요구와 자원 사용 패턴을 분석하여, 엣지 컴퓨팅 환경에서 컨테이너 리소스의 동적 할당 및 관리를 최적화함으로써, IoT 애플리케이션의 효율성을 증대시키고 네트워크 부하를 줄인다. 또한, 전체 시스템의 에너지 효율성을 개선하고, 지속 가능한 디지털 전환을 위한 실질적인 IoT 서비스 솔루션을 제공하는 것을 목표로 한다.

서론

현대 사회의 급속한 디지털화는 사물인터넷(IoT) 기술의 발전과 밀접하게 연관되어 있으며, 다양한 산업 분야에서 IoT의 적용 범위는 계속해서 확대되고 있다[1]. IoT 기술은 데이터 수집, 처리, 그리고 분석을 통해 실생활과 산업 환경에서의 의사 결정을 지원하는 핵심 도구로 자리 잡았다[2]. 하지만 IoT 시스템의 복잡성과 대규모 데이터 처리 요구는 IoT 서비스의 성능 및 확장성에 새로운 도전 과제를 제시하고 있다.

이러한 도전 과제에 대응하기 위해, 본 연구는 인공지능(AI) 알고리즘과 엣지 컴퓨팅 기술을 통합한 지능형 컨테이너 오케스트레이션 방법론을 제안한다. 이 방법론은 IoT 디바이스로부터 수집된 실시간 데이터 처리 요구와 자원 사용 패턴을 분석하고, 이를 기반으로 엣지 컴퓨팅 환경에서의 컨테이너 리소스를 동적으로 할당 및 관리함으로써 IoT 애플리케이션의 효율성을 증대시키고 네트워크 부하를 줄이는 것을 목표로 한다. 또한, 본 연구는 전체 시스템의 에너지 효율성을 개선하고, 지속 가능한 디지털 전환을 위한 실질적인 IoT 서비스 솔루션을 제공하려고 한다.

본 논문은 먼저 IoT 서비스 최적화의 현재 상황과 관련된 문제점을 검토하고, 기존 연구와의 차별점을 명확히 한다. 이어서 제안하는 AI 통합 지능형 컨테이너 오케스트레이션 방법론의 구체적인 설계 및 구현 방법을 상세히 설명할 것이다.

본론

1. 기술적 배경 및 관련 연구

현대의 IoT 서비스 환경은 방대한 양의 데이터를 실시간으로 처리하고, 다양한 사용자 요구에 신속하게 대응할 수 있는 높은 성능 및 확장성을 요구한다[3]. 이러한 요구를 만족시키기 위해, 본 연구에서는 AI 기반 알고리즘과 엣지 컴퓨팅 기술을 통합한 지능형 컨테이너 오케스트레이션 방법론을 제안한다. 기존 연구에서는 대규모 IoT 시스템에서의 데이터 처리와 자원 관리 문제를 다루었으나[4][5][6], 이들 대부분은 엣지 컴퓨팅 환경에서의 동적 리소스 관리에 중점을 두고 AI의 역할을 제한적으로 탐구하였다. 본 연구는 이러한 한계를 극복하고, AI를 활용하여 IoT 서비스의 성능 최적화와 자원 할당의 효율성을 극대화하는 새로운 접근 방식을 모색한다.

2. 방법론

본 연구에서 제안하는 방법론은 IoT 서비스의 성능 및 확장성을 극대화하기 위해 AI 기반 알고리즘과 엣지 컴퓨팅을 통합한 지능형 컨테이너 오케스트레이션 접근 방식을 중심으로 한다. 이 접근 방식의 주요 목표는 실시간 데이터 처리 요구 사항과 다양한 자원 사용 패턴을 고려하여, 엣지 컴퓨팅 환경에서 컨테이너 리소스의 동적 할당 및 관리를 최적화함으로써 IoT 애플리케이션의 효율성을 증대시키고 네트워크 부하를 줄이는 것이다.

