2.5 A 40×40 Four-Neighbor Time-Based In-Memory Computing Graph ASIC Chip Featuring Wavefront Expansion and 2D Gradient Control

Luke R. Everson, Sachin S. Sapatnekar, Chris H. Kim

University of Minnesota, Minneapolis, MN

Single-source shortest path (SSP) problems have a rich history of algorithm development [1-3]. SSP has many applications including AI decision making, robot navigation, VLSI signal routing, autonomous vehicles and many other classes of problems that can be mapped onto graphs. Conventional algorithms rely on sequentially traversing the search space, which is inherently limited by traditional computer architecture. In graphs which become very large, this slow processing time can become a bottleneck in real world applications. We propose a time-based ASIC to address this issue. Our design leverages a dedicated hardware implementation to solve these problems in linear time complexity with superior energy efficiency. A 40×40 four-neighbor grid implements a wavefront (WF) expansion with a first-in lockout mechanism to enable traceback. Outside the array, a programmable resistive ladder provides bias voltages to the edge cells, which enables pulse shaping reminiscent of the A* algorithm [3].

Figure 2.5.1 provides a high-level schematic of the chip. The chip is structured to model a graph with a regular Manhattan grid structure. Each of the 1600 vertices have four connections to its neighbors in the cardinal directions (N, S, E, and W). The chip functions by propagating a pulse between the vertices through the edges. The time it takes for a pulse to travel from each cell is proportional to the distance, or cost, to travel through that edge. Each vertex operates autonomously; it senses and stores the direction of the input, prevents other pulses from overwriting it, and propagates it to neighboring stages. The first pulse, or set of simultaneous pulses, represents the fastest way to reach that cell. Since the first pulse is the only pulse latched in each cell, tracing the pulse chain back to the start will reveal the shortest path. Although the core was initially designed for SSP, each evaluation contains all shortest paths to the start node.

Figure 2.5.2 describes the schematic of the vertex and corresponding timing diagram. The functionality of the cell will be described with an example of two pulses arriving: North and then South. First, a global enable signal, *EN*, is asserted enabling the core. Additionally, not shown in this figure, a pulse is started from somewhere in the array. The four inputs are merged together in a detection circuit to determine if a pulse has arrived in the cell. In the example, *IN<N>* will flow through and latch P_{IN}, or Pulse Input. P_{IN} is compared with the input from each of the four directions. If P_{IN} is asserted and the input is not from the direction that asserted P_{IN}, Pulse Latch, in this example *PL<{S, E, W}>*, will assert. *PLb* is connected to the SRAM DMA, which will flip the *IP<{S, E, W}>* SRAM that will be read out after evaluation during the path traceback. *IP<3:0>* shows how the vertex stores the input pulse. Initially all four bits are cleared. The bits that do not correspond to the first input are flipped with the DMA circuit. This notation corresponds to the Colormap Readout Key in Fig. 2.5.4. Finally, the pulse is propagated to the neighbors that were not responsible for latching the cell and have a connection stored in their respective local SRAMs. Connections are predetermined based on the description of the graph at runtime.

Figure 2.5.3 shows the edge schematic. Each edge consists of one current-starved inverter, one standard inverter, and four binary weighted bits of capacitor loading. The delay is modulated by the 4b weight stored locally in the SRAM and the voltage bias applied to each branch of the first inverter. The delay of each unit, defined as a vertex and an edge, is shown for different bias voltages and sets of weights. The top left colormap shows the simulated delays for each unit with the biases V_{X3} and V_{Y3} applied at the closed switches. V_{H3} and V_{V3} are larger than {V_{H1}, V_{H2}} and {V_{V1}, V_{V2}}. V_{H3} and V_{V3} are applied through a transmission gate controlled by the scan bits. All other points are held open which yields a linear voltage increase from the edges at each stage until target stage 30, illustrated in the lower left figure.

