# Symphony: Leveraging Probabilistic Graphical Models to Schedule Tasks to Clusters of Heterogeneous Processors

Subho S. Banerjee, Steve Lumetta, Zbigniew T. Kalbarczyk, Ravishankar K. Iyer

Departments of Computer Science and Electrical and Computer Engineering,
University of Illinois at Urbana-Champaign

## Abstract

In this paper, we present a summary of the CompGen system at the UIUC that addresses several problems at the intersection of healthcare (especially computational genomics), analytic tools and methods, and novel computer system architecture and design. The system uses a hardware-software co-design approach to improve computational performance and energy consumption significantly. The primary source of this improvement are i) the use of application-specific custom accelerators prototyped on FPGAs, as well as off-the-shelf accelerators like GPUs and MICs, and ii) the system uses an intelligent scheduling algorithm, Symphony, which utilizes real-time measurements from system performance counters and past data about application characteristics and fuses them using Bayesian inference in probabilistic graphical models to predict resource contention at a microarchitectural level thereby distributing computations effectively across a heterogeneous set of processors. Using the "Variant-Calling and Genotyping Analysis" as a driving example, we demonstrate the performance and isolation properties of Symphony. Our evaluation shows that the use of Symphony (to schedule tasks among accelerators) improves overall benchmark performance (on a single 2-socket CPU system) from 73 hours to under 45 minutes ($88\times$ and nearly $210\times$ in terms of performance-per-watt) for human genome datasets that are appropriate for clinical use.

## 1. Introduction

The use of application-specific accelerators like GPUs, FPGAs, ASICs are now becoming an integral part of modern datacenter workloads across supercomputing and cloud-service providers like Google, Microsoft, and Amazon (6; 8; 9; 10; 11). In addition to the challenge of designing these accelerators, a significant challenge in this setting comprises the design of runtime systems and scheduling strategies that can easily abstract away low-level systems details while maintaining optimal application performance. This extended abstract presents the design of the CompGen system at UIUC that addresses the above challenges with hardware-software co-design to build a computing engine tailored for computational genomics applications. At the heart of the system is Symphony, an intelligent scheduling algorithm that models the computational genomics applications, the system architecture, and real-time measurements coming from dynamic profiling of the system into a probabilistic graphical model (most generically expressed as a Factor Graph). It consequently uses approximate Bayesian inference to compute efficient schedules that can distribute the workloads across the heterogeneous processing fabric. Though our system is tailored for computational genomics applications, we claim that the lessons learned from this design can spur future applications that take advantage of heterogeneous compute infrastructure.

Two key aspects of the system presented here are:

1. *Hardware Component* – The architecture of "The Computational Genomics Accelerator" (TCGA), a FPGA-based dynamically reconfigurable co-processor that is specialized for computational *kernels* which are ubiquitous in a variety of genomic data analyses.
2. *Software Component* (and the focus of this abstract) – The design of "Symphony" a runtime system and scheduling algorithm that serves as an abstraction layer to provide a high-level programming interface to an underlying set of heterogeneous processors like TCGA, CPUs, and GPUs.

The subsequent sections describe each of these components. Due to space limitations, we briefly describe the hardware component and focus our attention on the design of the runtime system (*i.e., the scheduler*).

## 2. The Hardware Component

An essential characteristic of computational genomics workloads is that a small number of computationally intensive *kernels* (e.g., computation of Levenshtein distance, computation of probabilities using the forward algorithm on hidden Markov models) are reused across many different types of analyses, and contribute to the significant portion of their runtime (3; 2). The TCGA architecture, which is reminiscent of many-core processor designs, takes advantage of this workload characteristic. It consists of many *systolicized* reconfigurable processing elements (PEs) that efficiently compute the kernels mentioned above (4; 5). These PEs are dynamically reconfigured based on execution phase of the current application. As a result, TCGA can leverage data-path specialization for each kernel, thereby extracting maximal performance and performance per unit energy consumed. The use of reconfigurable logic fabrics like FPGAs provides TCGA the flexibility to be utilized in a range of computational genomics analyses

(i.e., reconfigure the device to the current set of kernels), and keep abreast of algorithmic changes in these analyses (i.e., update the library of available kernels). Further, as computational genomics tools are chained together in workflows (which is indeed the universal use-case (12)), TCGA can take advantage of cross-tool optimizations, and coalesce kernels (through systolicization) to reduce overheads (both in time and energy) for moving data through the system.