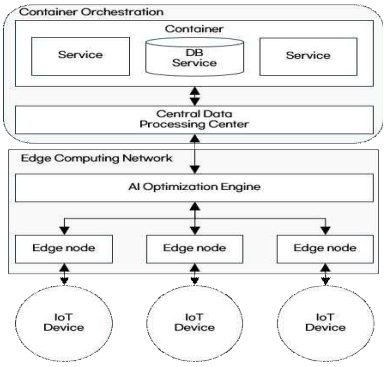

그림 1. System Architecture

그림 1은 제안하는 방법론을 적용한 전체 시스템의 구성도를 나타낸다. 이 구성도는 엣지 컴퓨팅 노드, 중앙 데이터 처리 센터, 그리고 이들 간의 데이터 흐름 및 컨테이너화된 서비스의 배치를 보여준다.

2.1 데이터 수집 및 전처리

IoT 디바이스로부터 수집된 데이터는 다양한 형식과 구조를 가진다. 그림 1에서 보듯이, 이 데이터는 먼저 엣지 컴퓨팅 노드로 전송되며, 여기에서 초기 전처리 과정을 거친다. 전처리 과정에는 노이즈 제거, 결측치 처리, 데이터 정규화 등이 포함되어 데이터의 품질을 향상시킨다.

2.2 엣지 컴퓨팅 환경에서의 데이터 처리

엣지 컴퓨팅 노드에서의 데이터 처리는 실시간 응답성 향상 및 중앙 서버의 부하 경감을 목적으로 한다[7]. 그림 1과 관련하여, 엣지 노드에서의 데이터 처리는 데이터 수집 직후에 이루어지며, 이 과정에서 컨테이너화된 마이크로서비스를 활용한다. 이러한 마이크로서비스는 특정 데이터 처리 작업을 전담하며, 필요에 따라 동적으로 스케일링될 수 있다.

2.3 AI 기반 최적화 알고리즘

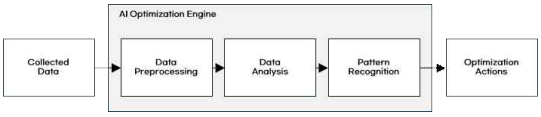

그림 2. AI Optimization Engine Process

본 연구의 핵심은 AI 기반 최적화 알고리즘을 통해 IoT 서비스의 성능을 자동으로 최적화하는 데 있다. AI 알고리즘은 그림 2에 개념적으로 나타나 있으며, 이는 데이터 패턴 분석, 예측 모델링, 자원 할당 결정 등을 포함한다. 이러한 AI 알고리즘은 엣지 및 중앙 처리 노드에서 수집된 데이터에 기반하여 동작하며, 시스템 전반의 성능 최적화를 도모한다.

2.4 컨테이너 오케스트레이션을 통한 서비스 관리

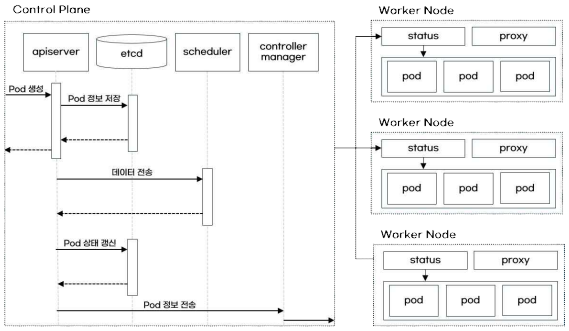

그림 3. Container Orchestration: Service

그림 3은 컨테이너 오케스트레이션을 통한 서비스 관리 방법을 시각적으로 보여준다. 여기에는 컨테이너화된 서비스의 배포, 업데이트, 모니터링 및 자동 스케일링이 포함된다. Kubernetes와 같은 오케스트레이션 도구는 이러한 작업을 자동화하고, 시스템의 안정성과 확장성을 보장한다.

