The SERRANO platform: Stepping towards seamless application development & deployment in the heterogeneous edge-cloud continuum

Aggelos Ferikoglou[⋆], Argyris Kokkinis[⋆], Dimitrios Danopoulos[⋆], Ioannis Oroutzoglou[⋆]
Anastasios Nanos[†], Stathis Karanastasis[‡], Marton Sipos[⋆⋆], Javad Fadaie Ghotbi[††]
Juan Jose Vegas Olmos[‡‡], Dimosthenis Masouros[⋆], Kostas Siozios[⋆]

[⋆]*Aristotle University of Thessaloniki, Greece, Email: aferikog@physics.auth.gr*
[†]*Nubificus Ltd, United Kingdom, Email: ananos@nubificus.co.uk*
[‡]*Innovation Acts Limited, Cyprus, Email: karanastasis@gmail.com*
[⋆⋆]*Chocolate Cloud ApS, Denmark, Email: marton@chocolate-cloud.cc*
[††]*High Performance Computing Center, Germany, Email: javad.fadaieghotbi@hlrs.de*
[‡‡]*NVIDIA Corporation, Israel, Email: juanj@nvidia.com*

***Abstract*—The need for real-time analytics and faster decision-making mechanisms has led to the adoption of hardware accelerators such as GPUs and FPGAs within the edge cloud computing continuum. However, their programmability and lack of orchestration mechanisms for seamless deployment make them difficult to use efficiently. We address these challenges by presenting SERRANO, a project for transparent application deployment in a secure, accelerated, and cognitive cloud continuum. In this work, we introduce the SERRANO platform and its software, orchestration, and deployment services, focusing on its methods for automated GPU/FPGA acceleration and efficient, isolated, and secure deployments. By evaluating these services against representative use cases, we highlight SERRANO 's ability to simplify the development and deployment process without sacrificing performance.**

***Index Terms*—Edge-Cloud Continuum, Heterogeneity, SDK**

I. Introduction

The explosive growth and increasing power of IoT devices, along with the emergence of 5G networks, has led to unprecedented data volumes. Emerging use cases around smart homes, autonomous vehicles, and smart factories require edge processing due to critical real-time requirements. To this end, multi-layered computing architectures are emerging where compute resources and applications are distributed from the edge of the network, closer to the point of data collection, to the cloud, realizing the edge-cloud computing continuum [1].

Even though edge-cloud architectures extend the compute capacity of the traditional cloud paradigm, the excessive compute demands of modern use-cases require the introduction of hardware accelerators in the computing stack. Specialized hardware acceleration platforms (e.g., GPUs, FPGAs) can achieve higher performance than typical processing systems for the same power envelope [2], [3]. Typical examples of accelerators include resource-constrained devices at the edge to high-performance, massively parallel devices at the cloud.

While such hardware platforms offer performance gains, these benefits do not come for free, as they typically require hardware knowledge to program. From the developer's perspective, a trade-off arises between performance and programmability. To relieve the developer of the programming burden, tools are needed for seamless development. In addition, the introduction of these devices into the edge-cloud computing continuum underscores the need for orchestration [4] and deployment tools to ensure efficient and secure executions.

To this end, we present the H2020 project SERRANO, which provides a new ecosystem of technologies to address the challenges of introducing heterogeneity in the edge- cloud computing continuum. Specifically, the contributions of SERRANO are:

- *Software Services* for automatic optimization of applications targeting GPU and FPGA devices.
- *Orchestration Services* for end-to-end cognitive orchestration along with closed-loop control.
- *Deployment Services* for isolated and private execution of the heterogeneous compute units.

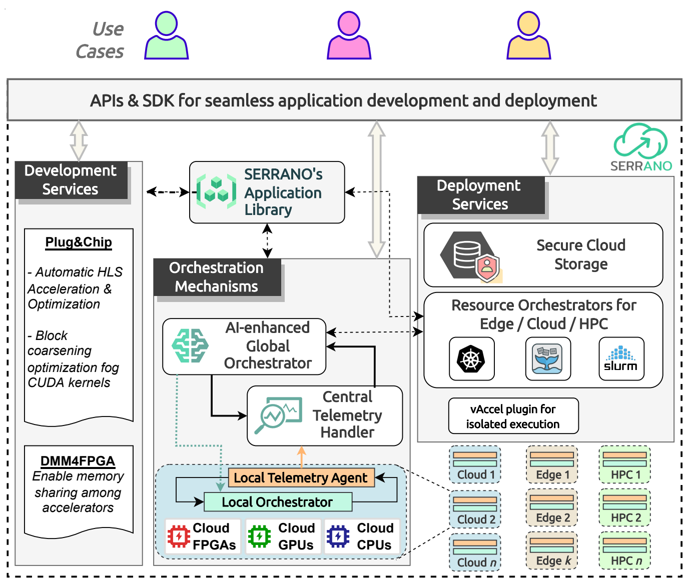
We evaluate the software and deployment services against a set of representative use cases. The experimental results highlight the ability of SERRANO to simplify the development and deployment of applications without sacrificing performance.