Figure 2.5.4 details examples from the SSP application and collision avoidance (CA), or minimal edge-effect problem. CA is useful if there is an incentive to avoid obstacles, such as self-driving cars or drone navigation. In this application, the pulse is started simultaneously from the sides of the obstacles. Where the WFs meet, shown in white, signifies the path that maximized the distance between the obstacles. In SSP, we show two outputs generated from the same map under different conditions. In this application, a map is shown with blockages shown as black blocks. The WF is initiated in the upper left and it propagates down and across the core. The top figure does not have a voltage gradient applied and each edge has the same weight. This gives the WF a very regular pattern as it traverses the map. The bottom figure has a voltage gradient applied that is weakest in the top-left corner and strongest in the bottom-right corner. The key difference between the two outputs is shown in lower-right, where the yellow cells signify that the upper half supersedes the lower half. Without the applied bias the pulses nearly arrived at the same time (barrier between blue and orange), however, with the voltage gradient, the pulse traverses the blockages faster.

This ASIC is not constrained to solving problems that conform to a 40×40 grid. Fig. 2.5.5 highlights the scalability of this core via a four-core example with a single blockage spanning three cores. First, core 0 is evaluated and the WF reaches cores 3 and 1, both at two points, which will be used to start the pulse in subsequent evaluations. Next, core 3 is evaluated, but the pulse does not uncover much of the map due to the complete vertical blockage. After this, core 1 is evaluated and it contacts core 2 in two locations: the bottom right corner and directly above the blockage. These two cells are used to start the evaluation for core 2. Finally, it is revealed that core 2 and core 3 share an unexplored boundary, and core 3 is re-evaluated to fully uncover the obstacle. It is important to note that multicore still solves the SSP problem in linear time as the grid size increases.

Motivated by [1], we set out to recreate the optics experiments in Manhattan geometries to illustrate the utility of our hardware architecture. Shown in the lower half of Fig. 2.5.6 is a sampling of the core readout at different time points during a single evaluation of a two-slit experiment. The pulse starts to the left and above of the first slit. The WF traverses the first block and then passes into the second block. Next, the WF reaches the middle of the second boundary and then spreads out until it reaches the two-slits and generates two leading WF. This behavior mimics what is reported in [1] and has interesting consequences for future physical explorations of novel applications with low-power CMOS. The comparison table in Fig. 2.5.6 is mainly used for general comparisons, since to our knowledge, this is the first ASIC for graph traversal. The peak power is quoted when all 156 perimeter vertices are evaluating corresponding to the pulse originating from the center, equating to 183.1µW/vertex. 55% of the power is due to SRAM access storing the pulse information in-situ. Compared to state-of-the-art FPGA [4], µProcessor, CPU, and GPU [5] implementations, our core has roughly 6 orders of magnitude superior energy efficiency. In this work, we described an in-memory computing ASIC graph processor and highlighted the versatile applications and scalability of the proposed chip.

Acknowledgements:
This research was supported in part by the National Science Foundation under award number CCF-1763761.

References:
[1] C. Y. Lee, "An Algorithm for Path Connections and Its Applications", *IRE Transactions on Electronic Computers*, vol. EC-10, no. 3, pp. 346-365, Sept. 1961.
[2] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numerische Mathematik,* vol. 1, pp. 269–271, 1959.
[3] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," in *IEEE Trans. on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100-107, 1968.
[4] S. Zhou, C. Chelmis and V. K. Prasanna, "High-Throughput and Energy-Efficient Graph Processing on FPGA," *IEEE International Symp. on Field-Programmable Custom Computing Machines*, pp. 103-110, 2016.
[5] Y. Zhou and J. Zeng, "Massively Parallel A* Search on a GPU," Conference on Artificial Intelligence (AAAI), pp. 1248-1254, 2015.

 978-1-5386-8531-0/19/$31.00

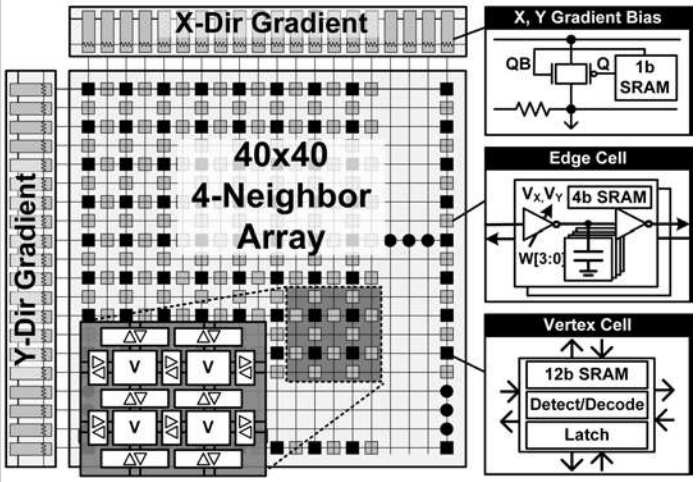


Figure 2.5.1: Proposed 40×40 graph ASIC chip for solving single-source shortest path problems based on 2-dimensional wavefront expansion.

