Computing Architectures and Data Systems
REVIEW | COMMISSIONED SYNTHESIS

Compiler Optimization for Heterogeneous Computing

Christian Murphy - Corresponding author

Abstract

This review examines compiler optimization for heterogeneous computing. The organizing question is how compiler transformations should balance portability, performance, energy, and numerical fidelity across devices. Ten related scholarly sources are synthesized through a decision-centered framework spanning problem definition, mechanism, measurement, evaluation, implementation, and governance. The review does not invent experiments, pooled estimates, or unreported quantitative results. It instead evaluates the strength and transferability of the available evidence, with particular attention to reporting device-specific tuning as a portable compiler improvement. The resulting framework links technical or empirical performance to explicit use conditions and identifies tests that should precede wider adoption in scientific, data-intensive, and machine-learning workloads.

Keywords: compilers, heterogeneous computing, accelerators, optimization, portability

Synthesis lens	Principal question
Mechanism	the chain connecting evidence inputs, compiler optimization for heterogeneous computing, intermediate outputs, and downstream decisions
Evaluation	construct validity, comparative performance, uncertainty, transfer across settings, and decision utility
Primary risk	reporting device-specific tuning as a portable compiler improvement

1. Introduction

Research on compiler optimization for heterogeneous computing sits at the boundary between a promising component and a consequential decision. The core question is how compiler transformations should balance portability, performance, energy, and numerical fidelity across devices. Answering it requires more than assembling favorable results. It requires a chain of evidence that makes the problem boundary, mechanism, measurement, comparator, uncertainty, and accountable decision visible to the reader.

This article presents a structured narrative review of ten related publications. Sources were selected for topical relevance, traceable publication metadata, and complementary coverage of technical, empirical, and governance questions. No meta-analytic pooling is attempted because the included studies differ in designs, populations, system boundaries, and outcomes. The purpose is decision-oriented synthesis: to identify what each source can support, where transfer remains uncertain, and which evidence would justify the next stage of use.

The central risk is reporting device-specific tuning as a portable compiler improvement. A top-line metric or broad policy label can appear decisive while concealing the conditions that produced it. Accordingly, the synthesis separates observation from interpretation and implementation from validation. This separation is especially important when the same system creates benefits for one user or institution while transferring cost, risk, or administrative burden to another.

2. Evidence Base and Problem Boundary

A defensible review begins by defining the unit of analysis. For the present topic, the relevant boundary includes inputs, institutional or technical context, the mechanism producing an output, the person or system acting on that output, and the downstream outcome. Evidence falls outside the boundary when it measures only an intermediate proxy yet is used to claim an end-to-end benefit.

Xing et al. (2019) addresses "DNNVM: End-to-End Compiler Leveraging Heterogeneous Optimizations on FPGA-Based CNN Accelerators." In this synthesis, that work is used as a source on problem definition within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Willsey et al. (2018) addresses "Iterative Search for Reconfigurable Accelerator Blocks With a Compiler in the Loop." In this synthesis, that work is used as a source on conceptual boundary within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Ashcraft et al. (2017) addresses "Compiler Optimization of Accelerator Data Transfers." In this synthesis, that work is used as a source on measurement within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Together, these works provide complementary entry points, but their conclusions should not be flattened into a single ranking. Differences in data generation, setting, temporal horizon, and outcome definition may be substantively meaningful. A review should therefore preserve those differences and state where analogy, rather than direct replication, connects one domain to another.

3. Mechanism and System Design

The operative mechanism is the chain connecting evidence inputs, compiler optimization for heterogeneous computing, intermediate outputs, and downstream decisions. This formulation discourages component-only reasoning. A model, platform, policy, assay, or infrastructure service acquires practical meaning only when its interfaces and use conditions are specified. The evidence chain should describe what information is available, what transformation occurs, which assumptions make that transformation credible, and who is authorized to act.