II. SERRANO's platform overview

The SERRANO platform combines (a) a set of tools that simplify the development/deployment process of accelerated applications and (b) runtime mechanisms that ensure that the quality of service (QoS) requirements of deployed applications are met while efficiently leveraging the underlying heterogeneous infrastructure. These technologies and services are abstracted through APIs and SDKs to simplify their use (e.g., Plug&Chip [5]). Figure 1 shows an overview of the platform.

Service Development Kit: SERRANO contributes to the development and deployment of applications that leverage heterogeneous resources by providing a Service Development Kit (SDK) to increase developer productivity in building, deploying, and managing novel applications while having better control over compute, storage, and network infrastructure. In

This work has been supported by the E.C. funded program SERRANO under H2020 Grant Agreement No: 101017168 (https://ict-serrano.eu/).



Use Cases
APIs & SDK for seamless application development and deployment
SERRANO
Development Services
Plug&Chip
- Automatic HLS Acceleration & Optimization
- Block coarsening optimization fog CUDA kernels
DMM4FPGA
Enable memory sharing among accelerators
SERRANO's Application Library
Orchestration Mechanisms
AI-enhanced Global Orchestrator
Central Telemetry Handler
Local Telemetry Agent
Local Orchestrator
Cloud FPGAs
Cloud GPUs
Cloud CPUs
Deployment Services
Secure Cloud Storage
Resource Orchestrators for Edge / Cloud / HPC
slurm
vAccel plugin for isolated execution
Cloud 1
Cloud 2
Cloud n
Edge 1
Edge 2
Edge k
HPC 1
HPC 2
HPC n

Fig. 1: The SERRANO platform

addition to increasing developer productivity and thus reducing time-to-market, the proposed toolkit provides mechanisms that can provide solutions for an application, i.e., different application versions, that compromise between performance, energy efficiency, and accuracy. Finally, the development services of SERRANO are used to optimize the computationally intensive parts of the applications of UC and to create a database of application versions with different trade-offs targeting the different resources available on the platform.

Orchestration Approach: SERRANO proposes an orchestration system that manages the underlying infrastructure in an abstract and disaggregated manner. This is achieved through a hierarchical architecture consisting of three components: i) the central resource orchestrator, ii) the local resource orchestrators, and iii) the telemetry framework.

In this context, submitting an application to the SERRANO platform requires providing a set of QoS metrics (e.g., QPS, maximum error) that describe the desired state of the application. The central orchestrator, a mechanism that knows the current state of the underline resources, decides on the optimal placement, i.e., the edge, cloud, and HPC resources that can ensure that the application's requirements are met while minimizing resource consumption and hence energy consumption of the entire infrastructure. Once the optimal placement is decided, the central resource orchestrator assigns the workload to the selected resources along with the desired QoS metrics and coordinates the necessary data movement. Then, the local orchestrators are responsible for actually deploying the application using the appropriate application versions provided by the SDK. Finally, the proper functioning of the SERRANO orchestrator would not be possible without the telemetry framework, which captures information about the current infrastructure and application status.

Use Case Scenarios: The SERRANO platform is evaluated using three use-cases from different scientific domains that illustrate the platform's ability to solve multiple computationally intensive problems with different requirements.

Secure Data Storage: This use case focuses on security and distributed data storage. Protecting files from malicious third parties is done through encryption and novel erasure coding. Therefore, SERRANO will explore acceleration solutions for encoding and decoding tasks that fragment the encoded data into multiple pieces so that the encoded pieces can be stored in distributed locations, providing a secure storage solution.

High-Performance Fintech Analysis: The second use case comes from the field of financial technology, more specifically from the field of portfolio management and analysis. It deals with various AI algorithms accelerated by the heterogeneous computing resources of SERRANO to automatically manage multiple personalized portfolios simultaneously.

