백서 | 벤치마크

RNGD meets Backend.AI:

주권 AI 시대를 위한 AI 추론 인프라

저자:
래블업 | 허진호, 조규진, 김준기, 신정규
퓨리오사에이아이 | 최현식, 강지훈, 김종욱

2026년 7월 v1.0

목차

Executive Summary

생성형 AI 서비스를 직접 운영하는 조직에게는 어떤 모델을 선택할지 뿐 아니라, 그 모델을 어떤 인프라에서 안정적으로 서빙 할지도 중요한 설계 과제입니다. 추론은 서비스 운영 기간 내내 지속되는 상시성 워크로드이며, 운영 비용은 연산 자원 자체뿐 아니라 전력과 데이터센터 수용 조건의 영향을 크게 받습니다. 일부 최신 GPU는 새로운 냉각 방식을 요구하는 것과 동시에 높은 전력량을 사용합니다. 이러한 제약은 신규 설비 투자나 운영 구조 변경으로 이어질 수 있으므로, 추론 인프라를 검토할 때는 연산 성능뿐 아니라 전력 밀도와 배치 적합성도 함께 고려해야 합니다.

이 백서에서는 FuriosaAI의 데이터센터 추론용 가속기 RNGD를 래블업의 AI 인프라 운영 플랫폼 Backend.AI로 운영하는 구성에 대해 살펴봅니다. RNGD는 카드당 TDP 180W의 추론 전용 가속기이며, Backend.AI는 이기종 AI 가속기를 단일 플랫폼에서 할당, 격리, 서빙 할 수 있도록 지원하는 운영 소프트웨어 계층입니다.

RNGD와 Backend.AI의 스위트(Suite) 구성은 주권 AI(Sovereign AI) 관점에서도 의미가 있습니다. 국내에서 설계한 추론 칩과 국내에서 개발한 오픈소스 기반 플랫폼의 조합은 폐쇄망 운영, 해외 벤더에 종속되지 않는 공급망 선택지, 코드 수준의 투명한 검증 가능성, 그리고 운영 자립이라는 주권 AI의 실무 요건에 대응할 수 있습니다.

래블업과 FuriosaAI는 RNGD AI 가속기 및 동급 사양 가속기에서 Qwen3-32B 모델을 FP8 정밀도로 서빙하는 벤치마크를 수행했습니다. RNGD 4장은 NVIDIA RTX PRO 6000 Blackwell Server Edition 4장 대비 동시 요청 256 기준 약 95%의 처리량(2,336 대 2,467 tok/s)을 기록, 가속기 합산 전력은 56~70% 수준에 머물렀습니다. 이를 전력당 처리량으로 환산하면 측정한 모든 동시성 구간에서 RNGD가 약 1.3~1.5배 높았으며, 첫 토큰 응답 시간(TTFT) 역시 전 구간에서 RNGD가 짧았습니다.

해당 백서는 위 벤치마크 결과를 바탕으로, 추론 전용 가속기와 이기종 오케스트레이션 플랫폼을 결합한 인프라 구성이 성능, 전력, 운영 측면에서 어떤 조직에 적합한지를 살펴봅니다.

1. 서론: 추론 시대의 인프라 제약

1.1 상시 워크로드가 된 LLM 추론

추론은 이제 AI 인프라의 경제성을 좌우하는 핵심 변수입니다. 기업의 AI 투자가 오랫동안 모델 학습에 집중되어 왔지만, LLM 서비스가 사내 어시스턴트, 고객 응대, 문서 처리와 같은 일상적인 업무 영역으로 확산되면서 최근 추론으로 중심축이 이동하고 있습니다. 학습과 추론은 워크로드의 성격만큼이나 비용의 성격도 다릅니다. 학습은 모델 개발 프로젝트가 끝나면 마무리되는 자본 지출(CapEx)에 가깝습니다. 반면 추론은 운영 비용이 서비스의 규모에 따라 증가하는 운영 지출(OpEx)입니다. 결국 AI 인프라의 경쟁력은 모델 성능 자체보다, 추론 단계의 단위 효율과 확장성을 얼마나 일관되게 확보하느냐에 달려 있습니다.

1.2 전력과 냉각의 제약

추론 용량 확대는 가장 직관적으로는 AI 가속기 증설을 통해 달성할 수 있지만, 실제 환경에서는 전력과 냉각 제약으로 인해 그 확장이 쉽지 않습니다. 최근 출시되는 하드웨어는 고밀도 구성 시 수냉 설비를 전제하는 경우가 많습니다. 공랭 기준으로 설계된 상면에 이러한 장비를 도입하기 위해서는 냉각 설비를 개조해야 하는데, 이는 늘어난 도입 기간과 비용을 마주하게 됩니다. 국내 기업과 공공기관의 전산실 상당수가 공랭 설계라는 점을 감안하면, 추론 인프라를 계획하는 입장에서 액체냉각을 요구하는 하드웨어로의 증설은 쉽게 고를 수 있는 선택지가 아닙니다. 최근 출시되는 고성능 GPU의 경우 서버 당 적게는 7kW에서 많게는 15kW까지의 최대 전력을 요구합니다. 전력량 확장 여력이 크지 않은 기존 환경에서 이러한 고전력 요구사항은 AI 가속기의 확장을 발목 잡을 수 있습니다.

AI 인프라를 직접 보유하는 대신 클라우드로 우회하는 방법도 있지만, 한계도 뚜렷하게 나타납니다. 총 비용도 따져봐야 하고, 금융, 공공, 의료와 같은 규제 산업에서는 데이터를 외부로 보내는 것이 허용되지 않는 경우가 많아 폐쇄망 온프레미스를 선택해야 하는 경우가 많습니다. 여기에 최근에는 데이터뿐 아니라 인프라와 기술 스택 전체의 통제권을 확보해야 한다는 주권 AI 논의까지 더해지고 있습니다. 주권 AI에 대한 내용은 1.4절에서 확인할 수 있습니다.

1.3 추론 전용 가속기, 그리고 운영 소프트웨어의 필요성

추론 전용 가속기는 이러한 제약을 겨냥하여 등장한 하드웨어입니다. 추론 전용 가속기는 범용적인 GPU 연산을 수행하는 대신, 이미 학습이 끝난 AI 모델이 입력을 받아 결과를 추론해 내는 단계를 더 빠르고 효율적으로 처리하도록 설계된 전용 하드웨어입니다. 학습에 필요한 역전파와 옵티마이저 상태 관리 같은 넓은 연산 범위를 다루지 않는 대신, 추론 단계에서의 지연시간, 처리량, 전력 효율에 가속기의 설계를 최적화한 구조입니다.

RNGD 역시 이러한 설계 철학을 따르는 추론 가속기입니다. 추론 전용 가속기는 범용성을 덜어낸 만큼 동일한 서빙 작업을 더 적은 전력으로 처리할 수 있습니다. 대규모 추론 서비스가 상시 운영되는 환경에서 이러한 전력 효율의 차이는 누적되는 비용에 반영되고, 전반적인 운용효율 계산의 차이로 이어집니다.

그러나 추론 전용 가속기가 효율적인 서빙을 가능하게 하더라도, 새로 도입한 가속기가 조직의 기존 인프라와 자연스럽게 결합되지 않으면, 하드웨어의 효율을 운영 수준에서 실현하기 어렵습니다. 결국 추론 전용 가속기의 실질적인 도입은 칩의 성능뿐 아니라, 그것을 기존 GPU와 동일한 방식으로 다룰 수 있는 운영 소프트웨어의 존재 여부에 달려 있는 것입니다.

1.4 선택을 넘어 필수가 된 주권 AI

주권 AI는 데이터, 모델, 인프라, 운영 등 AI 시스템의 핵심 구성 요소를 국가 또는 조직의 통제 범위 안에 두는 접근 방식입니다. 가속기 하드웨어와 운영 소프트웨어 양쪽에서 특정 해외 공급망에 대한 의존도가 높아질수록 수급 불안, 수출 규제, 가격 변동과 같은 리스크도 함께 커지게 됩니다. 최근에는 글로벌 공급망의 불확실성이 커지고, 보안과 안보를 고려한 지정학적 관점까지 중요해지면서 데이터 저장 위치, 모델 배포와 업데이트, 접근 권한, 운영 로그, 장애 대응, 공급망 선택에 이르기까지 다양한 요소를 직접 통제하려는 움직임이 확산되고 있습니다.

이러한 흐름에서 조직이 추론 인프라를 도입할 때, 해당 인프라를 자체 통제 하에 안정적으로 운영할 수 있는지 여부는 성능이나 비용에 이은 또 다른 평가 기준이 됩니다. Backend.AI와 RNGD는 각각 국내에서 개발된 운영 플랫폼과 국내에서 설계된 추론 가속기로, 이 둘의 결합은 국산 공급망 확보, 폐쇄망 운영, 코드 수준의 검증 가능성 등 주권 AI의 실무 요건에 대응하는 인프라 구성을 가능하게 합니다. 이 백서는 이러한 조합이 기존 GPU 인프라와 어떻게 통합되고, 실제 추론 워크로드에서 어떤 성능, 효율 특성을 보이는지를 벤치마크 데이터와 함께 검토하고자 합니다.

2. 솔루션 개요

래블업과 FuriosaAI는 1장에서 다룬 전력 및 상면 제약과 운영 소프트웨어의 부재에 대응하기 위해 RNGD와 Backend.AI를 하나의 통합 구성으로 제시합니다. RNGD 8장을 탑재한 NXT RNGD Server 위에서 사용자는 FuriosaAI의 컴파일러로 모델을 빌드하고, Furiosa-LLM을 통해 서빙할 수 있습니다. 여기에 엔터프라이즈 AI 운영 플랫폼인 Backend.AI가 더해지면 자원 할당, 스케줄링, 사용량 측정, 모델 서빙 엔드포인트 관리 등 조직 단위의 운영 기능을 활용할 수 있습니다.