Park et al. (2024) addresses "NEST-C: A deep learning compiler framework for heterogeneous computing systems with artificial intelligence accelerators." In this synthesis, that work is used as a source on system mechanism within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Venkataramanaiah et al. (2019) addresses "Automatic Compiler Based FPGA Accelerator for CNN Training." In this synthesis, that work is used as a source on representation choice within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Şuşu (2020) addresses "A Vector-Length Agnostic Compiler for the Connex-S Accelerator with Scratchpad Memory." In this synthesis, that work is used as a source on failure analysis within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

These studies also show why modular claims require interface tests. A component may perform well while the complete pathway fails because inputs drift, users interpret outputs differently, recovery semantics are ambiguous, or institutional incentives change. Design documentation should therefore include not only the intended pathway but also foreseeable deviations, degraded modes, and conditions under which the system should stop or defer.

4. Evaluation and Uncertainty

Evaluation should center on construct validity, comparative performance, uncertainty, transfer across settings, and decision utility. Average performance remains useful, but it should be accompanied by distributions, error categories, relevant subgroups or operating regimes, and uncertainty at the point where a decision changes. Comparators must isolate the value of the proposed addition rather than compare a mature intervention with an intentionally weak baseline.

Chang et al. (2019) addresses "Deep neural networks compiler for a trace-based accelerator." In this synthesis, that work is used as a source on comparative evaluation within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Hayashi et al. (2016) addresses "Exploring compiler optimization opportunities for the OpenMP 4.x accelerator model on a POWER8+GPU platform." In this synthesis, that work is used as a source on uncertainty within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

External validation should introduce meaningful shifts in setting, time, population, workload, market structure, or environmental conditions. A nominally independent sample can still be too similar to test transfer. Conversely, a failed transfer is scientifically informative when the changed conditions are measured and the mechanism of failure is examined rather than hidden in an aggregate score.

5. Translation and Governance

Translation to scientific, data-intensive, and machine-learning workloads depends on more than technical readiness. Decision rights, documentation, maintenance, monitoring, appeal, and recovery are part of the intervention. The governance requirement is traceable evidence, named responsibility, version control, monitoring, appeal, and explicit revalidation. Each control should be connected to a named failure mode and should identify an owner, evidence source, review interval, and escalation path.

Liu et al. (2020) addresses "Survey and design of paleozoic: a high-performance compiler tool chain for deep learning inference accelerator." In this synthesis, that work is used as a source on implementation within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

Tiotto et al. (2020) addresses "OpenMP 4.5 compiler optimization for GPU offloading." In this synthesis, that work is used as a source on governance within compiler optimization for heterogeneous computing. The connection is deliberately bounded: the cited study informs the evidence map, but it does not by itself establish readiness for scientific, data-intensive, and machine-learning workloads. That distinction keeps the review close to the published record and prevents an adjacent result from being converted into a broader causal claim.

A credible implementation plan also defines sunset and revalidation conditions. New data, software updates, institutional restructuring, population change, or shifts in incentives can invalidate previously reasonable assumptions. Monitoring should therefore test the validity of the evidence chain, not merely whether a service remains online or a headline metric remains stable.

6. Research Agenda

Future studies should preregister or otherwise predefine the intended decision, principal outcomes, exclusions, and analysis plan when feasible. They should report missingness, sensitivity analyses, negative or null findings, and the cost of errors to differently situated stakeholders. Reproducible artifacts should include sufficient metadata to reconstruct data provenance, model or analytical versions, parameter choices, and the sequence from evidence to conclusion.

Cross-disciplinary work needs a shared object of explanation rather than a collection of adjacent vocabularies. For compiler optimization for heterogeneous computing, that shared object is the complete decision pathway. Technical, clinical, social, environmental, or economic expertise should each change the design or interpretation of the study. When evidence remains incomplete, the useful output is a bounded hypothesis, a transparent uncertainty statement, or a request for additional measurement.