Machine Anomaly Detection in Manufacturing Environments: The third use-case belongs to the field of machine anomaly detection. In particular, high-frequency sensors generate large amounts of data that are processed in real time to automatically detect anomalies in machines. SERRANO will analyze the collected data and accelerate its processing.

III. SERRANO TECHNOLOGIES AND RESOURCES

A. *SERRANO's hardware infrastructure*

The SERRANO platform encapsulates hardware and HPC platforms both at the edge and in the cloud. The cloud infrastructure is coupled with both programmable FPGA accelerators and GPU devices. Accelerators are used to increase the performance and energy efficiency of the workloads being executed.

The cloud FPGAs and GPUs connected to the cloud side of SERRANO are: **(i)** a Xilinx Alveo U200 accelerator card, **(ii)** a Xilinx Alveo U50 accelerator card, and **(iii)** two NVIDIA Tesla T4 GPUs. In addition, NVIDIA BlueField-2 data processing units (DPUs) are used. System on a Chip (SoC) devices are used in the edge infrastructure. The edge FPGA and GPU devices in the infrastructure are: **(i)** a Xilinx MPSoC ZCU102 device, **(ii)** a Xilinx MPSoC ZCU104 device, **(iii)** an NVIDIA Xavier AGX, and **(iv)** an NVIDIA Xavier NX. In addition, the SmartBox, an industrial-ready box for data acquisition at the edge of the network with a 60 GB hard disk, is used for edge processing. Among the many HPC resources available on the SERRANO platform is the HPE Apollo 9000 Hawk.

B. *SERRANO's Software Services*

1) Development Services: SERRANO provides a unified framework consisting of three tools for HLS and CUDA accelerator development and optimization.

Automatic HLS Optimization: SERRANO offers a tool that automatically optimizes synthesizable C/C++ kernels for Xilinx FPGAs through High-Level Synthesis. This optimization scheme identifies points of interest, i.e., loops and arrays, applies directives (e.g., loop unrolling, array partition), and performs synthesis to get the latency and resource utilization. By applying different combinations of directives, the optimizer proposes an approximation to the Pareto-optimal designs with respect to the underline architecture of the target device.

Due to the large design space, the DSE is performed using the algorithm NSGA-II [6]. The exploration phase consists of the following steps: a) the configuration population is initialized, b) each configuration of the current population is applied to the source code using a source-to-source compiler and the output is synthesized using the Xilinx Vitis tool chain, and c) the synthesis outputs of the population are passed to NSGA-II to build the next generation configurations. Steps b) and c) are executed iteratively until the termination criterion is reached.

Automatic CUDA Optimization: A CUDA auto-tuning framework for kernel source codes targeting Nvidia GPUs has also been developed. This framework is based on block coarsening, a kernel transformation that merges the workload of 2 or more thread blocks while keeping the number of threads per block the same. Consequently, multiple adjacent blocks are merged to deal with the issues associated with extensive fine-grained parallelism. The proposed framework is based on two components: a) a regression model trained on a representative source code dataset, and b) the source-to-source compiler. The regressor predicts the optimal block coarsening factors for different applications, workload inputs, and GPU architectures, while the source-to-source compiler applies the predicted source code transformations.

Dynamic Memory Management in HLS: A tool has been developed that allows multiple FPGA-implemented HLS accelerators to share and reuse on-chip memory resources at execution time with minimal external fragmentation [7], [8]. This tool groups portions of on-chip memory into structures comparable to those of traditional computer architectures, forming a shared memory area consisting of the local memories of the reconfigurable platform. Dynamic memory allocation is performed by the HLS accelerators through a first-fit allocator implemented on-chip.

To minimize external fragmentation of the heaps and thus increase memory efficiency, an HLS on-chip garbage collector is implemented to compact the fragmented memory regions of the heaps. To reduce the performance overhead of executing the garbage collector, an offline stochastic analysis of the accelerators' memory patterns is performed to determine when (i.e., at the time when the heaps' fragmentation percentage exceeds a user-defined fragmentation threshold called Θ) the compaction algorithm should be executed. This analysis is based on a Monte-Carlo model that pseudo-randomly emulates the memory patterns created by running many accelerators in parallel for different Θ thresholds.

2) Orchestration Services: The AI orchestration services of SERRANO follow a hierarchical architecture that provides end-to-end closed-loop orchestration of running workloads. At the top is the AI-assisted service orchestrator (AISO), which works in conjunction with the resource orchestrators responsible for the application deployments.

