Програмна реалізація виконана на Python із залученням бібліотек scikit-fuzzy для роботи з нечіткою логікою, NumPy і SciPy для математичних розрахунків, Matplotlib для візуалізації та Tkinter для створення графічного інтерфейсу. Математична модель теплопровідності базується на одновимірному нестаціонарному рівнянні Фур'є з граничними умовами третього роду, яке розв'язується чисельно методом Рунге-Кутти четвертого порядку з інтервалом інтегрування 0,1 секунди.

Підхід до формування бази знань передбачає розбиття експертної інформації операторів на прості правила прийняття рішень. Для трьох вхідних змінних із 3 – 4 термами кожна теоретично можливе формування бази з 27 – 48 правил. Проте на практиці використовується лише підмножина найважливіших комбінацій. Наприклад, ситуація типу «дуже висока температура + висока швидкість + низьке охолодження» класифікується як критична і потребує максимального керуючого впливу (+ 50%). Водночас конфігурація «низька температура + низька швидкість + високе охолодження» вимагає значного зниження впливу (– 50%), щоб уникнути термічного шоку металу.

Проведені випробування на реальних виробничих даних засвідчили високу результативність запропонованого підходу. Аналіз показав, що рекомендації, сформовані нечіткою логічною системою, майже повністю збігаються з рішеннями, які зазвичай приймають досвідчені оператори. Значення коефіцієнта кореляції 0,91 підтверджує тісний взаємозв’язок між результатами системи та експертними оцінками, що свідчить про адекватне відтворення логіки фахівців у моделі. Один повний цикл обчислень виконується за приблизно 0,15 секунди на типовому персональному комп’ютері, що значно швидше порівняно з класичним методом скінченних елементів. Важливо, що система демонструє надійну роботу навіть за змін умов чи параметрів процесу, забезпечуючи стабільні та коректні рекомендації без збоїв.

Метод демонструє високу адаптивність і зручність масштабування. Щоб додати нові параметри, наприклад марку сталі, товщину виробу або якість мастильного матеріалу, достатньо розширити базу правил, не змінюючи принципи роботи системи. Вона здатна підлаштовуватися під реальні умови експлуатації, автоматично коригуючи параметри функцій належності на основі даних про роботу обладнання.

Практичні результати впровадження цього підходу підтверджують доцільність використання нечіткої логіки для розв’язання завдань інтелектуального керування у металургійній сфері. Поєднання досвіду фахівців із потужністю обчислювальних алгоритмів створює ефективну альтернативу класичним математичним моделям і підвищує якість прийняття технологічних рішень. Також підхід характеризується стійкістю до збоїв окремих сенсорів або впливу шумів у вимірювальних каналах. Завдяки використанню нечітких функцій належності система може працювати навіть у разі отримання неповних або неточних даних, аналізуючи ситуацію за ступенем приналежності до встановлених термів. У промислових умовах, де вплив навколишніх факторів (температура, вібрація, пил) ускладнює точність вимірювань, такі властивості надають значні переваги порівняно з класичними алгоритмами керування.

Подальший розвиток методу може враховувати інтеграцію модулів прогнозування динаміки температурного поля за допомогою методів машинного навчання. Такий підхід дасть змогу реалізувати не тільки реактивні, а й проактивні управлінські дії: наприклад, попереднє регулювання інтенсивності охолодження у відповідь на прогнозовані зміни швидкості розливання чи температури металу. Комбінація таких рішень на основі нечітко-логічного підходу відкриває шлях до створення повноцінних систем «розумного лиття», здатних до автоматичної адаптації в умовах реального часу.

Список джерел

1. Mendel J.M. Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions. 2nd Edition. Springer International Publishing, 2018. 684 p. URL: https://doi.org/10.1007/978-3-319-51370-6 (дата звернення: 20.10.2025)

2. Karaboga D., Kaya E. Adaptive network based fuzzy inference system (ANFIS) training approaches: a comprehensive survey. *Artificial Intelligence Review*. 2018. Vol. 52, no. 4. P. 2263–2293. URL: https://doi.org/10.1007/s10462-017-9610-2 (дата звернення: 30.10.2025).