결론

본 연구는 IoT 서비스의 성능 및 확장성을 극대화하기 위해 AI 기반 알고리즘과 엣지 컴퓨팅을 통합한 지능형 컨테이너 오케스트레이션 방법론을 제안하고 탐구하였다. 이 방법론은 엣지 컴퓨팅 환경에서 컨테이너 리소스의 동적 할당 및 관리를 최적화함으로써 IoT 애플리케이션의 효율성을 증대시키고, 네트워크 부하를 줄이며, 전체 시스템의 에너지 효율성을 개선하는 것을 목표로 하였다. 추후 연구로는 다양한 IoT환경에서의 적용성 및 효과를 평가하기 위한 추가 연구와 더욱 고도화된 AI 알고리즘을 개발하고 통합하여, IoT 서비스의 성능 최적화 및 자원 관리 효율성을 더욱 향상시키는 연구를 진행할 예정이다.

사사문구

"본 연구는 2024년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업의 연구결과로 수행되었음"(2022-0-01068)

참고문헌

[1] Madakam, S., Ramaswamy, R., & Tripathi, S., "Internet of Things (IoT): A Literature Review", *Journal of Computer and Communications*, Vol.3 No.5, 2015.

[2] Laghari, A. A., Wu, K., Laghari, R. A., Ali, M., & Khan, A. A., "A review and state of art of Internet of Things (IoT)", *Archives of Computational Methods in Engineering*, 2021, 1-19.

[3] 표철식, 강호용, 김내수, 방효찬, "IoT(M2M) 기술 동향 및 발전 전망", *한국통신학회지(정보와통신)*, 30(8), 2013, 3-10.

[4] 이승철, 신동하, "IoT 시스템에서 최소 자원을 사용하는 TCP/IP 구현", *한국컴퓨터정보학회논문지*, 25(10), 125-133, 2020.

[5] 나문성, 김승훈, 이재동, "클라우드 환경에서 대규모 콘텐츠를 위한 효율적인 자원처리 기법", *한국산업정보학회논문지*, 15(4), 17-27, 2010.

[6] 김선욱, 최현화, 장수민, 김대원, 차재근, "지능형 엣지 서비스를 위한 클라우드 엣지 SW 플랫폼 기술 동향", *한국통신학회지(정보와통신)*, 37(8), 46-54, 2020.

[7] 홍정하, 이강찬, 이승윤, "엣지 컴퓨팅 기술 동향", *전자통신동향분석*, 35(6), 78-87, 2020.

이산사건 시스템 형식론 기반 OSC 공법 적용 모델 구성 및 분석

김창인*, 이재윤, 장수영, 최창범

OSC Method Model Composition and Analysis using Discrete Event System Formalism based

Changin Kim, Jaiyun Lee, Suyoung Jang, Chanbeom Choi

Abstract

본 연구는 건설 산업에서 온실가스 배출을 줄이는 데 중요한 역할을 하는 이산사건 시스템 형식론 기반 Off-Site Construction(OSC) 공법 적용 모델의 구성과 분석에 초점을 맞추고 있다. 건설 산업은 전 세계 온실가스 배출의 주요 원인 중 하나이며, 이를 저감하기 위한 노력이 중요한 환경적 과제로 부각되고 있다. 이 연구에서는 OSC 공법과 전통적인 건설 방법을 비교 분석함으로써, OSC 공법이 온실가스 저감에 어떤 영향을 미치는지를 탐구한다. OSC 공법은 현장 외부에서 부재나 부품을 생산한 후 현장에 설치하는 방식으로, 공사 기간 단축, 안전사고 감소, 폐기물 배출량 감소 등 다양한 장점을 제공한다. 본 연구는 자재, 운송, 폐기물, 조립식 부품, 장비 및 기술 온실가스 배출 모델을 포함하여 OSC 공법의 다양한 온실가스 배출 모델을 구성 및 분석 하였다. 이산사건 시뮬레이션을 통해 수집된 데이터는 OSC 공법이 전통적 방법보다 환경적으로 우수함을 보여주었다. 이 연구는 건설 산업의 지속 가능한 발전을 위한 실질적인 전략 수립에 중요한 기초 자료를 제공한다.