AI-enhanced service orchestrator: The AISO is the central orchestrator of the platform and is responsible for translating the parameters specified by the end user into the appropriate deployment constraints. The output of this component is a list of possible deployment scenarios (edge, cloud). The input of the AISO is a JSON file containing various constraints related to the application and its execution. The main component of the AISO is the Requests Manager, which is responsible for processing the data provided by the end user regarding the execution of an application, as well as the telemetry data collected by the telemetry framework. The other components of the AISO are the translation mechanism and the prediction mechanism. These two mechanisms make decisions about deployment scenarios after analyzing the telemetry data (e.g., compute, memory, disk, network, and hardware information) and user parameters based on machine learning models (ML).

3) Deployment Services: SERRANO contributes to application deployment providing services that target trusted execution, workload isolation, and lightweight virtualization.

vAccel framework: To achieve isolated and private execution of the hardware accelerators on the platform's disaggregated infrastructure, the vAccel framework [9]is used to virtualize and deploy the hardware accelerators in multi-tenant environments. vAccel decouples the user application from the HLS and CUDA kernels. The actual hardware-specific code that implements these functions for a given hardware device is provided in the form of plugins that are loaded at runtime. Consequently, the deployed applications can be migrated from one host to another without the need to modify or recompile the code. At the same time, its modular nature prevents user code from running on shared accelerators. Only the code contained in the plugin is executed on the hardware accelerator, which enables secure hardware execution.

IV. Experimental Results

A. Development Services

Automatic HLS Optimization: The automatic HLS optimization scheme was used to optimize the Kalman filter algorithm provided by the high-performance fintech analysis use case. The proposed methodology is compared to a) the HLS optimizations performed by the Vitis Unified Software platform (version 2021.1), b) a human expert, and c) an automated optimizer that randomly navigates the design space of directives. The proposed NSGA-II based (GenOpt) and the random (RandOpt) optimizers operate for 12 hours. The optimization process targets the Xilinx MPSoC ZCU104 FPGA available on the SERRANO platform. We synthesized the input source code using the HLS optimizations performed by Vitis and determined the latency and resources of the output design. This optimization scheme results in an infeasible design (BRAM%>100%), showing that it cannot always account for the architecture of the target device. GenOpt achieves 40% and 27% lower latency compared to RandOpt (1.023 ms) and human expert (1.009 ms), respectively, with no significant difference in resource usage. Consequently, the proposed optimization method outperforms both the naive automatic DSE approach and the human expert in terms of latency and yields designs that take into account the available resources without human intervention.

Automatic CUDA Optimization: The automatic CUDA optimization method was evaluated using PolyBench-ACC, an open-source benchmark suite that includes CUDA kernels from

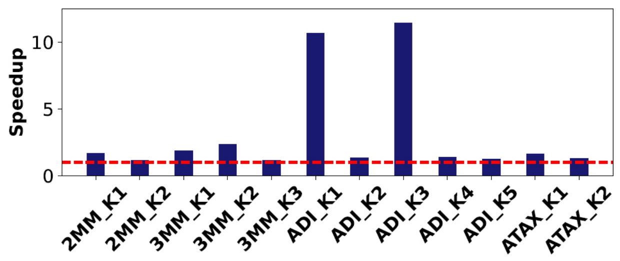


Fig. 2: CUDA automatic optimizer evaluation

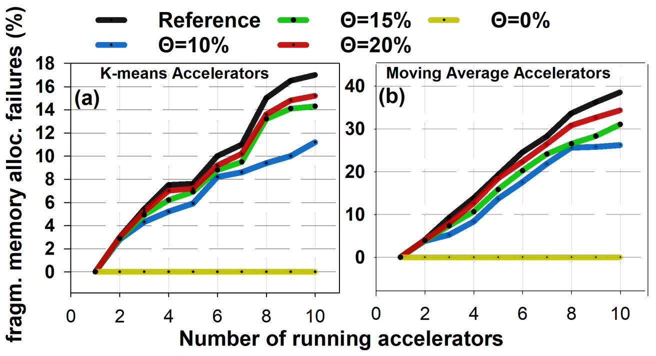


Fig. 3: Percentage of the fragmentation induced allocation failures of the overall 10,000 parallel executions of the (a) K-means HLS accelerators and (b) moving average HLS accelerators

data mining, linear algebra, and stencils. The proposed scheme also targets various edge-cloud GPUs from Nvidia (e.g., Xavier NX, T4) available on the SERRANO platform. To determine the best predictor for the block coarsening factor, several regression models were evaluated using MSE and the R^2 metric. For the model evaluation phase, the 10% of PolyBench-ACC suite was used. XGBoost has the highest prediction accuracy with an MSE of 0.02 and R^2 metric of 0.88. Figure 2 shows the speedup of the optimized source code targeted to the Nvidia T4 GPU for the evaluation dataset applications. As shown, the automatic CUDA optimizer achieves an average speedup of $\times 3.1$ compared to the native implementations.