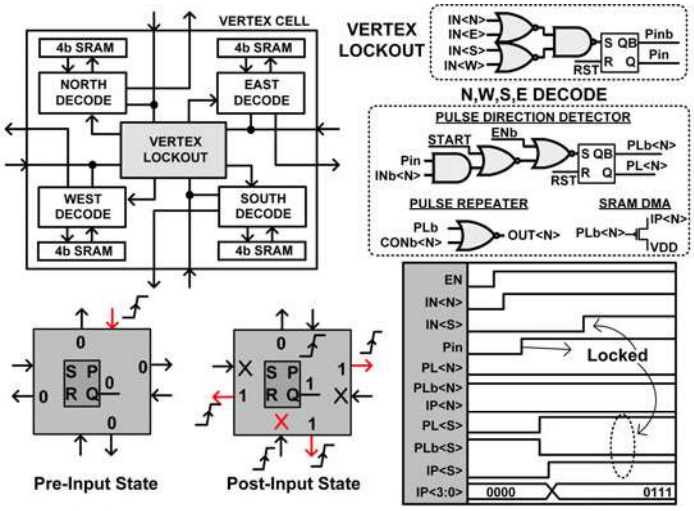


Figure 2.5.2: Details of the vertex cell, including the implementation of lockout and decode logic. Timing diagram shows lockout procedure with pre- and post-input state of vertex cell.

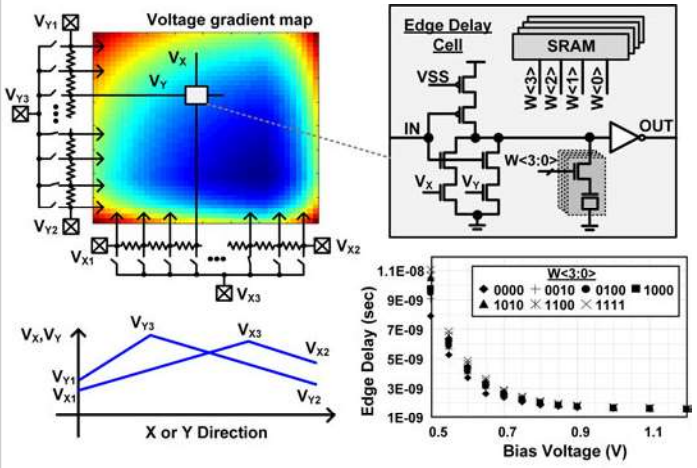


Figure 2.5.3: Voltage gradient map for proposed graph chip and schematic of edge cell with digital and analog delay control options; analog control voltage profiles in x and y directions; measured delay vs. bias voltage for different digital capacitor loads.

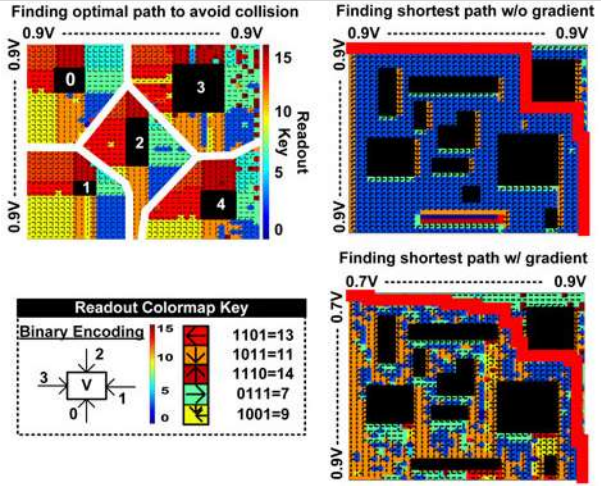


Figure 2.5.4: Test chip measurement results. Left: example of Collision Avoidance application where black boxes represent obstacles and white lines denote optimal path to avoid obstacles. Right: A* shortest path application shows routes with and without voltage gradient to accelerate wavefront.