## 3. The Software Component

To use TCGA in conjunction with CPUs and other general purpose accelerators like GPUs and MICs we present Symphony, a runtime framework (and underlying scheduling algorithm) that allows automation of low-level system-tasks like 1. reconfiguration of the TCGA accelerator, and 2. transparent scheduling of tasks to all processors in the system (e.g., CPUs, GPUs and TCGA). Applications for the CompGen system are described as acyclic control- and data-flow graph (CDFG) consisting of the aforementioned computational kernels. Such a CDFG based programming model is common in modern data analytics frameworks like (7; 13) and allows the programmer to write code for a homogeneous model of processors while the runtime framework appropriately interprets it to optimize performance/energy (and potentially accuracy) tradeoffs for each of the underlying heterogeneous processors. Symphony, in conjunction with the operating system, serves as an abstraction layer between the heterogeneous compute fabric and the application CDFG.

Symphony uses a data-driven strategy of integrating real-time measurements of performance counter events, prior knowledge about workloads, and information about system architecture and topology (i.e., interconnect and PCIe network topologies) into a probabilistic graphical model (PGM; specifically a factor graph). These real-time measurements from the processor's performance counters are then used to infer hidden system resource utilization using Bayesian statistics. The factor graph is the most generalized version of such a Bayesian model that integrates probabilistic (data-driven) and algebraic (system architecture) relationships between the hidden and observed variables. The system's data-driven component of the model is used to describe application-to-system resource relationships, potential errors in measurements because of restrictions on the number of performance counters on the CPU, as well as potential network delays in relaying measurements over a distributed system. These are used to efficiently search the performance-cost trade-off space (using probabilistic inference) thereby making intelligent scheduling decisions by taking into account

1. **Processor Affinity**: Relating application defined notions of performance (e.g., goodput, throughput, latency) and efficiency (e.g., power, energy, cost) with available processors and accelerators.
2. **Data Locality**: Minimizing intra-computer (over the memory hierarchy and system bus) and inter-computer (over the network) movement of data and offsetting this cost with potential gains in performance by using accelerators.
3. **Shared Resource Contention**: Maximizing isolation between co-located tasks by tracking the utilization of key processor- and system-level resources that adversely affect performance.

The technique improves overall utilization of compute resources by dense packing of tasks based on fine-grained resource sharing, as well as enabling the efficient use of accelerators. Symphony also facilitates the execution of CDFG based applications in a distributed setting, much like (7; 13).

We demonstrate the efficacy of the TCGA accelerator and runtime environment for the *variant calling and genotyping analysis* (12) on human genome datasets appropriate for clinical use. Taken together, on a single node, the accelerator, and the runtime improve runtime of the variant calling and genotyping workload from 73 hours to under 45 minutes ($88\times$) and performance per unit energy by $\sim 210\times$ over a software baseline executing on a 192-thread, 24-core IBM Power8 system. We evaluate the efficacy of the proposed techniques on a heterogeneous 11 node test-bed containing Intel Xeon and IBM Power8 CPUs, NVIDIA K40 and K80 GPUs, as well as Xilinx and Altera FPGAs (which is representative of the maximal diversity in accelerator deployments for datacenters like Microsoft Azure (6) or Amazon EC2 (1)). Our experiments show that the proposed system shows near-linear weak scaling behavior for the experimental test-bed.

