A High Performance C++ Generic Benchmark for Computational Epidemiology

Aniket Pugaonkar, Sandeep Gupta, Keith R. Bisset and Madhav V. Marathe (aniketnp, sandeep, kibisset, mmarathe)@vbi.vt.edu

The Network Dynamics and Simulation Science Laboratory, Virginia Tech, USA

INTRODUCTION

We design a benchmark consisting of several kernels which capture the essential computational tasks involved in simulating the patterns of high-performance contagion diffusion simulations used in computational epidemiology. The kernels serve as generic components that are flexible for implementing a wide range of alternative implementations for computing the contagion. By configuring different implementations of the kernels, we can perform experiments that yield useful information about the performance of various architectures on typical benchmark problems.

The proposed benchmark is designed using C++ generic programming constructs and templates. It is flexible for parallelizing the computation to explore the scalability of different compute platforms. The benchmark is agnostic to any particular high-performance computing platform, and it can be used to evaluate the performance of a wide range of hardware and software architectures.

CHALLENGES AND GOALS

- **Challenges:**
  - Generic algorithms used in application kernels need to be fine-tuned for specific platforms.
  - Designing and implementing strategies to handle the memory management and data structures on different platforms.
  - Behavioral constraints due to different processor families.
  - Handling a diverse range of architectures.

- **Goals:**
  - Evaluate the performance of the kernels on different compute platforms.
  - Develop a flexible and scalable computational framework.

CONTRIBUTIONS

- Develop benchmark specifications for our kernels.
- Develop parallel and distributed memory architectures to simulate varied techniques for modeling disease diffusion.
- Provide generic kernels for graph-based and contact-based calculation engines.
- Develop an effective system for controlling the execution of the kernels on different platforms.
- Develop a benchmark suite that can be evaluated on a wide range of architectures.

RELATED STUDY

- Several studies have been conducted on the performance of kernel implementations using parallel and distributed memory systems. Often, the focus has been on the scalability and speedup of the kernels.
- Application-specific benchmarking is essential for fast research – to verify that the parallel and distributed memory implementations are independent of choice.

CITATION MODEL

|computes the cost of the 
kernel execution derived from Table 1, 
where d is the work per thread 
and I is the inter-node communication cost. 
Then, we can compute the contribution of each  
kernel to the total execution time. 

**KERNEL FLOW DIAGRAMS**

**KERNEL 4**

- Computes the costs for the 
4 kernels (Person), 
L1 (Adjacency Matrix), 
L2 (Global Matrix), 
L3 (Comm Matrix), 
L4 (Comm Matrix), 
L5 (Comm Matrix).

**KERNEL 5**

- Computes the costs for the 
4 kernels (Person), 
L1 (Adjacency Matrix), 
L2 (Global Matrix), 
L3 (Comm Matrix), 
L4 (Comm Matrix), 
L5 (Comm Matrix).

**BENCHMARK METRICS**

- **Serial Performance**
  - CPU Time: [h:m:s] for each iteration
  - Time: [h:m:s] for each iteration

- **Weak Scaling**
  - Speedup: [Speedup]: [h:m:s] for each iteration

- **Strong Scaling**
  - Speedup: [Speedup]: [h:m:s] for each iteration

**COMPLEX INTERVENTIONS**

- **Node Interventions (N)**
  - Alter properties of nodes (e.g., vaccinations, etc.)

- **Edge Interventions (E)**
  - Alter properties of edges (activities)

- **Kernel 2**
  - Person

- **Kernel 3**
  - Location

- **Kernel 4**
  - Adjacency Matrix

- **Kernel 5**
  - Global Matrix

- **Kernel 6**
  - Comm Matrix

**PERFORMANCE**

- **Graph 2**
  - Boost

- **Graph 3**
  - Redirected

- **Graph 4**
  - Default

- **Graph 5**
  - Default

**REFERENCES**


**ACKNOWLEDGEMENT**

We thank the Virginia Tech Digital Signature and Simulation Laboratory, and the Virginia Tech HPC cluster for the research support and computing time. We also thank the reviewers for their valuable comments and suggestions.

**CONCLUSIONS**

- We developed a benchmarking system for evaluating the performance of different compute platforms.
- We observed that the performance of the kernels is highly dependent on the hardware configuration.
- The kernels are highly scalable, and they offer good speedup on both parallel and distributed memory architectures.
- The benchmark provides a useful tool for evaluating the performance of different architectures in the context of computational epidemiology.