Масово паралельна реалізація MorphoNAS / Massively Parallel MorphoNAS Implementation
Медвідь С.О. / Medvid S.

Національний університет “Києво-Могилянська Академія” / National University of Kyiv-Mohyla Academy
04655, Київ, вул. Григорія Сковороди, 2, факультет інформатики, кафедра інформатики
E-mail: s.medvid@ukma.edu.ua

Our GPU-accelerated MorphoNAS (Morphogenetic Neural Architecture Search) enhances the biologically motivated developmental model of the original MorphoNAS, from sequential developmental simulations to large scale parallel simulation. Like the original CPU-based MorphoNAS system, the GPU enhanced version models morphogenesis (neural development) through reaction-diffusion, progenitor differentiation and axon-guided wiring that grow recurrent controllers. Unlike the original CPU-based version, the GPU-based version has the ability to formulate growth and control in parallel execution of thousands of genomes. Through use of sparse, device-resident representations of developmental dynamics and recurrent rollouts, we produce behaviorally equivalent recurrent controllers with up to three orders of magnitude speedup. As a result, we have enabled the possibility of performing evolutionary search over populations of developmental programs, transforming MorphoNAS from a proof-of-concept model of artificial morphogenesis and adaptive architecture discovery to a scalable framework for those objectives.

Background

MorphoNAS [1] is a biologically inspired approach to neural architecture search that views the development of recurrent neural networks as a process of morphogenesis [2].

In its original form, MorphoNAS executes **development** sequentially by first simulating morphogen secretion, diffusion, inhibition, and cell fate transitions from a single progenitor cell to a recurrent neural network. Because of this sequential nature of the process, it limits the size of the search space that can be explored. It exhibits both control-flow irregularity and memory access irregularity [3].

MorphoNAS also defines how the resulting networks will behave as **controllers**. The original MorphoNAS implementation performs the control loop sequentially by evaluating the behavior of each developed network individually and executing them in parallel using multiple CPU cores. However, this approach scales poorly. The cost of the rollouts increases quadratically with the number of neurons, and must be performed for each of the controller-environment pairs.

MorphoNAS GPU Implementation

To construct the **developmental** component of MorphoNAS, we preserve the morphogenetic rules used in the original MorphoNAS implementation and reformulate them to enable batched, massively parallel execution: diffusion and secretion are implemented as convolutional CUDA kernels; genome parameters are broadcast across the developmental grid of each network being grown; and cell division, differentiation, and synaptogenesis are performed for thousands of networks at a time.

For the **controller** component of MorphoNAS, we preserve the exact same control laws as the original MorphoNAS implementation: here, we represent them in a sparse CSR-like format representation (i.e., row pointers, edge indices, weights); then concatenate multiple such sparse representations of controllers, along with additional metadata describing their properties (neuron range, edge range, input neuron indices, output neuron indices, and internal settling depth) into large device buffers. We then execute the rollout portion of the MorphoNAS control loop using a single CUDA kernel that evaluates all active controllers in parallel. Controllers’ observations are injected into the appropriate input neurons; the kernel then iteratively updates neurons of each controller using on-GPU ping-pong buffers for state; and finally, the kernel collects activity of output neurons of each controller as an action. Temporal memory of controllers is preserved throughout the entire loop, without incurring the overhead associated with moving the hidden state between host and device [4].

Functionally, "Evaluate one dense recurrent controller sequentially" becomes "Evaluate a population of sparse recurrent controllers simultaneously," allowing us to execute the rollout portion of the MorphoNAS control loop in parallel over the entire population of controllers. Therefore, we can evaluate hundreds or thousands of different controllers simultaneously, and apply selection based on their performance on the task at hand.

Performance and Scaling

Developmental growth is now population-scale. GPU-driven growth reaches 9,710 networks/s on RTX 5090 and 5,379 networks/s on RTX 3090, versus 15-20 networks/s on high-core-count CPUs (12-core Apple M2 Max: 15.2 networks/s; 56-core Xeon: 14.4 networks/s; 192-core AMD EPYC: 19.9 networks/s). Poor CPU speedup is attributed to limited multithreading scalability in the BLAS/MKL implementations. Apple’s Accelerate demonstrated the highest parallel efficiency which confirms

findings in [5]. GPU implementation results in ~354x-639x speedup over CPU-based growth, even against 192-core servers.

