Accelerated Biological Meta-Data Generation and Indexing on the Cray XD1

Eric Stahlberg, Joseph Fernando, Kevin Wohlever
(The Ohio Supercomputer Center)

Jeff Doak
(Cray, Inc)

May 18, 2005

OSC SPRINGFIELD

ASC

First Things First:
Nothing Gets Done Without the Help of Others

- Sherry Sun (OSC Springfield)
- Pete Carswell (OSC Columbus)

Outline

- About OSC
- Project context and motivation
- Project description
- Details
- Conclusions
- Looking and moving ahead

About The Ohio Supercomputer Center

- Located in Columbus, Ohio
- The Ohio Supercomputer Center was established in 1987 to position Ohio universities and industries at the forefront of computationally intensive research, development, engineering and networking.
- Delivering high performance computing and high capacity optical network for production and research

OSC: Ohio's Answer to High Performance Computing Innovation

OSC holds a unique niche among supercomputing centers

- ***Provides a reliable high performance computing and communications infrastructure***
- ***Serves a diverse, statewide/regional community including education, academic research, industry, and state government***
- ***Is a key enabler for the state's aspirations in advanced technology, information systems, and industry***
- ***Acts as a catalytic partner of Ohio universities and industries focusing on new research and business opportunities***

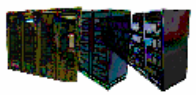
OSC Strategic Directions

- *OSC provides a reliable high-performance computing and communications infrastructure for a diverse, statewide/regional community including education, academic research, industry, and state government.*

- *OSC strives to be in the forefront of computational research in order to act as a key enabler for the state's aspirations in advanced technology, information systems, and advanced industries, and,*

- *OSC acts as a catalytic partner of Ohio universities and industries to enable Ohio to compete for international, Federal, and State funding, focusing on new research and business opportunities in:*
  - *Basic Sciences*
  - *Bioinformatics*
  - *Advanced Manufacturing*
  - *Agriculture*
  - *Data-centric applications*
  - *Materials*
  - *Modeling, Testing and Instrumentation*