7. Conclusion

The literature supports a disciplined approach to compiler optimization for heterogeneous computing: define the decision, preserve system boundaries, test consequential failure modes, connect uncertainty to action, and assign accountable control. The strongest conclusion is not that one method is universally superior. It is that credible use in scientific, data-intensive, and machine-learning workloads requires an auditable link from source evidence to design choice, evaluation result, and governed decision.

Declarations

Ethics approval: Not applicable. This article synthesizes previously published literature and reports no new research involving humans, animals, human tissue, or identifiable sensitive data.

Consent to participate and consent for publication: Not applicable.

Funding: No external funding is reported for this commissioned synthesis.

Competing interests: The authoring group declares no competing interests for this commissioned synthesis.

AI-assisted writing: Generative AI supported drafting, source organization, and language editing. Accountable authors and editors must verify every claim, citation, and disclosure before publication.

Data, code, and materials availability: No new dataset or code was generated. All evidence is identified in the reference list.

References

Ashcraft, M. B., Lemon, A., Penry, D. A., & Snell, Q. (2017). Compiler Optimization of Accelerator Data Transfers. International Journal of Parallel Programming, 47(1), 39-58. https://doi.org/10.1007/s10766-017-0549-3

Chang, A. X. M., Zaidy, A., Vitez, M., Burzawa, L., & Culurciello, E. (2019). Deep neural networks compiler for a trace-based accelerator. Journal of Systems Architecture, 102, 101659. https://doi.org/10.1016/j.sysarc.2019.101659

Hayashi, A., Shirako, J., Tiotto, E., Ho, R., & Sarkar, V. (2016). Exploring compiler optimization opportunities for the OpenMP 4.x accelerator model on a POWER8+GPU platform. Scholarly publication, 68-78. https://doi.org/10.5555/3019120.3019127

Liu, Z., Leng, J., Lu, G., Wang, C., Chen, Q., & Guo, M. (2020). Survey and design of paleozoic: a high-performance compiler tool chain for deep learning inference accelerator. CCF Transactions on High Performance Computing, 2(4), 332-347. https://doi.org/10.1007/s42514-020-00044-7

Park, J., Yu, M., Kwon, J., Park, J., Lee, J., & Kwon, Y. (2024). NEST-C: A deep learning compiler framework for heterogeneous computing systems with artificial intelligence accelerators. ETRI Journal, 46(5), 851-864. https://doi.org/10.4218/etrij.2024-0139

Tiotto, E., Mahjour, B., Tsang, W., Xue, X., Islam, T., & Chen, W. (2020). OpenMP 4.5 compiler optimization for GPU offloading. IBM Journal of Research and Development, 64(3/4), 14:1-14:11. https://doi.org/10.1147/jrd.2019.2962428

Venkataramanaiah, S. K., Ma, Y., Yin, S., Nurvithadhi, E., Dasu, A., Cao, Y., & Seo, J. S. (2019). Automatic Compiler Based FPGA Accelerator for CNN Training. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1908.06724

Willsey, M., Lee, V. T., Cheung, A., Bodik, R., & Ceze, L. (2018). Iterative Search for Reconfigurable Accelerator Blocks With a Compiler in the Loop. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(3), 407-418. https://doi.org/10.1109/tcad.2018.2878194

Xing, Y., Liang, S., Sui, L., Jia, X., Qiu, J., Liu, X., Wang, Y., Shan, Y., & Wang, Y. (2019). DNNVM: End-to-End Compiler Leveraging Heterogeneous Optimizations on FPGA-Based CNN Accelerators. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(10), 2668-2681. https://doi.org/10.1109/tcad.2019.2930577

Şuşu, A. E. (2020). A Vector-Length Agnostic Compiler for the Connex-S Accelerator with Scratchpad Memory. ACM Transactions on Embedded Computing Systems, 19(6), 1-30. https://doi.org/10.1145/3406536