3. Backend.AI: 이기종 가속기에서 대규모 LLM 토큰을 서빙하는 운영 플랫폼

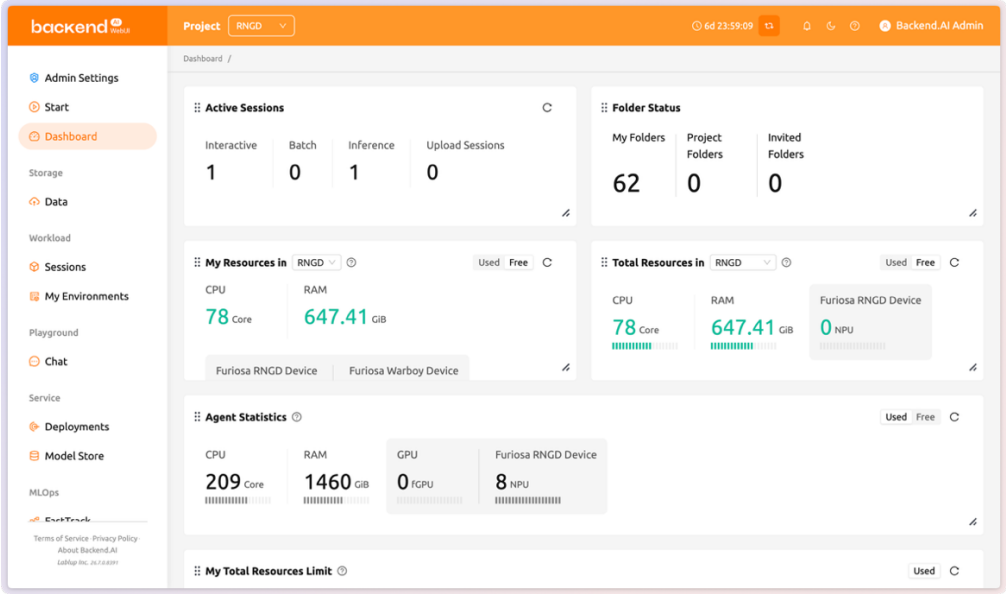


그림 1. Backend.AI의 WebUI Dashboard

Backend.AI는 AI 개발과 서비스 운영을 아우르는 엔터프라이즈 AI 인프라 운영 플랫폼입니다. GPU, NPU, TPU 등 다양한 형태의 이기종 AI 가속기를 하나의 운영 체계 안에서 다룰 수 있도록 설계되었으며, 다중 사용자 환경에서 자원을 효율적으로 배분하고 통제하는 실행 기반을 제공합니다. 또한 하드웨어 구성, 프레임워크 의존성, 사용자별 자원 배분, 정책 적용 등 AI 인프라의 복잡한 요소를 운영 계층에서 추상화하여, 개발자와 데이터 과학자가 모델 개발과 실행에 집중할 수 있는 환경을 제공합니다. Python, R, C/C++, Java, Rust 등 다양한 프로그래밍 언어와 PyTorch, Megatron-LM, Furiosa-LLM, vLLM 같은 다양한 AI 라이브러리를 지원하는 등 특정 프레임워크에 종속되지 않는 범용 실행 환경을 지향하며, 다양한 설치환경을 지원하여 온프레미스, 클라우드, 그리고 하이브리드 환경에서도 동일한 경험을 목표로 Backend.AI를 사용할 수 있습니다.

이번 장에서는 Backend.AI의 핵심 구성 요소를 순서대로 살펴봅니다. 워크로드를 다루는 단위인 '세션', 자원을 관리하기 위한 오케스트레이터 'Sokovan', 이기종 가속기를 균일하게 다루기 위한 '자원 추상화', 조직 규모의 운영을 지원하는 '멀티테넌시', 가속기 확장을 위한 '플러그인 아키텍처', 그리고 '모델 서빙과 운영' 기능에 대해 확인할 수 있습니다.

3.1. 워크로드의 단위 추상화: 세션

Backend.AI에서 모든 연산 작업은 '세션'이라는 단위로 표현됩니다. 세션은 자원 사양(가속기 몇 장, CPU 몇 코어, 메모리 얼마)과 실행 이미지, 마운트할 데이터를 하나로 묶은 실행 요청이며, Backend.AI는 이 요청을 컨테이너 단위로 구체화합니다. 세션은 용도에 따라 인터랙티브, 배치, 인퍼런스 3가지 유형으로 분류됩니다.

세션 유형	용도	생애주기 특성
Interactive	개발 / 실험 (Jupyter, IDE, Terminal)	사용자가 직접 접속하며, 유휴 판정에 따라 자원 회수 대상이 됨
Batch	학습 / 배치 작업	스크립트 완료시 세션 종료, 자원 자동 반환
Inference	모델 서빙	외부 접속 가능한 엔드포인트 노출, 가속기 사용량 및 추론 메트릭 기반 스케일링

표 1. Backend.AI가 지원하는 세션 유형

추론 전용으로 구성된 클러스터라 하더라도 실제 운영 환경에서는 모델 갱신을 위한 인터랙티브 세션이나, 주기적인 배치 작업이 함께 실행될 수 있습니다. 세션 유형별로 서로 다른 생애주기 정책(유휴 자원 회수, 자동 종료, 상시 자원 점유 유지)을 플랫폼이 구분하여 적용해야 가속기를 효율적으로 운용할 수 있습니다.

Backend.AI는 세션의 전체 생애주기를 단계별로 관리합니다. 자원 할당 대기(PENDING)부터 이미지 준비, 컨테이너 생성, 워크로드 실행(RUNNING), 자원 반납(TERMINATED)에 이르기까지 단계별 상태를 기록합니다. 운영자는 어떤 세션이 어느 단계에서 왜 머물러 있는지 추적할 수 있습니다. 멀티노드 세션의 경우 여러 개의 커널이 자원 요구량 대로 생성되고 전용 격리 네트워크로 묶여, 분산 추론이나 분산 학습을 하나의 작업 단위로 관리할 수 있습니다.

3.2 Sokovan 스케줄링 파이프라인과 자원 예약

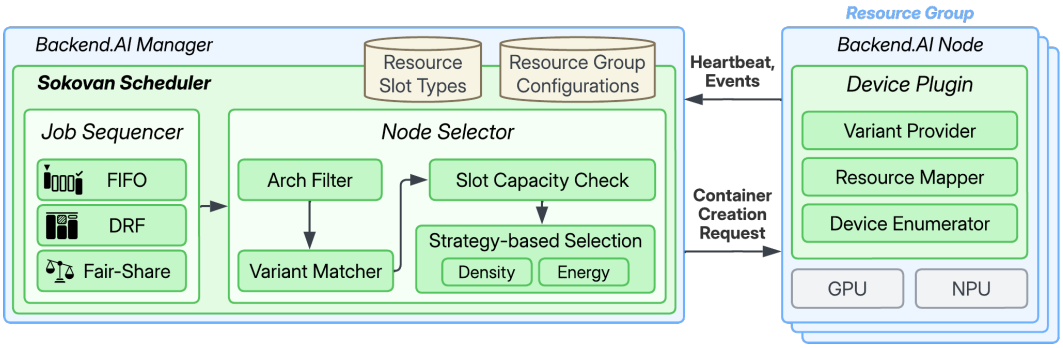


그림 2. Backend.AI의 Sokovan Orchestrator

세션을 노드에 배치하는 역할은 Sokovan 오케스트레이터가 담당합니다. Sokovan은 대규모 AI 및 GPU 워크로드를 위해 래블업이 독자 설계한 컨테이너 오케스트레이터입니다. 클러스터 수준과 노드 수준의 두 계층 스케줄러를 통해 작업 밀도와 우선순위를 조절하며, 멀티 노드, 멀티 테넌트 인지 스케줄링을 지원하기 때문에 다중 사용자와 작업이 병렬로 존재하는 환경에서 자원 활용률과 실행 효율을 높이는 데 초점을 두고 있습니다.

Sokovan은 가속기 및 워크로드 특성을 인지하는 스케줄링을 제공합니다. 여러 세션 실행 요청이 대기열에 들어오면, Sokovan은 이를 어떤 순서로 처리할지 결정합니다. 운영자는 환경에 맞는 스케줄링 방식을 선택할 수 있습니다. 선입선출 원칙을 지키는 FIFO (First-In-First-Out) 방식, 자원별 점유율이 가장 높은 사용자를 후순위로 스케줄링하는 DRF (Dominant Resource Fairness) 방식, 과거 사용량을 기반으로 시간 감쇠를 적용하는 Fair Share 방식 중에 적합한 방식을 선택할 수 있습니다.

스케줄링 방식	동작 방식
FIFO (First-In-First-Out)	요청이 들어온 순서대로 처리
DRF (Dominant Resource Fairness)	자원별 점유율이 가장 높은 사용자를 후순위로 배치하여 자원 독점 방지
Fair Share	누적 사용량에 시간 감쇠를 적용, 자원을 적게 사용한 사용자를 우선 배치

표 2. 스케줄링 방식의 종류

여기에 더해, 우선순위가 높은 세션이 대기중일 때 이미 실행중인 낮은 우선순위 세션의 자원을 회수하여 재배치하는 선점 (preemption) 정책도 선택적으로 적용할 수 있습니다. 자원이 부족한 상황에서도 긴급한 워크로드나 상위 우선순위 작업이 대기 없이 실행될 수 있도록 보장하는 기능으로, 추론 서비스와 같이 중단 없는 자원 확보가 요구되는 환경에서 유리합니다.