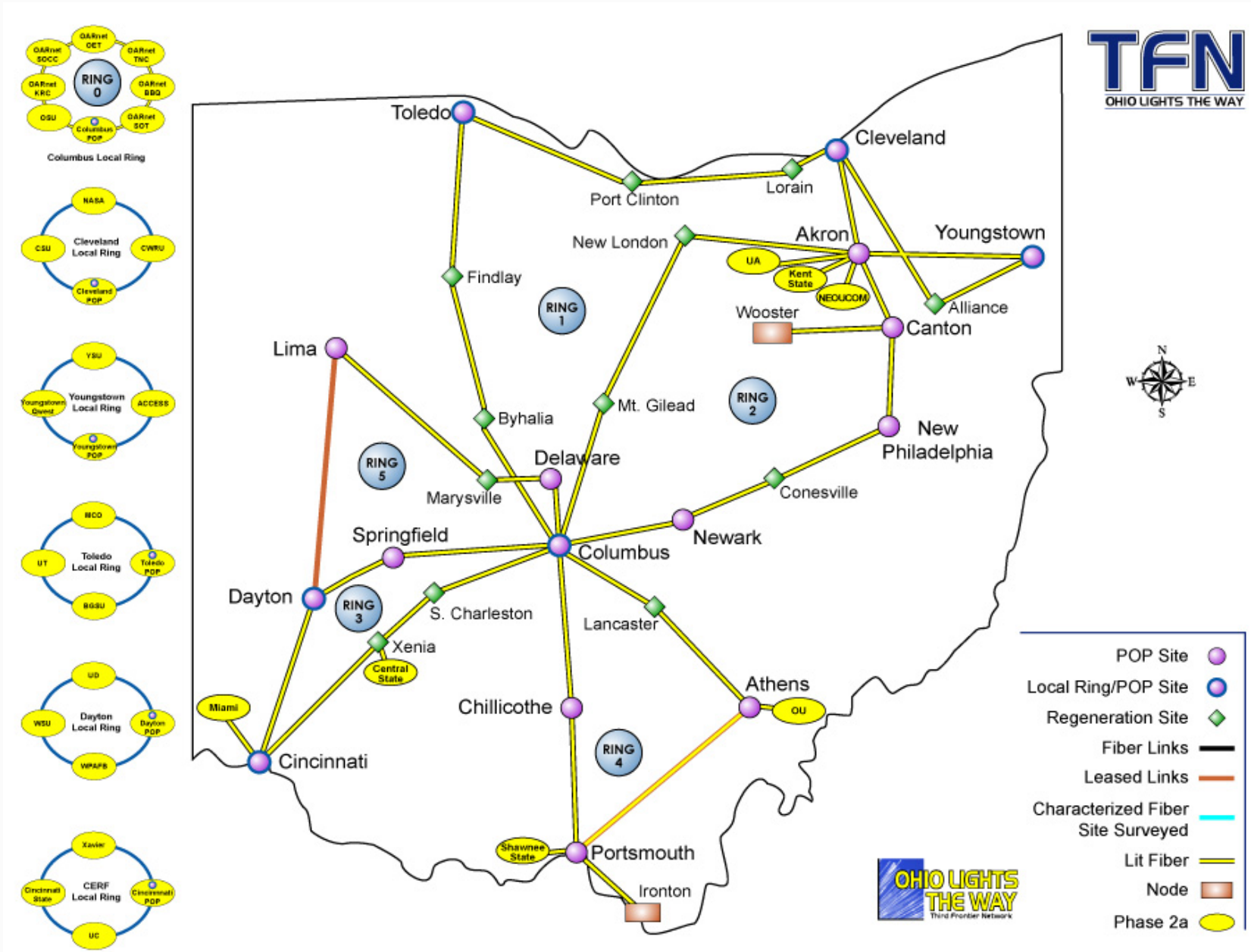
Production Computing Environments

System	Processors	Total Memory	Peak Performance	Special Purpose
	300 900 MHz Intel Itanium2	752 GB	1.1 TF	IA-64 Distributed parallel and serial applications
	512 2.4 GHz Intel Pentium 4 Xeon	1024 GB	2.5 TF	IA-32 Distributed parallel (128 nodes with Infiniband interconnect) and serial applications
	100 1.53 GHz Athlons	100 GB	.31 TF	BALE Visualization cluster with NVIDIA Quadro4 900 XGL graphics board / node
	192 550 MHz Pentium III Xeon, 144 733 MHz Itanium, 256 1.4 GHz AMD Athlons	736 GB	1.92 TF	Cluster Ohio -- Distributed computational grid at 15 institutions around the state
Alliance Grid Testbed (AGT) Cluster	52 2.2 GHz Pentium 4 Xeon	52 GB	.23 TF	One of 8 sites around the country for computational grid testing and deployment
	16 800 MHz vector processors	64 GB	.20 TF	CRAY X1 AC
	48 900 MHz UltraSPARCIII	48 GB	.04 TF	SUN COE with Time logic boards for specialized bionformatics apps
	32 900 MHz Intel Itanium 2	64 GB	.20 TF	SGI Altix – Linux IPF Shared Memory System

What is OSC Doing in Springfield, Ohio?

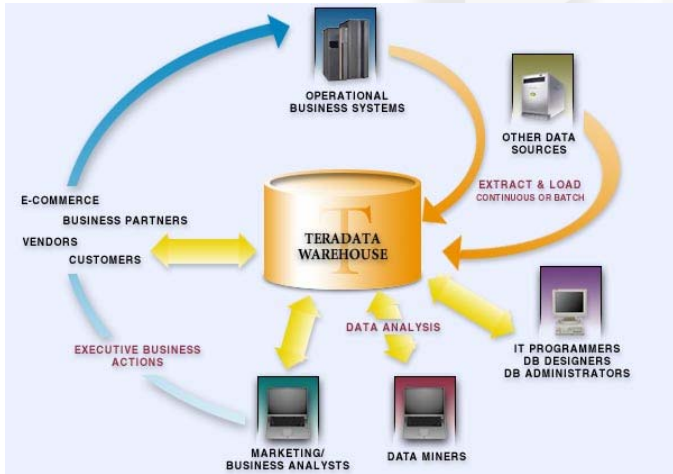
- Centrally located between Columbus and Dayton along the Third Frontier Network
- Work with organizations in the area (LexisNexis, NCR, Wright-Patterson Air Force Base) and nationally on *data intensive computing* challenges
- Working with the DOE ASCI program to investigate a number of HPC issues