Key Words : 이산사건 시스템 형식론, OSC 공법

1. 서론

건설 산업은 전 세계적으로 온실가스 배출에 중추적인 역할을 담당하고 있다. 이 산업 분야에서의 온실가스 저감 노력은 현대 사회의 중대한 환경적 과제로 자리 잡고 있으며, 건축 활동은 전 세계 온실가스 배출의 상당 부분을 차지하므로, 온실가스 저감을 지향하는 글로벌 노력에 있어 핵심적인 분야로 인식되고 있다[1]. 본 연구는 Off-Site Construction(OSC) 공법과 전통적인 건설 방법의 온실가스 배출량을 비교 분석하기 위해 이산사건 시스템 형식론 기반 온실가스 배출량 계산 모델들을 구성하고 시뮬레이션으로 분석하였다. 이산사건 시뮬레이션은 복잡한 건설 과정을 효과적으로 모델링하고 다양한 시나리오를 분석하여, 예상 결과를 예측하는 데 있어 강력한 도구로 활용된다. 본 연구는 건설 과정에서 발생하는 다양한 변수들을 포괄적으로 모델링하여, OSC 공법의 온실가스 배출량이 전통적인 방법보다 우위를 점할 수 있는지를 탐구한다.

OSC 공법은 현장 외부에서 부재나 부품을 생산한 후 현장에 운반하여 설치 및 시공하는 방식으로, 공사 기간 단축, 숙련공 부족 문제해결, 품질 향상, 안전사고 감소, 폐기물 배출량 감소 등 다양한 장점을 제공한다[2]. 이러한 혁신적 공법을 이산사건 시뮬레이션을 통해 체계적으로 분석함으로써, 건설 산업의 지속 가능한 발전을 위한 구체적은 전략을 모색한다.

2. 이산사건 시스템 형식론(DEVS : Discrete Event System Specification)

이산사건 시스템 형식론이란 외부의 사건 또는 시간의 흐름에 따라 내부적 상태 천이가 가능한 수학적 형식론이다. 동작을 표현한 원자모델과 모델들의 결합으로 이루어진 결합모델로 구성되어 있으며 원자 모델은 외부의 입력 사건에 따라 모델의 상태를 변화하는 외부 상태 천이 함수, 시간의 증가와 같은 내부적 요소에 따라 모델의 상태를 변화하는 내부 상태 천이 함수, 모

* 김창인, 한밭대학교 컴퓨터공학과, 학사과정, 20217140@edu.hanbat.ac.kr
이재윤, 한밭대학교 컴퓨터공학과, 석사과정, jaiyunlee@edu.hanbat.ac.kr
장수영, 한밭대학교 컴퓨터공학과, 교수, syjang@hanbat.ac.kr
최창범, 한밭대학교 컴퓨터공학과, 교수, cbchoi@hanbat.ac.kr

델의 특정 상태에 머물고 있는 시간을 기술한 시간 전진 함수, 내부 상태 천이가 발생하는 시점에 외부로 출력 사건을 발생하는 출력함수 이렇게 4가지 함수로 구성 되어있다.

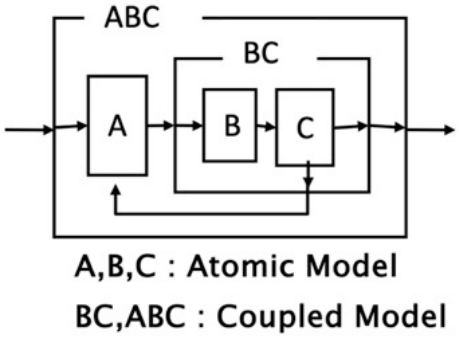

Figure 1 이산사건 시스템 형식론

3. OSC 공법 적용 모델 구성 : 온실가스 배출량

건설 단계의 온실가스 배출에 초점을 맞춘 연구는 거의 없다. 연구자들이 정의한 건설 단계 온실가스 배출 원인은 다양했다. 그래서 본 연구에서는 Chao et. al. [3]의 문헌에서 정의한 5가지 배출원인을 활용하였다.