Sokovan의 스케줄링 정책은 클러스터를 독립적인 정책 단위로 나눈 논리적 파티션인 '자원 그룹' 단위로 적용됩니다(6.1절 참조). 세션 배치 전에 Sokovan의 Validator는 테넌트의 동시 세션 수, 쿼터, 세션 간 의존성,

대기 한도를 확인하여 즉시 배치할 수 없는 요청을 사전에 걸러냅니다. 검증을 통과한 요청에 대해서는 배치 가능한 노드를 탐색한 뒤, 자원 그룹별로 지정된 배치 전략에 따라 노드를 선택합니다. 워크로드를 소수 노드에 밀집시켜 단편화를 줄이는 집중 배치(bin-packing) 방식과 노드 사이 부하를 고르게 나누는 분산 배치(spreading) 전략 중 적합한 전략을 자원 그룹별로 지정할 수 있습니다.

Sokovan의 자원 예약은 무결성을 보장하는 업데이트로 처리되므로 사용자가 할당받은 자원은 해당 세션이 독점적으로 사용합니다. 동일한 가속기가 두 세션에 이중 배정되는 오버커밋도 원천적으로 발생하지 않습니다.

3.3 자원 추상화와 컨테이너 실행

Backend.AI의 강력한 자원 추상화는 여러 벤더의 가속기를 동일한 방식으로 다룰 수 있는 바탕이 됩니다. 플랫폼은 모든 자원을 슬롯(Slot)이라는 단위로 표현하며, 슬롯 유형은 가속기의 개수 단위 (CPU 코어 수, 가속기 장 수)와 바이트 단위 (메모리)로 나뉩니다.

추상화된 자원을 실제로 격리 컨테이너에 할당하는 것은 각 가속 노드에 상주하는 Backend.AI Agent 의 역할입니다. Backend.AI Agent는 각 가속기와 이를 활용하는 컨테이너의 생애주기를 관리하고, 활용률, 전력, 온도와 같은 메트릭을 수집하여 제어 평면에 보고합니다. Backend.AI Agent는 AI 가속기를 컨테이너에 할당할 때 NUMA 토폴로지를 반영합니다. 여러 장의 가속기를 동시에 사용하는 학습 및 추론 워크로드에서는, 같은 NUMA 노드에 속한 가속기끼리 묶어 배치하는 것이 성능에 직접적인 영향을 주며, Backend.AI Agent는 이러한 최적화를 기본으로 지원합니다.

사용자는 세션을 생성할 때 Furiosa-LLM이나 PyTorch 같은 프레임워크와 버전을 이미지 단위로 선택할 수 있습니다. 환경이 세션 단위로 격리되므로 팀마다 다른 버전 조합을 같은 클러스터에서 병행할 수 있습니다. Backend.AI는 컨테이너 실행 시 필요한 런타임 구성 요소를 읽기전용으로 주입하므로, 이미지 안에 Backend.AI의 구성 요소를 미리 포함해 둘 필요가 없습니다. 따라서 벤더가 배포하는 공식 SDK 이미지를 수정 없이 그대로 세션 이미지로 사용할 수 있습니다.

3.4 멀티테넌시 대응

Backend.AI는 도메인, 프로젝트, 사용자로 구성된 다층 테넌트 체계를 기반으로 자원 사용량을 관리합니다. 따라서 여러 층위로 구성된 복잡한 자원 관리 정책을 구현할 수 있습니다. 각 계층에 독립적인 자원 한도를 설정할 수 있으므로, 조직 수준과 개인 수준의 정책을 중첩하여 적용할 수 있습니다. 예를 들어, A 부서에 RNGD 16장을 할당하고 해당 부서 내 개인별로는 동시에 최대 4장까지 사용하도록 제한하는 구성이 가능합니다. 3.2절에서 다룬 FIFO, DRF, Fair Share 등 여러 자원 스케줄링 정책과 결합하면 시간에 따른 불균형도 방지할 수 있습니다. 예를 들어 GPU 자원 그룹에는 장시간 학습 작업에 맞는 FIFO를, RNGD 자원 그룹에는 다수 팀의 추론 세션이 경합하는 상황에 맞는 Fair Share를 적용하는 식으로 가속기 특성과 워크로드 성격에 맞춰 정책을 따로 가져갈 수 있습니다. 자원 그룹의 구성과 운영 방식은 6.1절에서 확인할 수 있습니다.

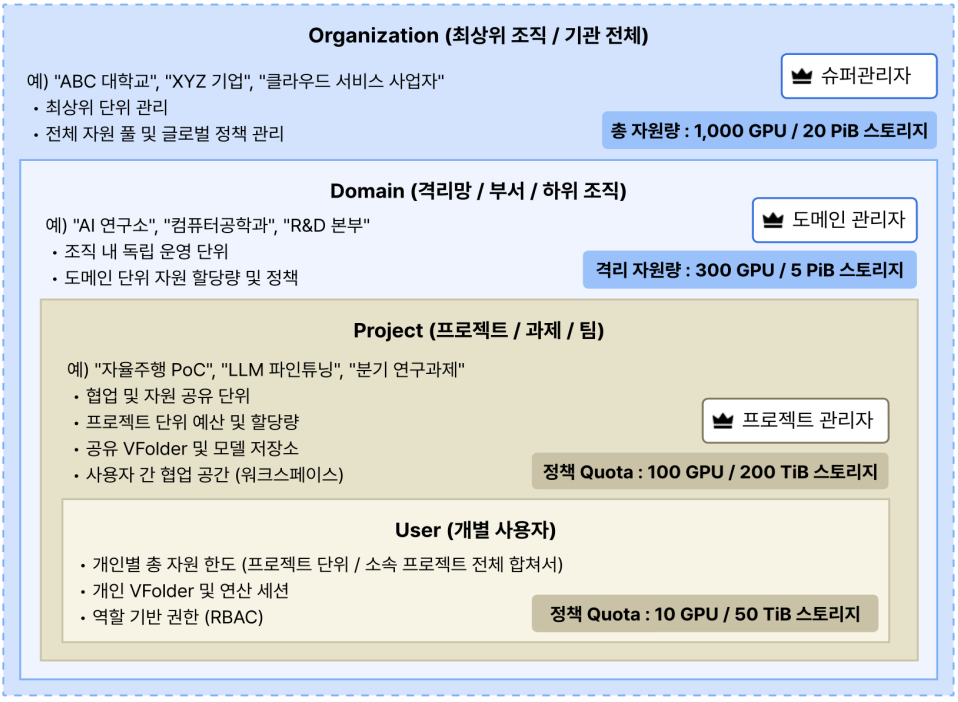


그림 3. Backend.AI의 테넌트 계층

테넌트 계층	대응 조직 단위	대표 정책 항목
도메인	회사 / 기관 (최상위 격리 경계)	도메인 수준 자원 정책
프로젝트	팀 / 부서	프로젝트 쿼터, 스토리지 총량
사용자	개인 계정	사용자별 쿼터, 폴더 수 한도, 동시 세션 수

표 3. Backend.AI의 테넌트 계층

Backend.AI는 할당된 자원의 사용량과 필요에 따라 할당된 자원을 종료하거나 수거하는 수단도 함께 지원합니다. 예를 들어, 마지막 트래픽이 감지된 이후 정책으로 설정된 시간이 경과되거나, CPU, 메모리 등 가속기 활용률이 설정된 임계치 미만으로 지속되거나, 세션에 할당된 최대 실행 시간을 초과하는 경우에는 스케줄러 판단 하에 할당된 자원이 회수됩니다.

유휴 판정 기준	동작 방식
네트워크 활동 미감지	마지막 트래픽 이후 설정된 시간 경과시 자동 회수
활용률 저조	CPU, 메모리, 가속기 활용률이 임계치 미만인 경우 자동 회수
세션 수명 초과	유형별 최대 실행 시간 도달하는 경우 자동 회수

표 4. Backend.AI의 유휴 활동 판정 기준

오판을 방지하기 위한 보호 로직도 적용되어 있습니다. 가속기가 할당되지 않은 세션은 가속기 활용률 기준에서 제외되며, 새로 생성된 세션에는 유예 기간이 부여되어 초기 구동 단계에서 회수되지 않습니다. 회수 임계치는 사용자별 자원 정책으로 개별 관리할 수 있어, 요금제나 팀별로 차등 운영이 가능합니다.

3.5 가속기 플러그인 아키텍처

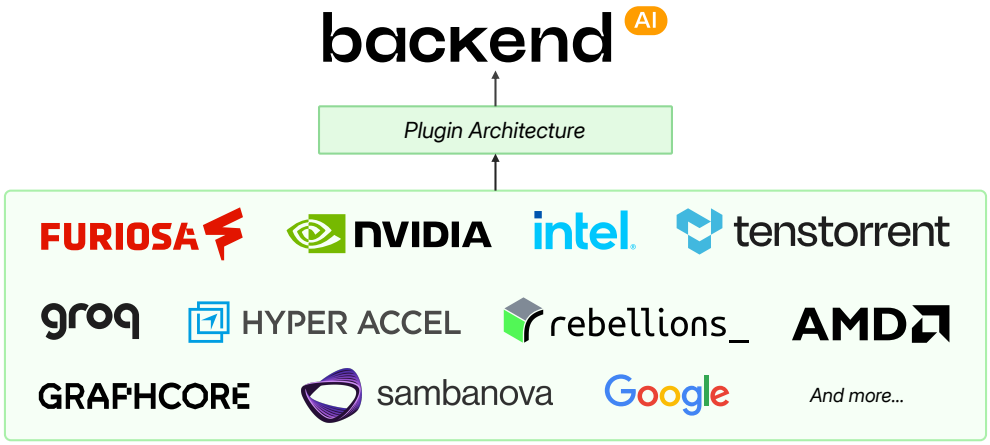


그림 4. 플러그인 구조로 지원하는 이기종 AI 가속기

Backend.AI는 이기종 가속기 지원을 위해 플러그인 구조를 채택하고 있습니다. 가속기 지원을 코어에 직접 추가하는 대신 플러그인으로 분리함으로써, 제품 릴리스 주기와 무관하게 새로운 가속기를 신속하게 지원할 수 있고, 기존에 운영 중인 가속기 환경에 영향을 주지 않으면서 새로운 가속기를 추가할 수 있습니다.