OSC OARnet: TFN Overview



Leveraging the Springfield Center

- ***HPC Vendors***
  - ✓ Include facility to easily construct and evaluate new HPC compute and mass storage architectures (DARPA HPCS, DOE, HECRTF)
  - ✓ Develop benchmarks to test options for production performance on DOE (and general scientific) applications
- ***Pursue Common Challenges Among Business and Science***
  - ✓ Data warehousing and distributed data access
  - ✓ Data mining for highly heterogeneous data types (numerical, images, sounds)
- ***Cooperation with Local Universities***
  - ✓ Hands-on training for students
  - ✓ Virtual laboratories for educational experiences



Dynamic Data Computing Environment at Springfield

- Cray XD1 with FPGA
- Apple Xserve G5
- Cray X1
- 1 Gigabit interconnect
- Local mass storage
- Remote massive storage at Columbus

Project Overview

Objective: To implement a highly portable hashing algorithm for bioinformatics sequences

Motivations

- To explore the XD1 FPGA development environment
- To determine portability of algorithm to FPGA environment
- To determine optimal algorithm implementation for XD1 FPGA environment
- To create a core for further indexing of bioinformatics sequences
- To build a foundation for comparative bioinformatics applications

Challenges Addressed

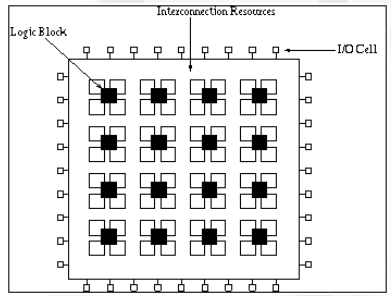
- No consistent identifier for sequences
  - Genbank has their own
  - TIGR has their own
  - Sequences may differ undetectably
- Descriptive headers are arbitrary and mutable
- No ability to integrate non-published data
- Difficult to always associate full sequence with derived data across datasets

Algorithm Design Goals

- Highly portable
  - Ease of implementation independent of platform
- Normalizing
  - Incorporates essential information only
  - Convention independent
- Usable
  - Readable, consistent, searchable
  - Embedded summary meta-data
- Evenly distributed results
  - Minimize collisions, maximize uniqueness
- Self-validating
  - Consistency checks for original data and derived meta-data

OSC First Experiences with Production FPGAs: TimeLogic DeCypher

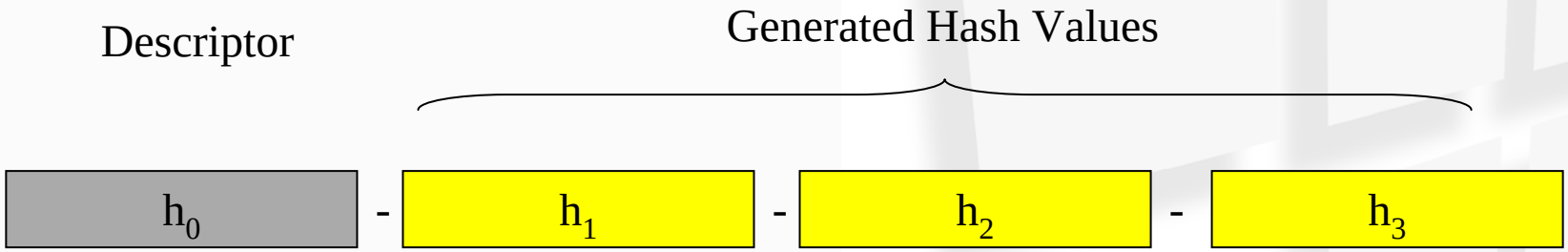
- FPGA for production bioinformatics
  - Field Programmable Gate Array
  - Deployed 2002 in SunFire 6800 COE systems
  - High throughput for searching/scanning data
- Available Algorithms
  - Tera-BLAST (N,P,X, TN, TX)
  - Smith-Waterman
  - HMM searches and queries
  - Profile scan
  - Custom target building
- Frustratingly closed to custom algorithm development