- 자재 온실가스 배출 모델
- 운송 온실가스 배출 모델
- 폐기물 온실가스 배출 모델
- 조립식 부품 온실가스 배출 모델
- 장비 및 기술 온실가스 배출 모델

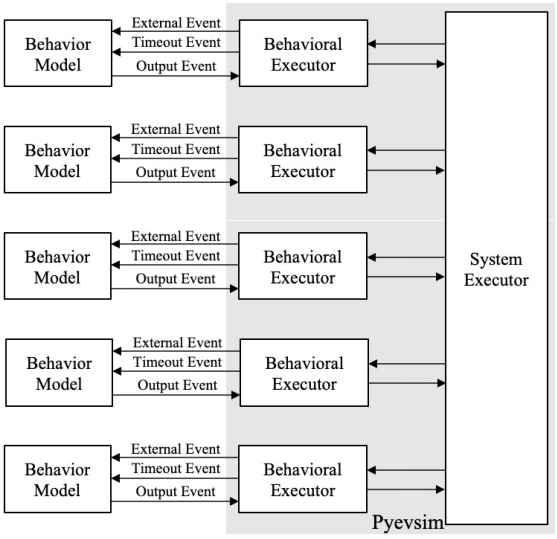

Figure 2 이산사건 시스템 형식론 기반 OSC 공법 적용 모델 구성

3.1. 자재 온실 가스 배출 모델

이 모델은 건축에 사용되는 주요 영구 자재들의 생산 과정에서 발생하는 온실가스 배출량을 계산한다. 여기에는 자재의 추출, 가공, 제조 과정에서 발생하는 온실가스가 포함된다.

3.2. 운송 온실가스 배출 모델

유통 센터에서 외부 조립식 공장이나 프로젝트 현장으로 자재를 운송하는 과정에서 연료 연소에 의해 발생하는 온실가스 배출을 평가한다. 이는 물류 및 운송 경로의 효율성을 고려한 배출량 분석에 중요하다.

3.3. 폐기물 온실가스 배출 모델

외부 조립식 공장이나 프로젝트 현장에서 매립지까지 건설 폐기물 및 토양을 운반하는 과정에서 발생하는 온실가스 배출을 측정한다. 이는 폐기물 관리 및 처리 방식의 환경적 영향을 이해하는 데 필수적이다.

3.4. 조립식 부품 온실가스 배출 모델

외부 조립식 공장에서 프로젝트 현장까지 조립식 부품을 운송하는 과정에서 발생하는 온실가스 배출을 분석한다. 이 부분은 조립식 건축 요소의 운송 효율성과 그 환경적 영향을 평가하는 데 중요하다.

3.5. 장비 및 기술 온실가스 배출 모델

장비 운영 및 건설 과정에서의 자원 및 에너지 소비에 의해 발생하는 온실가스 배출을 평가한다. 여기에는 디젤, 석유, 전기, 물 사용 등이 포함되며, 이는 건설 현장의 에너지 효율성 및 환경 영향을 이해하는 데 기여한다.

4. 사례연구 : 이산사건 시뮬레이션을 통한 건설 공법 비교 분석

본 연구에서는 Off-Site Construction(OSC) 공법과 전통적인 건설 방법의 성능을 비교하기 위해 이산사건 형식론 기반 OSC 공법 적용 모델을 활용하여 이산사건 시뮬레이션에 적용하였다. OSC 공법과 전통적인 방법의 성능을 정확하게 비교 평가하는 데 초점을 맞추었고 각 공법의 성능은 이산사건 시스템 모델을 통해 생성된 데이터에 기반한 종합적인 성과 점수로 계산되며, 이는 효율적이고 환경적으로 지속 가능한 건설 방법을 결정적으로 식별하는 데 중요한 역할을 한다. 이를 위해 Shu et. al. [4]의 문헌에서 OSC 공법과 전통적인 방법 사용했을 경우 발생하는 평당 비용 데이터를 가져와 새로운 모델을 추가하였다. 그 후 건설 현장의 다양한 조건을 반영한 'UrbanLarge' 및 'SuburbanSmall'과 같

은 다양한 건설 환경을 가정한 시나리오를 설정하였다.

모델은 시뮬레이션 실행 중에 발생하는 각 이벤트에 따라 시간의 흐름을 추적한다. 시뮬레이션의 각 단계에서, 선정된 파라미터 값을 기반으로 품질 점수를 계산하고 이를 결과 데이터에 기록한다. 이후, 시뮬레이션은 정해진 기간 동안 실행되어 다수의 시나리오를 통해 축적된 데이터를 생성한다. 이를 CSV 파일로 저장하였다. CSV 파일로부터 추출된 데이터는 Python의 Matplotlib 라이브러리를 활용하여 시각화된다. 시각화 과정은 데이터의 이해를 돕고, 공법별 성능차이를 명확히 드러내는 데 중점을 두었다.