Recurrent controller rollout accelerates 1-2 orders of magnitude depending on the networks diversity. For the real evolutionary runs with highly diverse networks, GPU recurrent rollout reaches ~0.5-1.1 million propagations per second depending on GPU and network class. This represents roughly 6-10x speedup over a parallel 12-core Apple M2 Max CPU baseline (113,837 propagations/s). Importantly, the GPU implementation preserves the exact recurrent control loop used on CPU.

The bottleneck moves from development to behavior. With GPU acceleration, developmental growth becomes effectively free compared to other parts. Growth on RTX 5090 proceeds at 9,710 networks/s, whereas even the best CPU setup reaches only 19.9 networks/s for the same genomes.

Implications. GPU implementation is able to simulate growth of thousands of networks in parallel and apply parallel evaluation of thousands controllers at a time. It has eliminated morphogenesis as a computational bottleneck in MorphoNAS and significantly widened target search space. The combination of biological realism with large-scale parallelism has potential to transform MorphoNAS into a scalable platform for discovering adaptive, self-organizing neural architectures.

Sources

[1] M. Glybovets and S. Medvid, "MorphoNAS: Embryogenic Neural Architecture Search Through Morphogen-Guided Development," *arXiv preprint arXiv:2507.13785*, 2025, [Online]. Available: https://arxiv.org/abs/2507.13785

[2] A. M. Turing, "The Chemical Basis of Morphogenesis," *Bulletin of Mathematical Biology*, Jan. 1990, doi: 10.1016/S0092-8240(05)80008-4.

[3] M. Burtscher, R. Nasre, and K. Pingali, "A quantitative study of irregular programs on GPUs," in *2012 IEEE International Symposium on Workload Characterization (IISWC)*, Nov. 2012, pp. 141–151. doi: 10.1109/IISWC.2012.6402918.

[4] G. Diamos *et al.*, "Persistent RNNs: Stashing Recurrent Weights On-Chip," in *Proceedings of The 33rd International Conference on Machine Learning*, PMLR, June 2016, pp. 2024–2033. Accessed: Oct. 31, 2025. [Online]. Available: https://proceedings.mlr.press/v48/diamos16.html

[5] P. Hübner, A. Hu, I. Peng, and S. Markidis, "Apple vs. Oranges: Evaluating the Apple Silicon M-Series SoCs for HPC Performance and Efficiency," in *2025 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, June 2025, pp. 45–54. doi: 10.1109/IPDPSW66978.2025.00013.

АДАПТИВНА МУЛЬТИМОДАЛЬНА КОГНІТИВНА СИСТЕМА ДЛЯ АВТОНОМНОГО ПРИЙНЯТТЯ РІШЕНЬ У СКЛАДНИХ ЛОГІСТИЧНИХ СЕРЕДОВИЩАХ

Пасєка П.І., Теренчук С. А.

Київський національний університет будівництва і архітектури

03037, Київ, проспект Повітряних Сил, 31, факультет автоматизації і інформаційних систем, кафедра інформаційних технологій проєктування та прикладної математики

E-mail: paaaa.paul@gmail.com, terenchuksa@ukr.net

This research focuses on a conceptual framework for an adaptive multimodal cognitive system designed to optimize logistics processes in dynamic and uncertain environments. Proposed architecture for a continually updated world-model (knowledge graph) from fused sensory inputs that employs interacting with cognitive agents to decompose logistics objectives, plan actions, and re-plan in response to incoming observations. Unlike traditional static optimization approaches, the proposed system implements perception–action feedback loops enabling real-time adaptation and robustness to incomplete or adversarial data. Expected contributions include a formalization of the perception-to-planning pipeline, a hybrid agentic optimization scheme, and evaluation scenarios demonstrating improved resilience and decision latency in non-standard logistics tasks. The research contributes to developing intelligent logistics systems capable of operating effectively in complex, non-standard, or military scenarios.

Сучасні логістичні системи функціонують у складних умовах, що характеризуються високою динамічністю, невизначеністю та нестачею достовірних даних. Це особливо актуально у контексті гуманітарних або військових операцій, де рішення щодо постачання, транспортування