가속기 통합 제약을 줄이는 플러그인 구조 덕분에 Backend.AI는 FuriosaAI RNGD 이외에도 1세대 FuriosaAI Warboy Vision NPU도 지원하고 있으며, NVIDIA, Intel, AMD, Google, Graphcore, Tenstorrent, rebellions, HyperAccel 등 다양한 벤더의 AI 가속기를 지원하고 있습니다.

3.6 모델 서빙과 운영 기능

Inference 세션은 모델 서비스 엔드포인트 단위로 묶여 운영됩니다. 엔드포인트는 다수의 Inference 세션(레플리카)에 트래픽을 분산하고, API 토큰을 발급하며, 평균 응답 시간과 큐 길이 등 메트릭을 기반으로 레플리카 수를 자동 조정합니다. 모델 서비스 엔드포인트에는 Inference 세션에서 제공하는 HTTP API를 외부에 노출하는 기능이 제공되며, 이를 통해 Furiosa-LLM, vLLM, SGLang 등의 추론 라이브러리에서 제공하는 OpenAI 및 Anthropic 호환 API를 그대로 지원합니다. 이를 통해 폐쇄망 환경에서도 LLM 클라이언트 라이브러리를 활용하여 폐쇄망 LLM 서비스를 구축할 수 있으며, 애플리케이션에서는 엔드포인트 주소만 변경하는 방식으로 연동할 수 있습니다.

모델 버전 교체시에는 롤링 배포, 블루-그린 배포, 카나리 배포 세 가지 배포 전략을 사용할 수 있습니다.

배포 전략	설명
롤링 배포 (Rolling Deployment)	레플리카의 순차 교체
블루/그린 배포 (Blue/Green Deployment)	신버전과 구버전을 동시에 운영 후 일괄 전환
카나리 배포 (Canary Deployment)	일부 트래픽만 신버전으로 이관하여 사전 검증 수행

표 5. Backend.AI에서 사용가능한 배포 전략

2026년 7월 기준 현재는 롤링 배포 전략을 사용할 수 있으며, 블루/그린 배포와 카나리 배포 전략은 26년 내 지원 예정에 있습니다. 현재 지원중인 롤링 배포의 경우에는 롤아웃 비율과 같은 요소들을 자유롭게 조정 가능합니다.

3.6.1 VFolder

Backend.AI는 NFS, CephFS, VAST Data, WekaFS 등 다양한 스토리지 백엔드를 VFolder라는 추상 계층으로 통합하여 세션에 마운트합니다. 스토리지 쿼터는 사용자 및 프로젝트 단위로 관리됩니다. 보안 측면에서는 시스템 콜 화이트리스트 기반의 컨테이너 샌드박싱을 통해 사용자 코드를 격리하며, 사용자 컨테이너에 특권(privileged mode)을 부여하지 않는 구성을 기본값으로 적용합니다. 폐쇄망 환경에서도 전체 기능이 동일하게 동작합니다.

3.6.2 Backend.AI Agent 및 all-smi

Backend.AI Agent는 세션, 노드, 가속기 단위의 활용률, 메모리, 전력, 온도 등 서빙 운영에 필요한 메트릭을 수집합니다. 수집된 메트릭은 대시보드를 통해 테넌트별로 실시간 확인할 수 있습니다. 래블업이 개발한 오픈소스 메트릭 익스포터인 all-smi를 함께 사용하면, RNGD를 포함한 이기종 가속기 혼합 환경에서도 벤더에 관계없이 동일한 형식으로 메트릭을 통합 조회할 수 있습니다.

3.6.3 Continuum Router 및 Continuum Hub

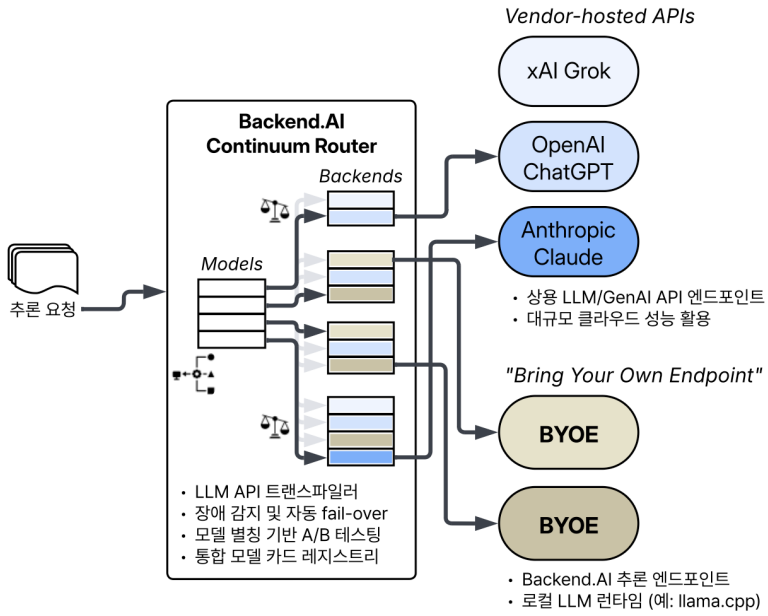


그림 5. Continuum Router 아키텍처

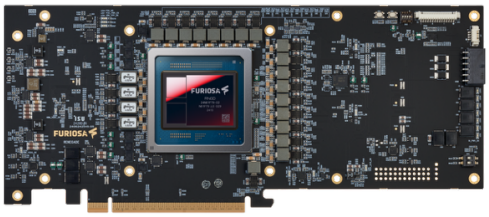
Continuum Router는 여러 서빙 엔드포인트와 추론 엔진 앞단에 위치하는 AI 라우팅 계층입니다. 수신된 요청을 적절한 대상으로 분배하며, 스마트 라우팅, 가드레일, 장애 대응 기능을 통합적으로 제공합니다. Continuum Router는 API 포맷 간 실시간 변환을 지원합니다. 사용자의 설정에 따라 오픈 모델과 자체 호스팅 모델 서비스를 OpenAI Responses API나 Anthropic Messages API 형식으로 노출할 수 있어, Claude Code, Codex 등 다양한 코딩 에이전트를 원하는 모델에 연결하여 사용할 수 있습니다.

Continuum Hub는 다수의 Continuum Router를 관제하는 상위 계층입니다. 사용자 요청이 어느 엔드포인트와 엔진을 거쳤는지, 어떤 방식으로 분산 처리되었는지를 추적하며, 부서 및 사용자별 토큰 사용량과 비용을 단일 관제 화면에서 확인하는 토큰 팩토리 체계를 구축할 수 있습니다. Continuum Router의 상세 기능은 다음 매뉴얼을 참조하시기 바랍니다.

4. FuriosaAI RNGD: 추론을 위해 설계된 가속기

RNGD(Renegade)는 대규모 언어 모델(LLM)과 멀티모달 모델의 추론을 위해 설계된 FuriosaAI의 2세대 AI 가속기로, 2026년 양산에 들어갔습니다. NPU(Neural Processing Unit)는 신경망 연산에 특화된 전용 가속기입니다. 범용 연산을 처리하는 CPU나 병렬 워크로드를 폭넓게 다루는 GPU와 달리, AI 모델 추론에 필요한 텐서 연산을 높은 전력 효율로 수행하는 것에 초점을 둡니다. RNGD는 TSMC 5nm 공정으로 제조되며, 48GB HBM3 메모리를 탑재하고 있습니다.

4.1 RNGD 종류 및 사양



아키텍처	Tensor Contraction Processor
프로세스 노드	TSMC 5nm
동작 클럭	1.0 GHz
BF16 성능	256 TFLOPS
FP8 성능	512 TFLOPS
INT8 성능	512 TOPS
INT4 성능	1,024 TOPS
메모리	HBM3 48 GB, 대역폭 1.5 TB/s
온칩 SRAM	256 MB
카드 전력	180 W
폼팩터	PCIe Gen5 x16, 패시브 쿨링 (공랭 방식)
보안	Secure Boot with Root of Trust, ECC Memory 지원

표 6. RNGD 사양

4.1.1 NXT RNGD Server

그림 6. FuriosaAI NXT RNGD Server

NXT RNGD Server는 RNGD 8장을 4U 섀시에 탑재한 데이터센터용 추론 서버입니다. 총 384GB HBM3 메모리와 12TB/s 수준의 합산 메모리 대역폭을 갖추며, FP8 기준 최대 4 PFLOPS의 연산 성능을 시스템 전력 약 3kW로 제공합니다. NXT RNGD Server는 공랭 냉각 방식으로 설계되어, 액체냉각 설비 없이 기존 상면에서 추론 클러스터를 구성할 수 있습니다. 이러한 낮은 전력 요구량 및 공조 환경을 감안한 설계를 통해 사용자는 NXT RNGD Server를 기존 데이터센터 환경 및 서버 설치 환경의 큰 변화 없이 도입할 수 있습니다.