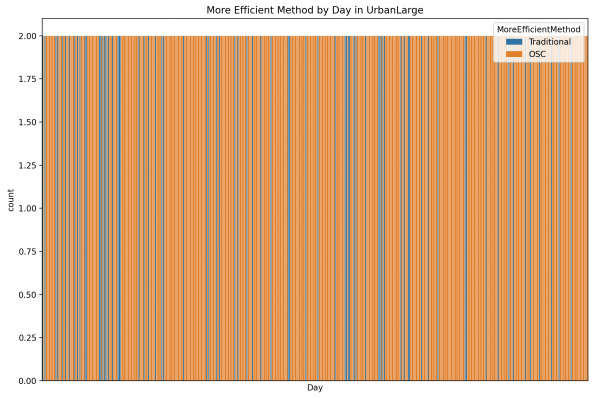

Figure 3 시각화 자료

시각 자료로 제공되는 그래프는 각 공법의 품질 점수에 대한 공법별 성능 차이를 명확하게 보여준다. 이 자료를 통해 주황색의 OSC 공법이 전통적 방법보다 우수한 성능을 보이는 경우를 직관적으로 파악할 수 있다.

5. 결론

본 연구는 이산사건 시스템 형식론 기반 OSC 공법 적용 모델을 구성하고 이를 활용하여 이산사건 시뮬레이션에 적용하였다. 이 과정을 통해, OSC 공법이 전통적인 건설 방법에 비해 온실가스 배출을 줄이는 데 얼마나 효과적인지를 탐구하였다. 추후 연구로 기후 변화 적응 모델, 생애주기 분석 모델 등 여러 모델을 추가적으로 통합하고 분석함으로써 OSC 공법이 건설 산업의 지속 가능한 발전에 기여하는 방식을 더욱 명확히 이해하고 연구를 확장시킬 것이다.

참고문헌

[1] 전수진, 정종홍. (2022). 건설분야 탄소중립(Net-Zero) 정책 동향과 스마트건설기술의 역할. 한국건설관리학회, *23*(2), 49-54.

[2] 차희성, 이성호. (2022). Off-Site Construction(OSC) 기반 건축생산방식 성과관리 시스템 구축 방안-PC 공동주택 사례를 중심으로-. 건축시공, *22*(2), 14-24.

[3] Chao Mao, & Qiping Shen, & Liyin Shen, & Liyaning Tang. (2013). Comparative study of greenhouse gas emissions between off-site prefabrication and conventional construction methods: Two case studies of residential projects. Energy and Buildings, Vol 66, 165-176

[4] Wang, Shu & Zhang, Hengchun & Wang, Chan & Wu, Yuanyuan. (2020). Cost Analysis Between Prefabricated Buildings and Traditional Buildings. IOP Conference Series: Materials Science and Engineering. 768. 052090. 10.1088/1757-899X/768/5/052090.

[5] Pyevsim : https://github.com/eventsim/pyevsim

프롬프트 엔지니어링을 활용한 업무 자동화 시스템 설계 및 구축 : 대학 업적 평가 행정업무 중심

김창인, 이재윤, 김진우, 허유정, 장수영, 최창범
한밭대학교 컴퓨터공학과
E-mail: 20217140@edu.hanbat.ac.kr, E-mail: jaiyunlee@edu.hanbat.ac.kr,
E-mail: 20181586@edu.hanbat.ac.kr, E-mail: 20211939@edu.hanbat.ac.kr,
E-mail: cbchoi@hanbat.ac.kr, E-mail: syjang@hanbat.ac.kr,

요 약

본 연구는 대학의 연구 성과 평가와 행정업무의 자동화를 목표로 프롬프트 엔지니어링과 OpenAI의 GPT-3.5 모델을 활용한 업무 자동화 시스템의 설계 및 구축을 중점으로 다룬다. 본 시스템은 논문의 사사표기 판단 과정을 자동화하며, 한밭대학교의 논문 데이터를 기반으로 95%의 정확도로 식별하였다. 이를 통해 전체 업무량을 약 61% 감소시키는 효과를 보였다. 복잡한 대학 행정 업무를 어떻게 자동화할 수 있는지에 대한 탐구와 프롬프트 엔지니어링의 기술적 원리를 상세하게 다루는 본 연구는 대학의 행정 업무 효율성을 향상시키는 방안을 제시한다.