## 4. Concluding Remarks

In summary, the key innovations in this work are:

1. Identifies key application-level and system-architecture-level factors that significantly affect performance of user applications when running on heterogeneous systems. These factors form the basis for modeling resource usage on target platforms.
2. Presents the design of a task scheduling system that uses quantifiable models for processor affinity, data locality and shared resource contention to abstract away heterogeneous hardware resources and present to the user, a homogeneous view of the system.
3. Presents a mechanism to trade-off accuracy and overhead of the scheduling decisions. Thereby reducing the total number of computations required for probabilistic inference on the proposed PGM.

## Acknowledgments

This research was supported by several grants: in part by the National Science Foundation under Grant No. CNS 13-37732; in part by the Blue Waters sustained-petascale computing project supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois; and in part by IBM, Xilinx, Intel for providing equipment support.

## References


[1] AMAZON. Amazon EC2 F1 Instances, 2017. https://aws.amazon.com/ec2/instance-types/f1/ [Accessed 2017-03-05].

[2] ASANOVIC, K., BODIK, R., DEMMEL, J., KEAVENY, T., KEUTZER, K., KUBIATOWICZ, J., MORGAN, N., PATTERSON, D., SEN, K., WAWRZYNEK, J., WESSEL, D., AND YELICK, K. A view of the parallel computing landscape. *Commun. ACM 52*, 10 (Oct. 2009), 56–67.

[3] BANERJEE, S. S., EL HADEDY, M., LIM, J. B., CHEN, D., KALBARCZYK, Z. T., CHEN, D., AND IYER, R. K. Asap: Accelerated short read alignment on programmable hardware (abstract only). In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays* (New York, NY, USA, 2017), FPGA '17, ACM, pp. 293–294.

[4] BANERJEE, S. S., EL HADEDY, M., LIM, J. B., CHEN, D., KALBARCZYK, Z. T., CHEN, D., AND IYER, R. K. Asap: Accelerated short read alignment on programmable hardware (abstract only). In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays* (New York, NY, USA, 2017), FPGA '17, ACM, pp. 293–294.

[5] BANERJEE, S. S., EL HADEDY, M., TAN, C. Y., KALBARCZYK, Z. T., LUMETTA, S., AND IYER, R. K. On accelerating pair-HMM computations in programmable hardware. In *2017 27th International Conference on Field Programmable Logic and Applications (FPL)* (sep 2017), IEEE.

[6] CAULFIELD, A. M., CHUNG, E. S., PUTNAM, A., ANGEPAT, H., FOWERS, J., HASELMAN, M., HEIL, S., HUMPHREY, M., KAUR, P., KIM, J. Y., LO, D., MASSENGILL, T., OVTCHAROV, K., PAPAMICHAEL, M., WOODS, L., LANKA, S., CHIOU, D., AND BURGER, D. A cloud-scale acceleration architecture. In *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)* (Oct 2016), pp. 1–13.

[7] CHAMBERS, C., RANIWALA, A., PERRY, F., ADAMS, S., HENRY, R., BRADSHAW, R., AND NATHAN. Flumejava: Easy, efficient data-parallel pipelines. In *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI)* (2 Penn Plaza, Suite 701 New York, NY 10121-0701, 2010), pp. 363–375.