4.2 Tensor Contraction Processor 아키텍처

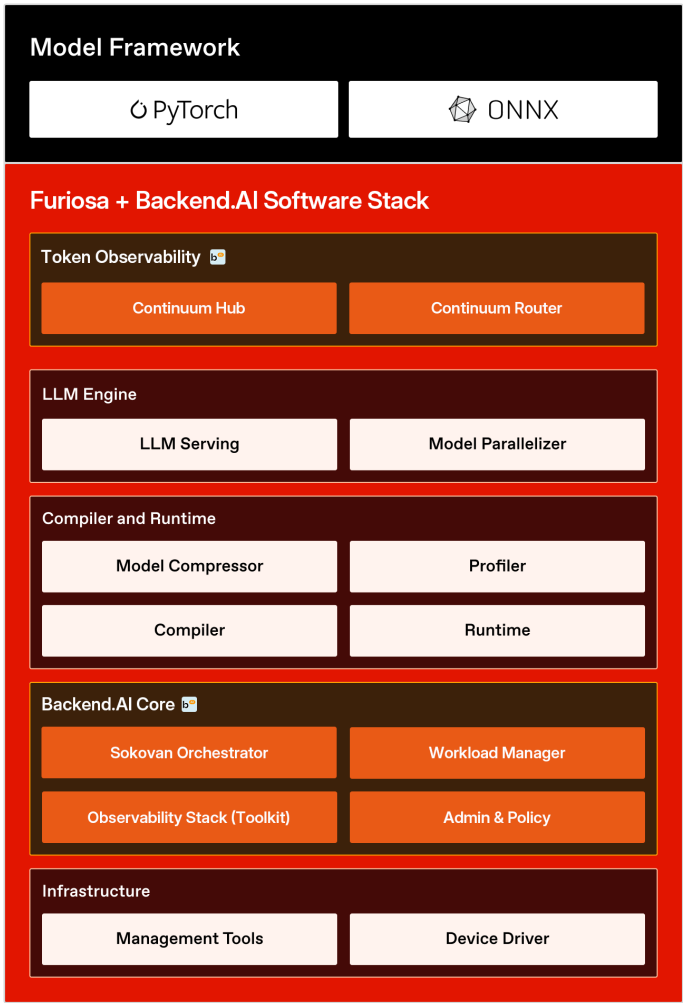
RNGD의 연산 코어는 FuriosaAI가 TCP(Tensor Contraction Processor)라고 부르는 독자 아키텍처입니다. 텐서 축약(Tensor Contraction)은 행렬 곱을 고차원 텐서로 확장한 연산을 의미하며, 트랜스포머 기반의 현대 LLM은 이러한 연산에 크게 의존합니다.

기존 가속기의 상당수는 고정 크기의 행렬 곱 유닛을 하드웨어에 구현하고, 컴파일러가 모델의 텐서 연산을 이 유닛 크기에 맞게 분할·배치합니다. 연산 형태와 유닛 크기가 정확히 맞아떨어지는 경우에는 효율적이지만, LLM 서빙처럼 배치 크기와 시퀀스 길이가 요청마다 달라지는 환경에서는 유닛을 채우지 못하는 조각 연산이 늘어나 하드웨어 활용률이 떨어집니다. TCP는 이 문제에 대응하기 위해 텐서 축약 연산 자체를 하드웨어의 기본 단위로 삼고, 컴파일러가 텐서의 분할과 데이터 이동을 명시적으로 계획하여 연산 유닛에 매핑하는 방식을 채택했습니다. 연산 형태가 변하더라도 매핑을 다시 계획하면 되므로, 가변적인 서빙 부하에서도 활용률을 유지하기 유리합니다.

4.3 메모리 서브시스템과 멀티카드 구성

RNGD의 메모리 서브시스템은 LLM 추론 워크로드에 맞춰 설계되었습니다. 칩 외부에는 HBM3 모듈 2개가 CoWoS-S 패키징으로 결합되어 48GB 용량과 1.5TB/s 대역폭을 제공합니다. 칩 내부에는 256MB SRAM이 장착되어 있어, 자주 재사용되는 텐서 조각을 온칩에 적재함으로써 외부 메모리 접근을 줄이고 연산 속도를 높일 수 있습니다. 카드당 48GB의 메모리 용량은 FP8로 양자화한 30B급 모델을 단일 카드에 적재할 수 있는 수준입니다. FuriosaAI의 서빙 프레임워크인 Furiosa-LLM은 최대 8장까지 텐서 병렬화를 지원하므로, 이보다 큰 모델은 여러 카드에 분산 적재할 수 있습니다. Solar Open, K-EXAONE 등 최신 주권 AI 모델도 이 구성으로 실행할 수 있습니다.

4.4 소프트웨어 스택: Furiosa SDK와 Furiosa-LLM



FuriosaAI는 RNGD를 다양한 환경에서 활용할 수 있도록 전용 소프트웨어 스택을 제공합니다. Furiosa SDK는 RNGD용 모델 배포와 서빙을 위한 통합 소프트웨어 스택으로, 이식성과 운영 효율을 함께 고려하여 설계되었습니다.

Furiosa Compiler

Furiosa Compiler는 모델 그래프를 최적화하고 NPU용 실행 바이너리를 생성하는 컴파일러입니다. 모델 아키텍처와 애플리케이션 요구 사항에 따라 하나의 모델을 여러 실행 파일로 컴파일할 수 있습니다. 기존 PyTorch 코드를 크게 수정하지 않고 RNGD용 실행 계획으로 변환할 수 있으며, Hugging Face Hub에 공개된 모델은 별도 변환 없이 바로 빌드할 수 있습니다.

그림 7. Furiosa와 Backend.AI가 결합된 소프트웨어 스택

Furiosa-LLM

Furiosa-LLM은 대규모언어모델(LLM) 및 멀티모달 모델을 위한 고성능 추론 엔진입니다. Furiosa-LLM은 vLLM 호환 API, 텍스트 및 멀티모달 서비스, 생성형 및 풀링 모델 지원, PagedAttention을 활용한 효율적인 KV캐시 관리, 공유 프롬프트 접두사 재사용을 위한 Radix-tree prefix caching, 다수 개의 NPU에서 데이터/텐서/파이프라인 병렬 처리, 다양한 디코딩 알고리즘, 가시성을 높이기 위한 Prometheus 메트릭 엔드포인트, Hugging Face 모델 및 허브 지원과의 통합 등을 지원합니다. Furiosa-LLM은 메모리 사용량, 추론 지연시간, 전력 소비를 줄일 수 있는 모델 양자화 (Quantization)를 기본으로 지원합니다. BF16(W16A16), FP8(W8A8), MXFP4, NVFP4를 포함한 다양한 학습 후 양자화(PTQ) 방식을 지원합니다.

Furiosa-LLM은 모델 지원을 위한 커널 프레임워크로 TCL(Tensor Contraction Language)을 사용합니다. TCL은 RNGD의 TCP(Tensor Contraction Processor) 아키텍처와 동일하게 텐서 축약(tensor contraction)을 기본 연산으로 삼는 선언형 언어로, 하드웨어 네이티브 연산을 고수준에서 직접 표현하면서 모델 구현 단계에서 다양한 최적화 힌트를 명시적으로 줄 수 있습니다. 또한 커널이 여러 모델이 공유하는 빌딩 블록으로 모듈화되므로, 새로운 모델을 기존 블록의 조합만으로 쉽고 효율적으로 포팅할 수 있습니다. Furiosa-LLM의 2026.2.0 릴리스 이후 추가된 모든 모델은 TCL을 기반으로 구현되어 있습니다.

* 2026년 하반기에 Furiosa-LLM과 함께 TCL을 공개, 사용자가 직접 모델을 추가할 수 있도록 지원할 예정입니다.

4.5 래블업의 MLxcel로 확장하는 RNGD의 모델 서빙 능력

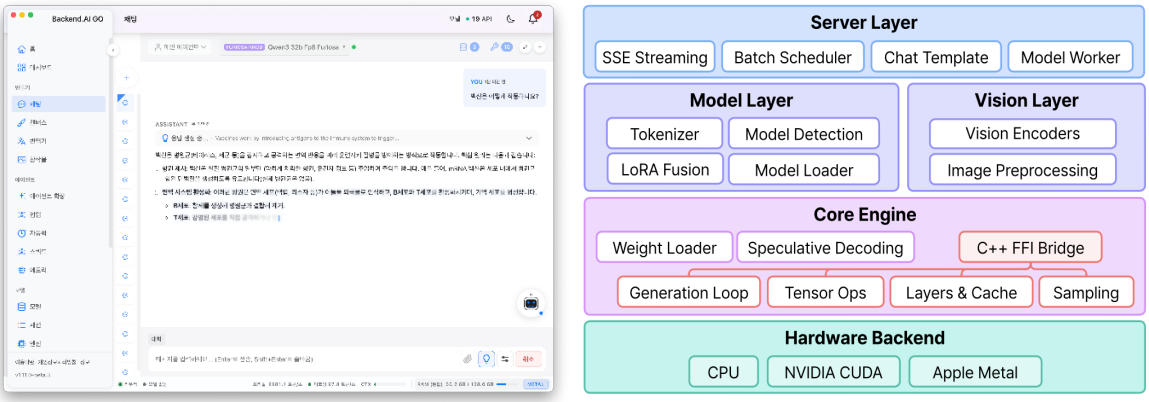


그림 8. RNGD를 통해 래블업의 AI:GO에서 구동되는 Qwen 32B FP8 모델 & MLxcel 추론 런타임 가속화 엔진 구조

래블업이 개발하는 추론 엔진 MLxcel(ML-accel로 발음)도 RNGD 지원을 준비하고 있습니다. MLxcel은 LLM, VLM, dLLM, STT 등 다양한 모델의 추론, 배치 스케줄링, KV 캐시 관리, 멀티노드 대규모 모델 서빙을 OpenAI 호환 API로 제공하는 통합 추론 엔진입니다. 하드웨어별 연산 계층을 분리하는 구조를 통해 OpenXLA-RNGD 지원을 추가하며, mlxcel이 지원하는 모델 생태계를 RNGD에서도 활용할 수 있게 됩니다.

* MLxcel의 RNGD 지원은 2026년 3분기 예정입니다.