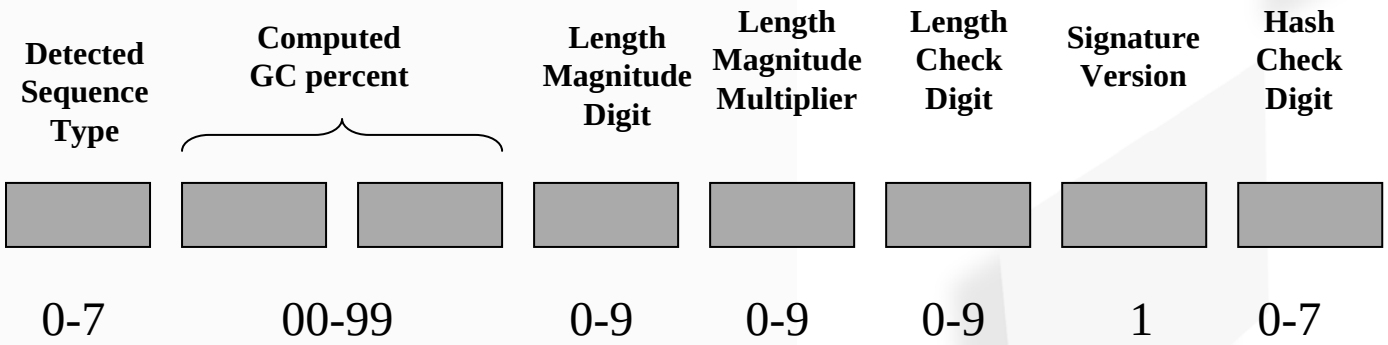
Resources Employed for this Project

- Cray XD1
- Xilinx Virtex-II Pro FPGA
- Xilinx ISE 6.3i development environment
- Cray X1
  - for early algorithm validation

BXID Layout



Descriptor (field level detail)



Sequence Types

Type 0 – Undefined, no valid character available

Type 1 – DNA, no undefined positions (Only ACTG present in sequence)

Type 2 – RNA, no undefined positions (Type 1 with U present)

Type 3 – DNA with undefined positions (Type 1 with N present)

Type 4 – RNA with undefined positions (Type 1 with both UN present)