Prompt Engineering, OpenAI GPT-3.5, RPA

Ⅰ. 서 론

대학에서의 연구 성과 평가와 행정업무 처리는 중요한 주제로서, 교육 기관에서 효과적이고 효율적인 방법으로 이를 수행하는 것은 교육의 품질과 연구 업적의 향상에 기여한다. 이러한 작업은 많은 양의 데이터와 복잡한 처리 과정을 포함하므로, 전통적인 수동 방법으로는 상당한 시간과 인적 자원이 많이 소요된다.

프롬프트 엔지니어링은 자동화와 인공 지능 기술을 결합한 현대적인 기법으로, 특정 업무 프로세스의 자동화를 가능하게 한다. 대학의 행정업무와 연구 성과 업무의 복잡성과 규모가 커지고 있으며, 이로 인해 비용과 노력의 증가가 불가피하다. 이러한 문제를 해결하기 위해 프롬프트 엔지니어링을 활용한 업무자동화 시스템을 설계하고 구축하는 것은 매우 중요한 과제로 여겨지고 있다.

본 연구는 프롬프트 엔지니어링을 대학의 업적 평가 행정업무에 중심으로 적용하는 방안에 대한 심층적인 분석을 제시한다. 특히, 복잡하고 시간이 많이 소요되는 대학 행정업무를 어떻게 자동화할 수 있는지에 대한 탐구하며, 프롬프트 엔지니어링의 기술적 원리와 실제 사례를 통한 구현 방법을 상세하게 다룬다.

Ⅱ. 본 론

1. OpenAI GPT-3.5

OpenAI의 GPT-3.5는 그 이전 버전인 GPT-3을 기반으로 개선된 고수준 언어 생성 모델로, 프롬프트 엔지니어링의 핵심 구성 요소 중 하나이다. 이 모델의 두드러진 특징 중 하나는 "few-shot learning" 능력으로, 제한된 수의 예시만을 통해 새로운 작업을 이해하고 수행할 수 있다는 것이다[1]. 본 연구에서는 이 능력을 활용하여 대학 행정업무의 일부, 특히 논문의 사사표기 판단 과정을 자동화하였다[1].

2. OpenAI GPT-3.5 모델을 활용한 업무 자동화 시스템의 구조

본 연구의 자동화 시스템은 GPT-3.5 모델에 의존하기 때문에 텍스트 분석 및 처리가 중요한 부분을 차지한다. 특히 키워드를 사용하여 모델에 원하는 응답을 유도하는 프롬프트 엔지니어링은 필수적인 과정이다. 또한 GPT-3.5 모델의 토큰 제한으로 인해 텍스트를 정제하고 최적화해야 하며, 이 과정은 효율적인 분석과 정확한 결과 도출에 기여한다.