5. 통합 아키텍처: Backend.AI에서 RNGD 운영하기

5.1 생애 주기 관리

학습과 추론에 요구되는 하드웨어 특성이 다르고, 추론 단계에서도 Prefill과 Decode를 나눠 운용하는 등 가속기 풀을 혼용 운영하는 환경이 늘어나고 있습니다. Backend.AI는 자원 그룹(Resource Group) 개념을 통해 이러한 환경을 일관된 방식으로 관리합니다. 운영자는 자원 그룹에 노드를 등록하고, 이후 하드웨어 자원부터 세션의 생성과 종료까지 전체 생애주기를 Backend.AI에서 관리할 수 있습니다.

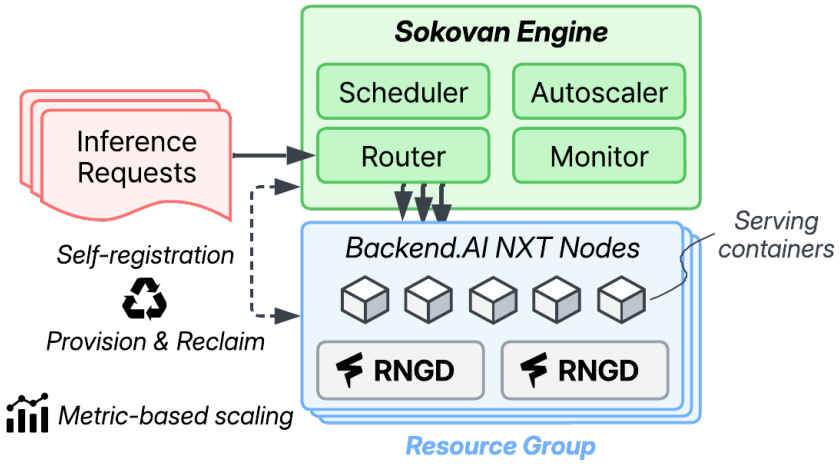


그림 9. Backend.AI Sokovan에서의 리소스 라이프사이클

- **자원 등록:** 각 NXT RNGD Server 노드에서는 Backend.AI 에이전트가 FuriosaAI 플러그인을 통해 RNGD 디바이스를 인식합니다. 인식된 장치는 NPU 슬롯 형태로 매니저에 등록되며, 이후 대시보드와 스케줄러에서 다른 가속기 자원과 동일하게 취급됩니다.

- **세션 요청:** 사용자는 필요한 가속기 수와 CPU, 메모리, 실행 이미지를 지정해 세션을 생성합니다. 이 요청은 하나의 실행 단위로 스케줄러에 전달됩니다.

- **스케줄링:** Sokovan은 테넌트 쿼터와 현재 자원 상태를 기준으로 배치 가능한 노드를 선택합니다. 이 과정에서 NUMA 배치도 함께 고려됩니다. 자원이 확정되면 세션은 컨테이너로 생성되며, 할당된 RNGD 장치만 노출됩니다. 동일 노드의 다른 가속기는 별도 세션에 독립적으로 배정됩니다.

- **서빙:** 컨테이너 내부에서 Furiosa-LLM이 모델을 로드하고 텐서 병렬 방식으로 추론을 수행합니다. Backend.AI는 해당 세션에 접근 가능한 엔드포인트를 생성하며, 외부 요청은 이 엔드포인트를 통해 세션으로 전달됩니다. 트래픽 분산과 오토스케일링은 엔드포인트 설정에 따라 구성됩니다.

- **운영, 모니터링 및 회수:** 실행 중인 세션의 자원 사용량과 전력 지표는 Backend.AI를 통해 수집되며, 테넌트 단위로 집계됩니다. 세션이 종료되면 할당된 자원은 즉시 반환되어 다른 요청에 재사용됩니다.

5.2 호스트 구성: 성능을 좌우하는 준비 작업

가속기의 성능을 안정적으로 확보하려면 호스트 수준의 설정도 함께 검토해야 합니다. 호스트 구성에 따라 동일한 가속기에서도 실측 성능이 달라질 수 있으므로, 다음과 같은 설정을 운영 환경에 맞춰 적용하는 것을 권장합니다. 해당 내용은 추후 업데이트를 통해 변동될 수 있으므로, 최신의 정보를 위해 다음 퓨리오사 문서를 참조하세요.

설정 항목	내용	기대 효과
PCIe ACS 비활성화	카드 간 P2P 통신이 루트 컴플렉스를 거치지 않도록 구성	다중 가속기 텐서 병렬 환경 통신 지연 감소
hugepages 활성화	대용량메모리 매핑 시 페이지 관리 오버헤드 감소	메모리 접근이 빈번한 추론 워크로드에서의 성능 안정화
NUMA 정렬	Backend.AI의 NUMA-aware 배치를 통해 CPU 코어를 가속기와 동일한 NUMA 노드에 할당	메모리 접근 지연 감소

표 7. 효율적인 운영을 위한 호스트 설정 항목

6. 벤치마크: Backend.AI 위에서 RNGD 운영하기

6.1 벤치마크 개요 및 측정 환경

가속기 종류	FuriosaAI RNGD 4장 (NXT RNGD Server)	NVIDIA RTX PRO 6000 Blackwell Server Edition 4장
모델 종류	furiosa-ai/qwen3-32b-fp8 (Furiosa 최적화)	Qwen/Qwen3-32B-FP8
서빙 엔진	Furiosa-LLM 2026.3.0, 텐서 병렬 4	vLLM 0.22.0 (26.05.29 릴리즈), 텐서 병렬 4
플랫폼	Backend.AI 26.4.6	베어메탈
워크로드	입력 1,024 토큰 / 출력 1,024 토큰 / 동시 요청 1~256	

표 8. 벤치마크 측정 환경 및 워크로드

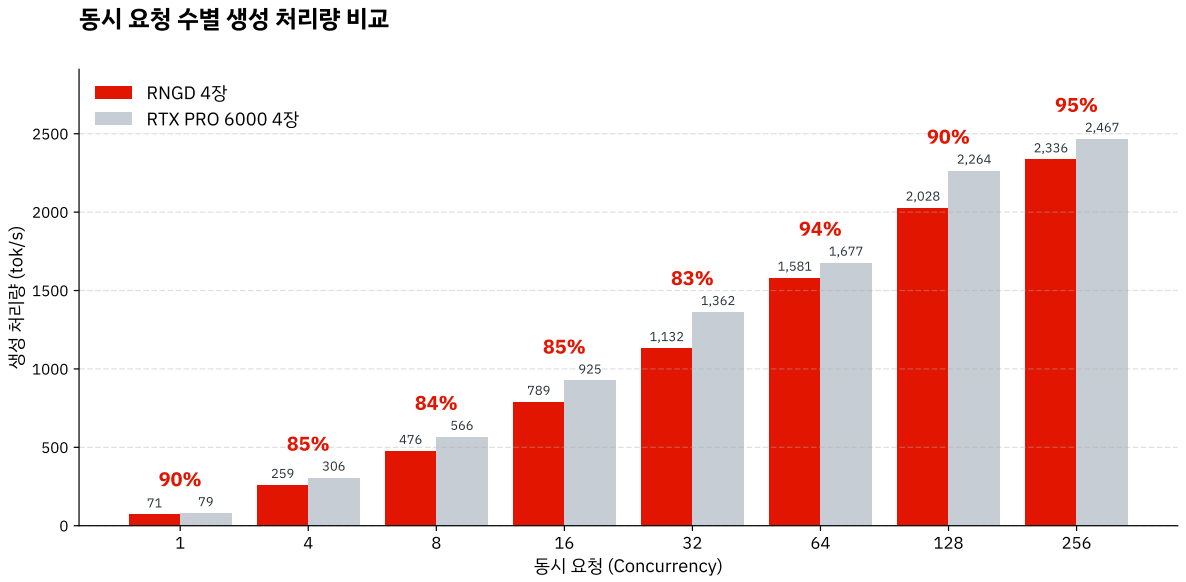
벤치마크를 위해 각각 4장의 FuriosaAI RNGD AI 가속기와 NVIDIA RTX PRO 6000 Blackwell AI 가속기가 사용되었으며, 실제 서비스 환경에서 널리 사용되는 오픈모델인 Qwen3의 32B 크기 모델이 사용되었습니다. 두 시스템 모두 동일한 모델을 FP8의 정밀도로 서빙, 텐서 병렬 4의 조건으로 측정되었으며, Radix-tree prefix caching을 활성화할 경우 프롬프트 접두사 공유율에 따라 결과가 달라질 수 있어, 가속기의 기준 성능을 비교하기 위해 Prefix Caching은 양쪽 모두 비활성 조건으로 측정하였습니다. 벤치마크는 2026년 6월에 측정되었습니다.

지표	정의	의미
처리량 (tok/s)	초당 토큰 생성 수	시스템 전체 처리능력
TTFT (Time-To-First-Token)	요청 이후 첫 토큰 생성까지의 지연시간	사용자 체감 응답성
TPOT (Time-Per-Output-Token)	첫 토큰 이후 토큰 간 평균 생성 간격	지속적 생성 속도

표 9. 성능평가 지표

성능 평가에 사용된 지표는 다음과 같습니다. 처리량은 초당 생성 토큰 수(tok/s)로 시스템 전체의 처리 능력을 나타냅니다. TTFT(Time To First Token)는 요청 이후 첫 토큰이 생성되기까지의 지연 시간을 의미하며, 사용자 체감 응답성과 직결되는 지표입니다. TPOT(Time Per Output Token)는 첫 토큰 이후 생성되는 토큰 간의 평균 간격으로, 지속적인 생성 속도를 평가하는 데 사용됩니다. 전력은 가속기 카드 4장의 합산 실측값 기준으로 측정했으며, 실제 시스템 단위의 총 전력 차이는 이보다 더 크게 나타날 수 있습니다.