Dynamic Memory Management in HLS: The implemented garbage collector and defragmentation methodology were evaluated on the Alveo U200 FPGA of the SERRANO platform to highlight their effectiveness in improving the memory efficiency of the platform under different threshold Θ and for different number of running accelerators. Figure 3 shows the percentage of fragmentation-induced memory allocation errors at different thresholds Θ when a different number (from 1 to 10) of accelerators run in parallel on the same platform and share a single heap. Note that the HLS accelerators K-means and moving average were run 10,000 times and statistical analysis was performed. The black reference lines correspond to HLS accelerators that do not use the garbage collector. At lower Θ values, the garbage collector is activated more frequently, leading to a significant reduction in fragmentation levels and, consequently, fragmentation-related allocation failures. However, frequent execution of the garbage collector introduces latency in the execution of the accelerators.

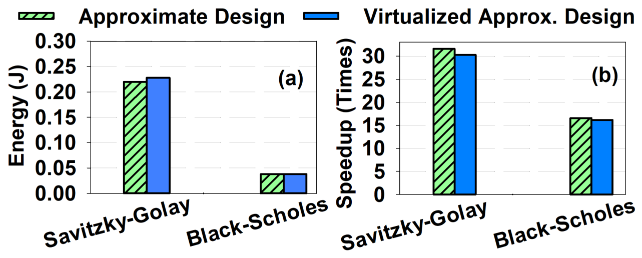


Fig. 4: Accelerator energy consumption (a) and execution time speedup (b) on the Alveo U50 for the virtualized/non-virtualized designs

B. Deployment Services

The vAccel framework was evaluated using two applications from the high-performance fintech analysis use-case: (i) the Savitzky-Golay filter and (ii) the Black-Scholes algorithm. These applications were virtualized via the vAccel framework, ported via the `vsock` socket, and executed on the Alveo U50 FPGA. Figure 4 shows the energy consumption and execution time speedup of the two HLS accelerators when running on the selected platform (patterned in green) and when running the same designs after they are virtualized by vAccel (blue). Virtualizing the designs results in negligible degradation in the energy consumption and speedup of the accelerators, ranging between 2% and 4% compared to the non-virtualized designs.

V. Conclusion

In this paper, we introduce the SERRANO platform and its software, orchestration, and deployment services, focusing on its methods for automated GPU/FPGA acceleration and efficient, isolated, and secure deployments. By evaluating these services against representative use cases, we highlight SERRANO 's ability to simplify the development and deployment process without sacrificing performance, underscoring its importance for heterogeneous resource adoption in the edge-cloud computing continuum.

References

[1] W. Shi *et al.*, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[2] D. Danopoulos *et al.*, "Approximate similarity search with faiss framework using fpgas on the cloud," in *Embedded Computer Systems: Architectures, Modeling, and Simulation*, (Cham), pp. 373–386, Springer International Publishing, 2019.

[3] D. Danopoulos *et al.*, "Adapt: Fast emulation of approximate dnn accelerators in pytorch," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2022.

[4] A. Tzenetopoulos *et al.*, "Evolve: Towards converging big-data, high-performance and cloud-computing worlds," in *2022 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 975–980, 2022.

[5] D. Diamantopoulos *et al.*, "Plug&chip: A framework for supporting rapid prototyping of 3d hybrid virtual socs," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 13, no. 5s, pp. 1–25, 2014.

[6] J. Blank *et al.*, "pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89497–89509, 2020.

[7] D. Diamantopoulos *et al.*, "Mitigating memory-induced dark silicon in many-accelerator architectures," *IEEE Computer Architecture Letters*, vol. 14, no. 2, pp. 136–139, 2015.

[8] A. Kokkinis *et al.*, "Dynamic optimization of on-chip memories for hls targeting many-accelerator platforms," *IEEE Computer Architecture Letters*, vol. 21, no. 2, pp. 41–44, 2022.

[9] Nubificus LTD, "Interoperable Hardware acceleration for Serverless Computing," 2020.