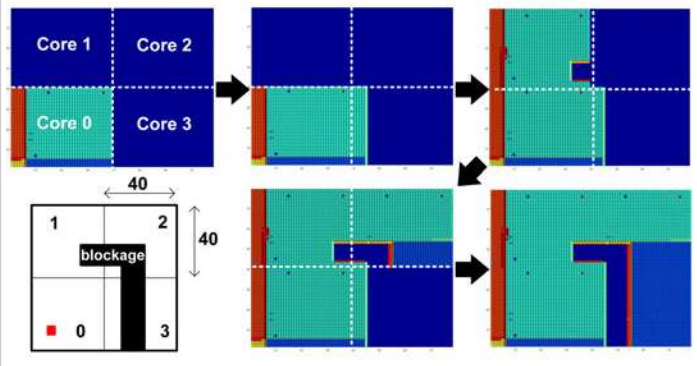


Figure 2.5.5: Four-core example shown to highlight scalability of the graph ASIC for larger maps. 4 cores can be stitched together or alternatively, a single core can be reused 4 times. The start node of next core is determined by the point of first impact on shared edge. Even with blockages spanning multiple cores, the framework can successfully traverse the entire map.

Architecture	This Work	FPGA [4]	µProcessor	CPU [5]	GPU [5]
Product	ASIC	Xilinx Virtex	ARM Cortex-M0	Intel Xeon E5630	NVIDIA Tesla K20c
Technology	65nm	20nm	40nm	32nm	28nm
Voltage	1.2V	-	1.1V	0.7-1.35V	-
Peak Power	26.4mW	24.22W	127µW	20W/core	225W
Throughput [MTEPS]	559	731	$5.34*10^{-4}$	0.83	9.0
Energy per Node	0.328pJ*	33nJ	89.1nJ	24.1µJ	25µJ
Normalized Energy	1x	10^5	$2.7x10^5$	$1.19x10^6$	$2.3x10^7$

*55% from SRAM Program (does not include cache access energy), Energy/Node=Unit Delay*Unit Power
MTEPS = Million Traversed Edges Per Second

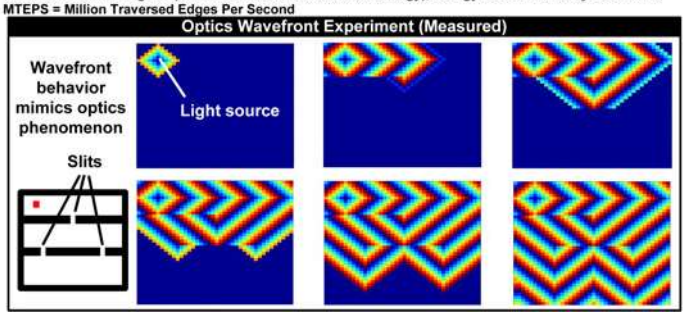
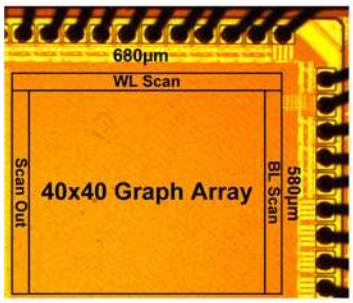


Figure 2.5.6: Comparison table shows 5 orders of magnitude improved energy efficiency over traditional implementations (upper). Measured results of scientific computation (i.e. optics wavefront propagation in two-slit experiment) (lower).



Applications	A* shortest path, obstacle avoidance, scientific computation (optics)
Technology	65nm LP CMOS
Architecture	Time-based
# of Vertices	1600
# of Edges	6400
Edge Resolution	4b + Analog Gradient
Voltage	1.2V
Peak Power	26.4mW
Delay per Node	1.79ns @ [V_B=.9V, V_{DD}=1.2V]
Power per Node	183.1µW
Energy per Node	0.238pJ

Figure 2.5.7: Die photo and chip summary.

978-1-5386-8531-0/19/$31.00 ©2019 IEEE