6.2 동등 조건 구성에서 비교 환경에 근접한 처리량



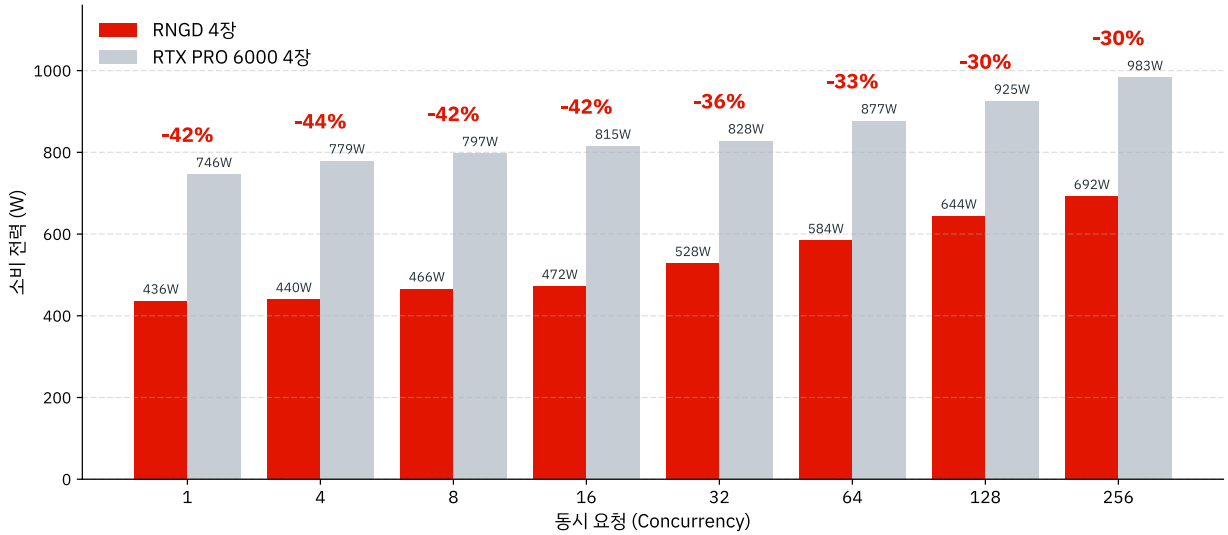
동시 요청 (Concurrency)	RNGD 4장	RTX PRO 6000 4장	비율
1	71 tok/s	79 tok/s	90%
4	259 tok/s	306 tok/s	85%
8	476 tok/s	566 tok/s	84%
16	789 tok/s	925 tok/s	85%
32	1,132 tok/s	1,362 tok/s	83%
64	1,581 tok/s	1,677 tok/s	94%
128	2,028 tok/s	2,264 tok/s	90%
256	2,336 tok/s	2,467 tok/s	95%

표 10. 동시 요청 수별 생성 처리량 비교

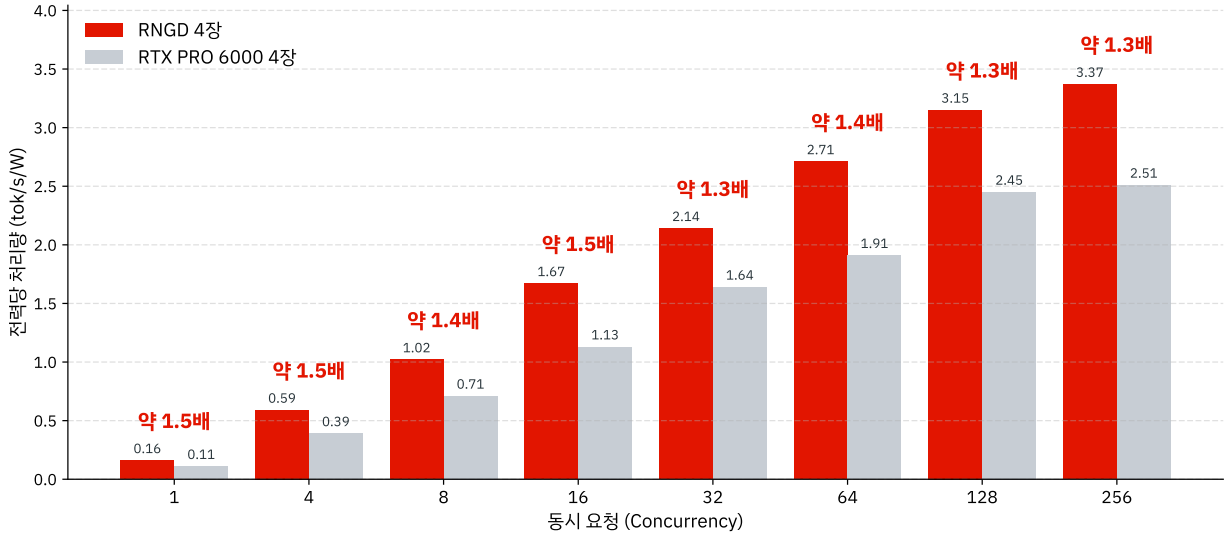
동시 요청 256 기준으로 RNGD 4장은 2,336 tok/s를 기록하여 RTX PRO 6000 4장(2,467 tok/s)의 약 95%에 도달했습니다. 다른 동시성 구간에서도 83~95% 범위를 유지합니다. 구간별 추이를 보면, 동시 요청 1에서 90%로 시작하여 중간 부하 구간(8~32)에서 83~85%로 차이가 벌어졌다가, 64 이상의 고부하 구간에서 90~95%로 다시 좁혀집니다. 동시 요청 1에서 256으로 올리는 동안 RNGD의 처리량은 32.8배, RTX PRO 6000은 31.2배 증가하여 양쪽 모두 부하에 비례해 확장됩니다. 최고 부하 구간(128→256)의 증가율은 RNGD가 15.2%, RTX PRO 6000이 9.0%로, 고부하 구간에서 RNGD가 상대적으로 여유를 보이는 것을 확인할 수 있습니다.

6.3 전력당 처리량 약 1.3~1.5배의 전력 효율

동시 요청 수별 소비 전력 비교



동시 요청 수별 전력당 처리량 비교



동시 요청 (Concurrency)	전력 (RNGD / RTX PRO 6000)	전력당 처리량 (RNGD / RTX PRO 6000)	배수
1	436W / 746W	0.16 / 0.11 tok/s/W	약 1.5배
4	440W / 779W	0.59 / 0.39 tok/s/W	약 1.5배
8	466W / 797W	1.02 / 0.71 tok/s/W	약 1.4배
16	472W / 815W	1.67 / 1.13 tok/s/W	약 1.5배
32	528W / 828W	2.14 / 1.64 tok/s/W	약 1.3배
64	584W / 877W	2.71 / 1.91 tok/s/W	약 1.4배
128	644W / 925W	3.15 / 2.45 tok/s/W	약 1.3배
256	692W / 983W	3.37 / 2.51 tok/s/W	약 1.3배

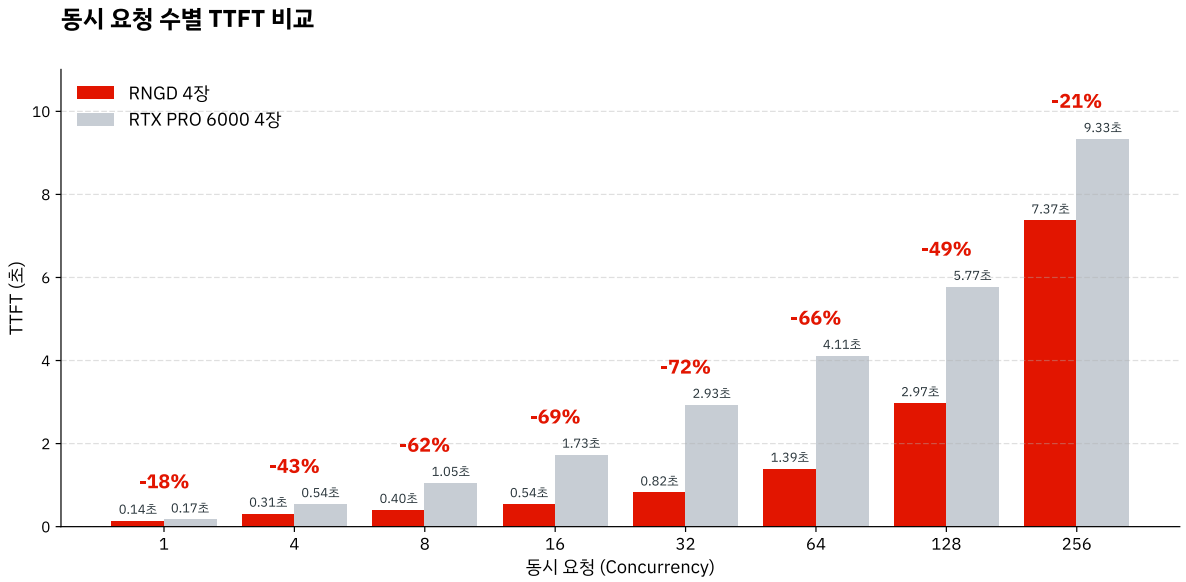
표 11. 동시 요청 수별 전력량 비교

RNGD 4장의 합산 전력은 모든 구간에서 RTX PRO 6000 4장 대비 56~70% 수준으로 측정 되었습니다. 전력당 처리량으로 환산하면, 모든 동시성 구간에서 RNGD가 약 1.3~1.5배 높은 효율을 보입니다. 처리량 격차는 5~17%인 반면 전력 격차는 30~43%로 더 크기 때문에, 전력 효율 우위가 나타나는 구조입니다.