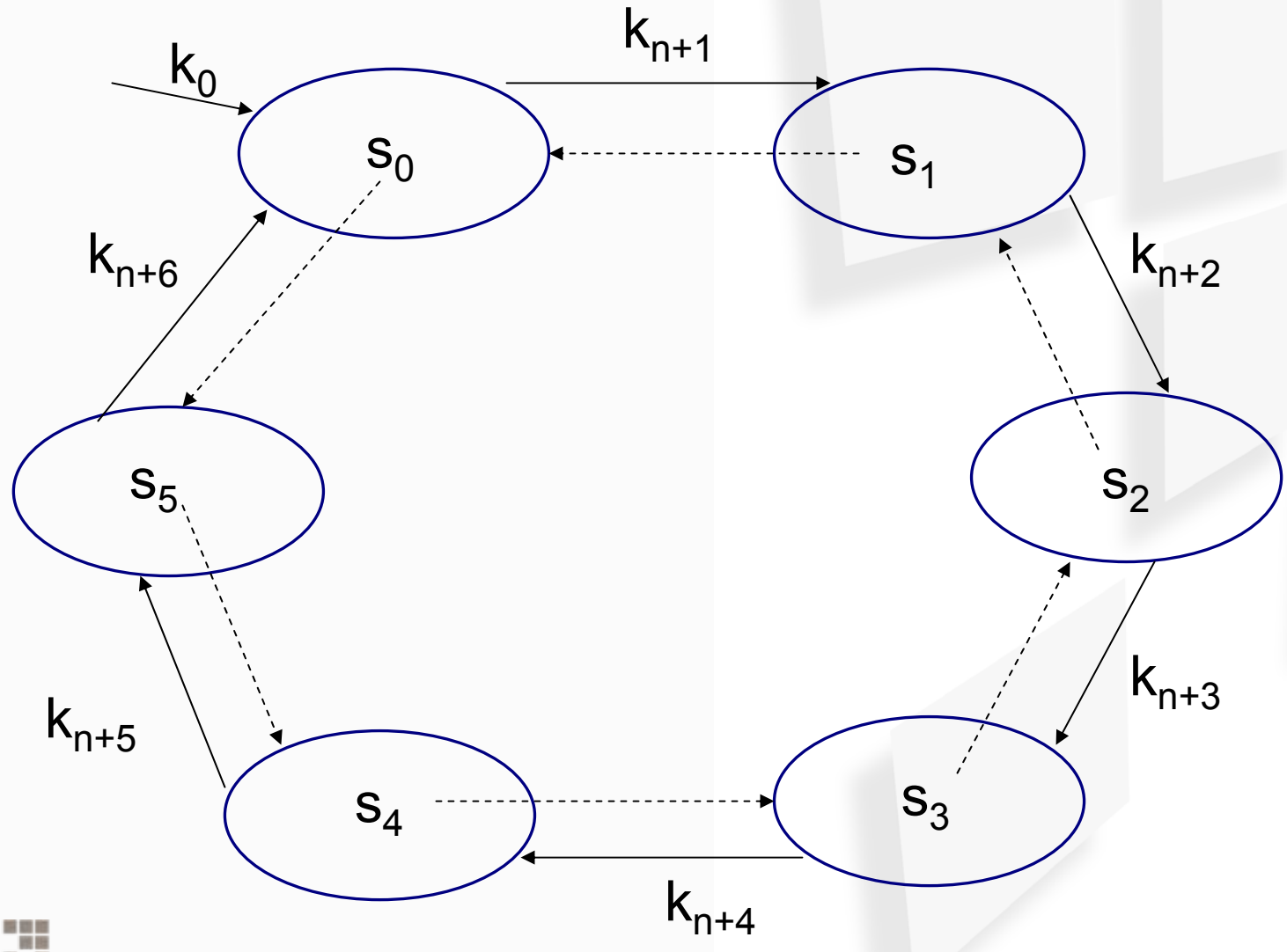
Type 5 - Inconclusive for DNA/RNA/Protein determination (Any of XIQRYDOBS present in sequence)

Type 6 – Protein (Any of EFHJKLMPVWZ present in sequence; supersedes type 5)

Type 7 – Protein with position gaps (any '-' present in sequence; supersedes type 6)

(All other characters in sequence are considered invalid and ignored)

FSA Hash Generation



Algorithm Details

1. Initialize ***seed*** = 255
2. Initialize uppermask = 2^23
3. Initialize lowermask = 2^8
4. Initialize maxsize = 32767
   Initialize ***si*** = ***seed***, ***i*** an element of {0,1,2,3,4,5}
5. Initialize g or c character count, ***gc***, and length, ***l***, to zero
6. Initialize sequence type state variable, ***q*** = 0 (undefined state)
7. For each character, ***c***, in sequence
8. Assign Index value, ***k***, A defined as position 1.
9. ***k*** = index(***c***, ' ACTGUNXIQRYDOBSEFHJKLMPVWZ*')
10. If ***k*** in range (***k*** > 0)
11. Increment length**, *l***
12. Update sequence type, ***q***
13. Update ***gc*** count
14. Update stage value, ***si,*** as follows
15. ***i*** = mod(***l*** ,6)
16. ***f*** = mod(***l*** + 1, 6)
17. ***si*** = mod(***seed*** + ***si***/2 + ***sf*** /2 + ***k*** + ***k*** * mod(***l***, 1021)), maxsize)
18. ***si*** = mod(***si*** ^2, uppermask) / lowermask
19. Composite final hash function values
20. ***h1*** = ***s0*** * 2**16 + ***s1***
21. ***h2*** = ***s2*** * 2**16 + ***s3***
22. ***h3*** = ***s4*** * 2**16 + ***s5***
23. Compute descriptor fields
24. length check digit, ***lc*** = ***l*** % 10
25. length magnitude, ***lm*** = min(floor(log10 ***l***) , 9)
26. length magnitude multiplier, ***lmm*** = ***l*** / ***lm***
27. set current algorithm version ***v*** = 1
28. gc percentage, ***p*** = min((***gc*** * 100) / ***l*** , 99)
29. compute check digit for generated hash fields,
30. ***x*** = mod((Σ(***i***=1,3) Σ(***k***=0,4) mod(***hi***,102***k***+2)/102***k***),8)
31. Composite descriptor fields in human identifiable form
32. ***h0*** = ***q*** * 10,000,000 + ***p*** * 100,000 +
33. ***lm*** * 10,000 + ***lmm*** * 1,000 + ***lc*** * 100 +
34. ***v*** * 10 + ***x***