[8] JOUPPI, N. P., YOUNG, C., PATIL, N., PATTERSON, D., AGRAWAL, G., BAJWA, R., BATES, S., BHATIA, S., BODEN, N., BORCHERS, A., BOYLE, R., CANTIN, P., CHAO, C., CLARK, C., CORIELL, J., DALEY, M., DAU, M., DEAN, J., GELB, B., GHAEMMAGHAMI, T. V., GOTTIPATI, R., GULLAND, W., HAGMANN, R., HO, R. C., HOGBERG, D., HU, J., HUNDT, R., HURT, D., IBARZ, J., JAFFEY, A., JAWORSKI, A., KAPLAN, A., KHAITAN, H., KOCH, A., KUMAR, N., LACY, S., LAUDON, J., LAW, J., LE, D., LEARY, C., LIU, Z., LUCKE, K., LUNDIN, A., MACKEAN, G., MAGGIORE, A., MAHONY, M., MILLER, K., NAGARAJAN, R., NARAYANASWAMI, R., NI, R., NIX, K., NORRIE, T., OMERNICK, M., PENUKONDA, N., PHELPS, A., ROSS, J., SALEK, A., SAMADIANI, E., SEVERN, C., SIZIKOV, G., SNELHAM, M., SOUTER, J., STEINBERG, D., SWING, A., TAN, M., THORSON, G., TIAN, B., TOMA, H., TUTTLE, E., VASUDEVAN, V., WALTER, R., WANG, W., WILCOX, E., AND YOON, D. H. In-datacenter performance analysis of a tensor processing unit. *CoRR abs/1704.04760* (2017).

[9] PUTNAM, A., CAULFIELD, A. M., CHUNG, E. S., CHIOU, D., CONSTANTINIDES, K., DEMME, J., ESMAEILZADEH, H., FOWERS, J., GOPAL, G. P., GRAY, J., HASELMAN, M., HAUCK, S., HEIL, S., HORMATI, A., KIM, J.-Y., LANKA, S., LARUS, J., PETERSON, E., POPE, S., SMITH, A., THONG, J., XIAO, P. Y., AND BURGER, D. A reconfigurable fabric for accelerating large-scale datacenter services. In *Proceeding of the 41st Annual International Symposium on Computer Architecuture* (Piscataway, NJ, USA, 2014), ISCA '14, IEEE Press, pp. 13–24.

[10] SHAO, Y., AND BROOKS, D. *Research Infrastructures for Hardware Accelerators*. Synthesis Lectures on Computer Architecture. Morgan & Claypool Publishers, 2015.

[11] SHAW, D. E., GROSSMAN, J. P., BANK, J. A., BATSON, B., BUTTS, J. A., CHAO, J. C., DENEROFF, M. M., DROR, R. O., EVEN, A., FENTON, C. H., FORTE, A., GAGLIARDO, J., GILL, G., GRESKAMP, B., HO, C. R., IERARDI, D. J., ISEROVICH, L., KUSKIN, J. S., LARSON, R. H., LAYMAN, T., LEE, L. S., LERER, A. K., LI, C., KILLEBREW, D., MACKENZIE, K. M., MOK, S. Y. H., MORAES, M. A., MUELLER, R., NOCIOLO, L. J., PETICOLAS, J. L., QUAN, T., RAMOT, D., SALMON, J. K., SCARPAZZA, D. P., SCHAFER, U. B., SIDDIQUE, N., SNYDER, C. W., SPENGLER, J., TANG, P. T. P., THEOBALD, M., TOMA, H., TOWLES, B., VITALE, B., WANG, S. C., AND YOUNG, C. Anton 2: Raising the bar for performance and programmability in a special-purpose molecular dynamics supercomputer. In *SC14: International Conference for High Performance Computing, Networking, Storage and Analysis* (Nov 2014).

[12] VAN DER AUWERA, G. A., CARNEIRO, M. O., HARTL, C., POPLIN, R., DEL ANGEL, G., LEVY-MOONSHINE, A., JORDAN, T., SHAKIR, K., ROAZEN, D., THIBAULT, J., BANKS, E., GARIMELLA, K. V., ALTSHULER, D., GABRIEL, S., AND DEPRISTO, M. A. *From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline*. John Wiley & Sons, Inc., 2013.

[13] ZAHARIA, M., CHOWDHURY, M., DAS, T., DAVE, A., MA, J., MCCAULEY, M., FRANKLIN, M. J., SHENKER, S., AND STOICA, I. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2012), NSDI'12, USENIX Association, pp. 2–2.