부하 증가에 따른 전력 변화도 주목할 만합니다. 동시 요청 1에서 256으로 증가하는 동안 RNGD의 합산 전력은 436W에서 692W로 상승했으며, 카드당 기준으로는 109~173W 범위입니다. 최대 부하에서도 카드당 소비 전력이 TDP 180W 이내에 머물러, 전력 계획 시 TDP를 상한으로 가정해도 실제 사용량이 이를 초과할 가능성은 낮습니다. RTX PRO 6000의 경우 합산 전력이 746W에서 983W까지 증가했으며, 카드당 187~246W로 나타났습니다.

전력 차이는 특히 저부하 구간에서 두드러집니다. 동시 요청 1~16 구간에서 RNGD는 436~472W를 유지한 반면, RTX PRO 6000은 746~815W를 기록했습니다. 이러한 특성은 트래픽 변동이 크고 저부하 상태가 오래 유지되는 서비스에서 누적 전력 비용의 차이로 이어집니다. 고가용성 등을 고려하여 여러 서버에 걸쳐서 상시 가동되는 추론 서비스는 평균 부하가 낮은 구간에 머무르는 시간이 긴 경우가 많으므로, 저부하 구간의 전력 효율도 중요한 고려 요소입니다.

6.4 빠르게 반환하는 첫 토큰 출력 (TTFT)



동시 요청 (Concurrency)	TTFT (RNGD / RTX PRO 6000)	TPOT (RNGD / RTX PRO 6000)
1	0.14초 / 0.17초	13.9ms / 12.5ms
4	0.31초 / 0.54초	15.1ms / 12.5ms
8	0.40초 / 1.05초	16.4ms / 12.9ms
16	0.54초 / 1.73초	19.7ms / 15.9ms
32	0.82초 / 2.93초	27.4ms / 20.7ms
64	1.39초 / 4.11초	38.9ms / 33.7ms
128	2.97초 / 5.77초	59.3ms / 52.1ms
256	7.37초 / 9.33초	99.2ms / 99.1ms

표 12. TTFT & TPOT 비교

TTFT: 모든 동시성 구간에서 RNGD가 더 짧은 TTFT를 기록했습니다. RTX PRO 6000은 동시 요청 8에서 1초를 넘고 32에서 약 3초까지 증가하는 반면, RNGD는 동시 요청 32까지 1초 이내를 유지합니다. 동시 요청 8~64 구간에서는 약 2배 수준의 차이가 나타납니다. 다수의 요청이 동시에 처리되는 운영 환경에서 TTFT는 사용자가 가장 직접적으로 체감하는 지표이므로, 이 구간의 차이는 서비스 품질에 의미 있는 영향을 줄 수 있습니다.

TPOT: 낮은 동시성 구간에서는 RTX PRO 6000이 더 짧은 TPOT를 보이며, 이 경향은 동시 요청 128까지 유지됩니다. 동시 요청 256에서는 두 시스템 모두 약 99ms로 수렴합니다. 다만, RTX PRO 6000이 우위를 보이는

저동시성 구간에서는 양쪽 모두 초당 50~80토큰 수준의 생성 속도를 보이므로, 스트리밍 기반 사용자 경험에서의 체감 차이는 제한적입니다.

7. 활용 시나리오와 도입 가이드

7.1 워크로드별 시나리오

벤치마크 결과를 토대로 RNGD를 실제로 사용할 수 있는 환경을 정리해보면 다음과 같습니다.

시나리오 1: 사내 LLM 어시스턴트·챗봇 서비스

다수 사용자가 동시에 사용하는 사내 어시스턴트 환경에서는 응답 시작 지연(TTFT)과 스트리밍 안정성이 주요 평가 지표가 될 수 있습니다. 벤치마크 기준으로 동시성 8~64 구간의 값을 샘플링하여 확인할 수 있으며, RNGD는 해당 구간에서 0.40~1.39초 수준의 TTFT를 기록한 반면 비교 솔루션의 경우 1.05~4.11초 범위를 보여줍니다.

시나리오 2: 대량 문서 처리·배치 추론

문서 요약, 분류, 정보 추출과 같은 배치성 워크로드는 처리량이 우선되는 경우가 많기 때문에 높은 처리량을 가진 가속기를 사용하는 것이 유리합니다. RNGD는 벤치마크 기준 동시 요청 128~256 구간에서 비교 솔루션 대비 약 70% 수준의 전력으로 약 90~95% 수준의 처리량을 보였습니다. 따라서 추론 인프라를 RNGD로 구성하면 동일 전력 예산 기준으로 처리 가능한 작업량을 늘릴 수 있습니다. 배치 처리 특성상 개별 요청 지연보다는 전체 처리 시간이 중요하므로 TPOT 차이는 주요 제약으로 작용하지 않는 경우가 많습니다. 고동시성 구간에서의 스케일링 특성은 배치 처리 시간 단축에 직접적인 영향을 줄 수 있습니다.

Backend.AI에서는 Batch 세션과 Inference 세션을 동일 자원 그룹에 배치할 수 있어, 시간대에 따라 자원을 사용하는 운영을 통해 워크로드 운영 효율을 끌어올릴 수 있습니다.

시나리오 3: 기존 운영중인 GPU 클러스터에 추론 계층 추가

기존에 도입한 GPU 장비 클러스터를 Backend.AI로 운영 중인 환경에서는 별도의 플랫폼 분리 없이 자원 그룹을 추가하는 방식으로 RNGD를 도입할 수 있습니다. 학습 및 파인튜닝 워크로드는 기존 GPU 자원 그룹에 유지하고, 추론 서비스만 별도의 RNGD 자원 그룹으로 분리하는 구성이 일반적입니다. 이 경우 학습용 GPU가 상시 서빙에 점유되는 상황을 줄일 수 있으며, 자원 활용 측면에서 분리 효과가 나타납니다. 운영 측면에서는 동일한 대시보드, 계정 체계, 사용량 계량 방식을 그대로 유지할 수 있고, 사용자 입장에서는 세션 생성 시 선택 가능한 자원 유형이 추가되는 수준의 변화만으로 운영 효율을 개선할 수 있습니다.

7.2 도입을 권장하는 환경

- 전력·냉각 제약 안에서 LLM 추론 용량을 늘려야 하는 데이터센터

- 기존 보유 GPU와 병행 운영하며 추론 워크로드를 분리 배치하려는 조직

- 국산 AI 반도체 도입을 검토하는 공공·엔터프라이즈

- 주권 AI 요건에 따라 폐쇄망·온프레미스에서 LLM 서비스를 구축하려는 조직

8. 결론 및 권고

Backend.AI는 이기종 가속기를 포함한 AI 인프라를 하나의 운영 체계 안에서 다루기 위해 설계된 플랫폼입니다. 멀티노드 환경에서 자원을 유연하게 배치할 수 있고, 테넌트 단위로 격리된 실행 환경을 제공합니다. 장애 발생 시에도 세션 단위로 복구할 수 있는 구조를 갖추고 있어, 운영의 일관성을 유지하면서 확장을 이어갈 수 있습니다.

RNGD는 이러한 운영 구조 위에서 성능과 전력 효율이라는 이점을 제공하는 추론 전용 가속기입니다. 본 백서의 벤치마크에서 확인된 바와 같이, 동시성 구간 전반에서 전력당 처리량이 약 1.3~1.5배 높게 나타났으며, 모든 구간에서 더 짧은 TTFT를 기록했습니다. 특히 상시 가동되는 추론 서비스에서는 전력 효율의 차이가 누적 운영 비용에 직접적으로 반영됩니다. Backend.AI를 활용하면, RNGD AI 가속기를 기존 GPU 자원과 동일한 방식으로 배치하고 관리할 수 있습니다.

이 조합의 핵심 이점은 기존 인프라를 처음부터 재구축할 필요가 없다는 점입니다. 운영 중인 환경에 가속기 유형만 추가하여 운영 범위를 넓힐 수 있고, 자원 그룹 단위의 분리를 통해 학습과 추론 워크로드를 독립적으로 운영할 수 있으며, 클러스터 전체 관점에서는 자원 활용률을 재조정하는 효과도 기대할 수 있습니다. 아울러, 국내에서 설계된 가속기와 국내에서 개발된 플랫폼으로 구성된다는 점은 데이터 주권, 공급망 자율성, 운영 통제권 확보가 요구되는 환경에서 실질적인 선택지가 됩니다.

Backend.AI와 RNGD로 구성한 추론 인프라를 직접 확인하려면 래블업과 퓨리오사에이아이에 문의하세요.

9. 회사 소개

래블업 (Lablup Inc.)

래블업은 AI 인프라 운영 플랫폼 Backend.AI를 개발하는 소프트웨어 기업입니다. 2015년 설립 이래 GPU 가상화, 이기종 가속기 통합 관리, 멀티테넌트 자원 운영 등 엔터프라이즈 AI 인프라의 핵심 문제를 풀어 왔습니다. Backend.AI는 온프레미스, 클라우드, 폐쇄망 환경에서 동일하게 동작하며, FuriosaAI를 포함한 12개 이상의 AI 가속기 제조사의 하드웨어를 단일 플랫폼에서 운영할 수 있습니다. 자세한 내용은 https://www.lablup.com 에서 확인하실 수 있습니다.

퓨리오사에이아이 (FuriosaAI)

퓨리오사AI(FuriosaAI)는 데이터센터와 엔터프라이즈를 위한 차세대 AI 반도체를 개발하는 기업입니다. 보다 지속가능한 AI 컴퓨팅을 구현함으로써, 더 많은 사람들이 강력한 AI 기술의 혜택을 누릴 수 있도록 하는 것을 목표로 합니다. 자세한 내용은 http://furiosa.ai/ 에서 확인하실 수 있습니다.