Initialize

State Updates

Composite Keys

Descriptor Prep

State Update Formula

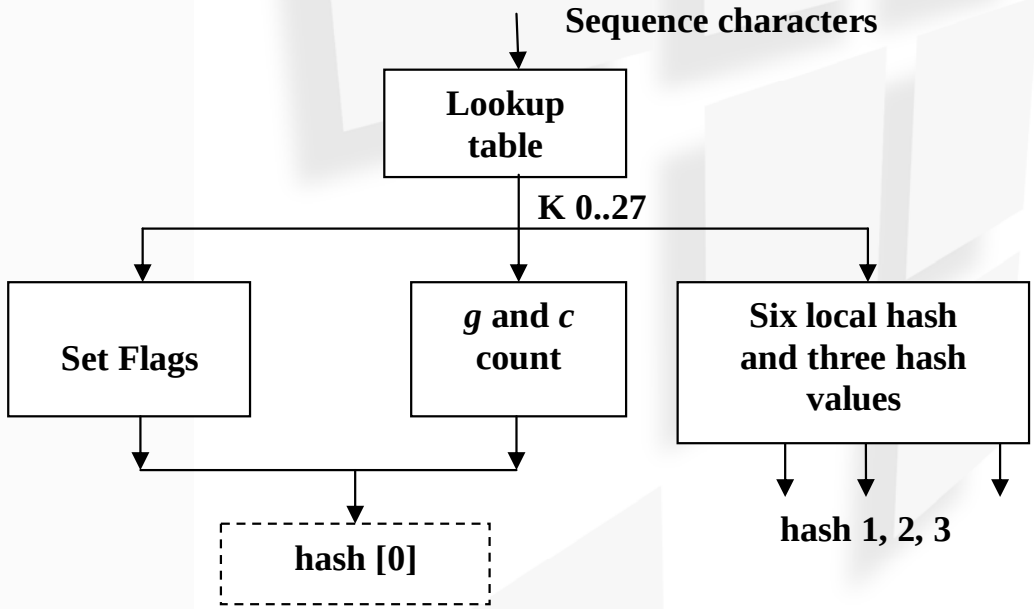
si = mod(***seed*** + ***si***/2 + ***sf*** /2 + ***k*** + ***k*** * mod(***l***, 1020)), maxsize)

si = mod(***si*** ^2, uppermask) / lowermask

si = h[l]
sf = h[(l+1)% 6]
i = number of supported characters in string thus far
k = normalized character index
seed = 255
maxsize = 32768
uppermask = 2**23
lowermask = 2**8