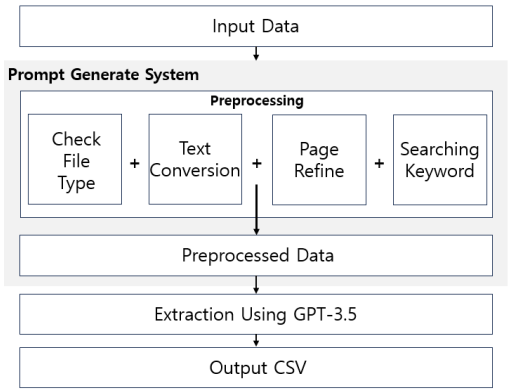

그림 1 시스템 전체 구조

본 연구에서 설계한 자동화 시스템은 논문, 보고서 등의 다양한 형태의 데이터를 입력으로 받아들이고, 이를 초기 전처리 과정을 통해 텍스트 형태로 변환한다. 변환된 텍스트는 GPT-3.5가 이해하기 쉽게 추가 전처리가 이루어지며, 여기서 프롬프트의 역할이 중요하게 작용한다. 프롬프트는 구조화된 질문이나 명령어로서, 텍스트 데이터를 GPT-3.5가 분석 가능한 형태로 가공하는 데 필수적인 역할을 한다. 전처리된 텍스트 데이터는 분석 가능한 형태로 재구성된 후 GPT-3.5를 통해 분석된다. 본 과정에서는 복잡한 패턴 인식과 분석 작업이 수행되며, 예를 들어 논문의 사사표기 판단 과정은 여기에서 처리된다.

그림 1은 시스템의 전체 구조를 시각적으로 나타낸 것이다. 이 그림은 각 작업의 흐름과 서로 어떻게 연결되는지를 명확하게 보여주며, 복잡한 시스템을 이해하는 데 도움을 준다.

Ⅲ. 사례연구

본 연구에서는 사례연구로 한밭대학교 성과를 위한 논문 데이터를 활용하였다. 논문 데이터는 한밭대학교 산업협력단에 요청하여 전체 논문 중 일부인 31편을 제공받아 연구를 진행하였다. 31개의 논문을 본 연구에서 개발한 업무 자동화 시스템에 적용했다. 이 중 실제로 사사표기가 포함된 논문은 20개이며, 나머지 11개 논문에는 사사표기가 존재하지 않았다. 31개의 논문을 처리한 결과, 사사표기가 있는 20개의 논문 중 19개를 정확하게 판별했다. 이는 시스템이 사사표기가 있는 논문을 95%의 정확도로 식별할 수 있음을 보여준다. 반면, 사사표기가 없는 11개의 논문은 정확하게 판별하였지만 이들 또한 수작업으로 검토해야한다.

다음 표 1는 이러한 결과를 나타낸 것이다.

표 1 논문의 사사표기 판별 결과

	전체 논문 수	판별된 사사표기 있는 논문 수	판별된 사사표기 없는 논문 수	수작업 검토 필요 논문 수
실제	31	20	11	
시스템 판별 결과	31	19	11	12
판별 정확도(%)	-	95%	100%	

이를 통해, 총 31개의 논문 중에서 19개는 시스템이 자동으로 처리하고, 나머지 12개에 대해서만 수동으로 검토가 요구된다. 이는 업무량을 약 61%(19/31*100)를 감소시키는 효과를 보여준다.

Ⅳ. 결 론

본 연구는 프롬프트 엔지니어링을 활용한 업무자동화 시스템의 설계와 구축을 중심으로 대학의 업적 평가 행정업무에 적용하는 방안을 제시하였다. 특히 OpenAI의 GPT-3.5 모델을 이용하여 논문의 사사표기 판단 과정을 자동화하는 시스템의 구조를 설명하였으며, 한밭대학교의 논문 데이터를 이용한 사례연구를 통해 시스템의 실질적 효과를 검증하였다.

본 연구의 주요 결과로는 사사표기가 있는 논문을 95%의 정확도로 식별할 수 있는 시스템의 능력과 전체 업무량을 약 61% 감소시키는 효과가 드러났다. 그럼에도 불구하고, 본 연구에서의 시스템은 아직 완전한 자동화를 이루지 못하고 일부 수작업 검토가 필요한 부분이 있다[2]. 이는 시스템의 추가 개선과 최적화를 통해 해결할 수 있는 문제로 보이며, 추후 연구에서 이러한 한계점을 극복하고 더 효과적인 업무 자동화 방안을 모색할 것이다.

References

[1] Brown, Tom B et al, “Language models are few-shot learners,” *Advances in Neural Information Processing Systems 33,* vol.N/A, N/A, 3-6, 2020.

[2] 윤상오, 정필운, 이해원, 박소영, “인공지능 기반 자동화행정의 주요쟁점에 관한 연구,” *Korean Public Management Review,* vo.34, no.3, pp.109-132, 2020