XD1 Implementation

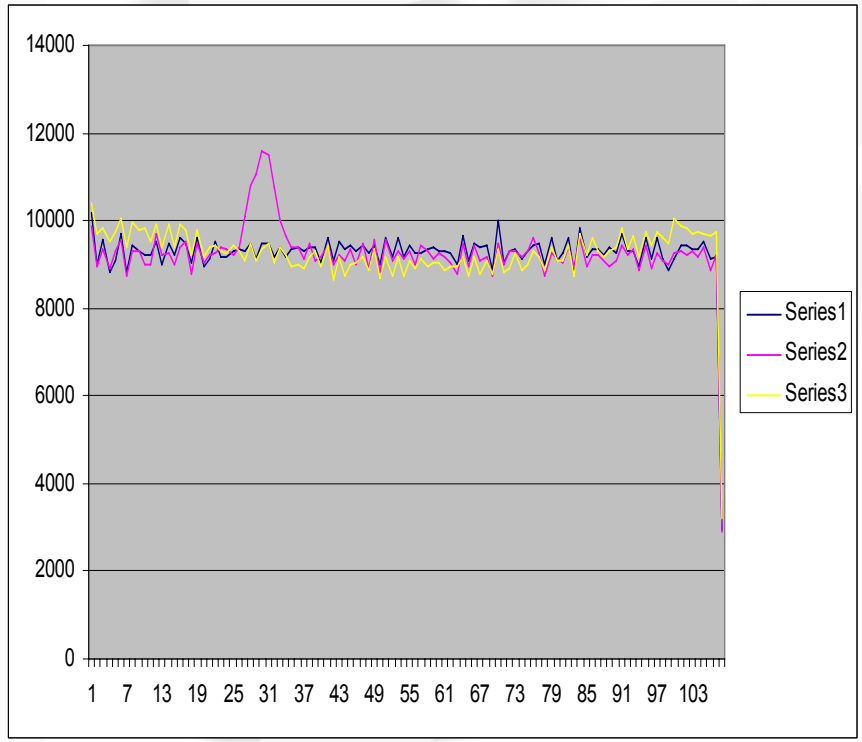
- Solid Boxes implemented on FPGA
- Dashed box implemented on Opteron



Overall Algorithm Effectiveness

- Random sequences
  - Evaluations conducted on X1
- DNA (4 bases only: ACTG)
  - 500 million sequences
    - lengths 20 -100
    - 16 duplicates and 2 collisions
  - 100 sequences l
    - length 1-3 nucleotides
    - 80 duplicates, 0 collisions
  - Others originally attempted had no duplicates or collisions
  - Stress testing reveals some collisions
- Proteins (ranges varying 20-3000)
  - 10 million sequences
    - No collisions
  - 100 million sequences
    - No collisions
  - 500 million sequences
    - No collisions
- Actual data reveals only duplicates

DNA Random Sequence Results
(1 million sequences; 20-100 bp)



Histogram bucket size = 10M

Challenges and Limitations

- Challenge 1 – Communication rate to FPGA – still 6 million characters per second not too bad
- Challenge 2 – Totally on-board implementation – developing self-contained core
- Challenge 3 -- 44 character signature sometimes too long
- Challenge 4 – A hash is still a hash. Collisions may occur in the domain being analyzed
- Challenge 5 – Code distribution and availability

Conclusions

1. FPGA implementation feasible for BXID algorithm with slight modifications

2. Relatively efficient core was developed

3. Algorithm demonstrates desired attributes

Future Efforts and Directions

- Make available through OBL – extensions to the foundations created by CBL/PCBL
- Finalize reference website for file processing and sequence tests/searches using algorithm
- Deeper integration in sequence related projects
  - www.plantmicrobeinteractome.org
  - www.coiled-coil.org
  - Others…
- Implement new communication subsystem developed at OSC

Addressing Common FPGA Challenges

- ***A cross-cutting organization fostering FPGA utilization in high-level applications***
- ***Focusing on challenges of***
  - *Interoperability*
  - *Communication*
  - *Development environments*
  - *Common practices and standards*
- ***Forums initiated in six key areas***
- ***Organizational and participation structure presently under development***
- ***Visit www.openfpga.org for more information, updates and to get involved***

Thank you…

Further questions?

Summary Contact Information:

Joseph Fernando – fernando@osc.edu – FPGA implementation

Eric Stahlberg – eas@osc.edu – Algorithm development/OBL

Jeff Doak – jdoak@cray.com – Algorithm validations/OBL

Kevin Wohlever – kevin@osc.edu – Direction and management