山东大学
SHANDONG UNIVERSITY

XSW: Accelerating Biological Database Search on Xeon Phi

School of Computer Science and Technology
Shandong University, China

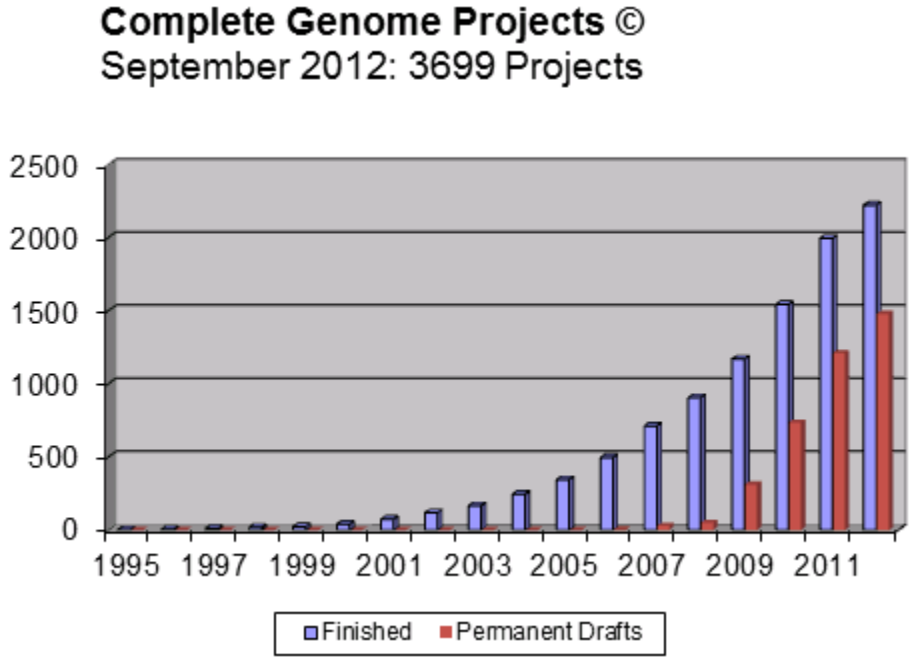
May, 2014

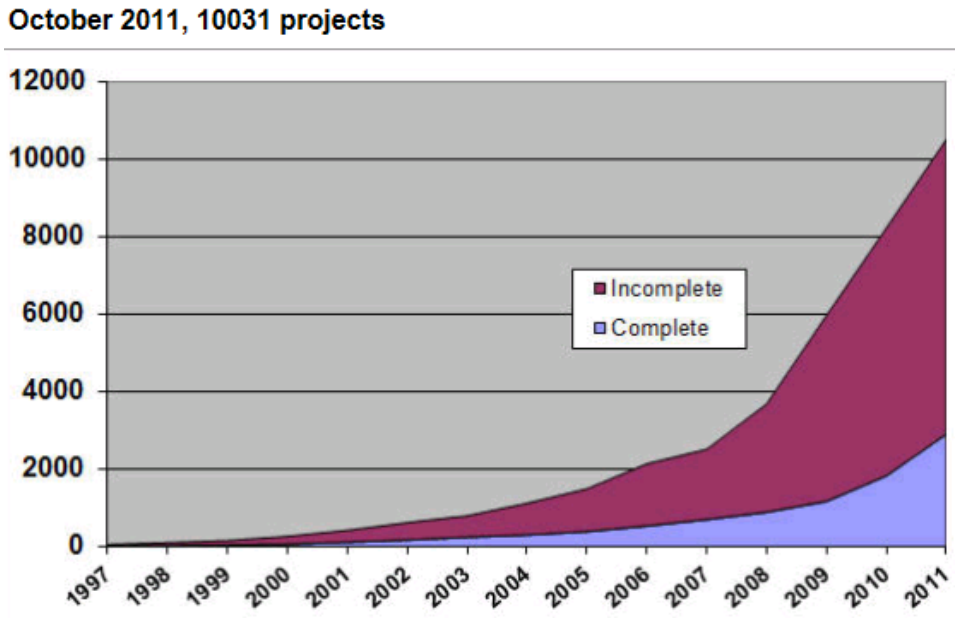
Contents

- Motivation
- Smith-Waterman Algorithm
- Mapping onto the Xeon Phi
- Performance Evaluation
- Conclusions and Future Work

Bio DB Scanning on Xeon Phi: Motivation(1/3)

- Genome sequence databases are growing rapidly
- Growth rate will continue, since multiple concurrent genome projects have begun, with more to come
  - 3699 genomes published
    (http://www.genomesonline.org/ (Sep, 2012))
  - 10031 genome sequencing projects ongoing





Bio DB Scanning on Xeon Phi: Motivation(2/3)

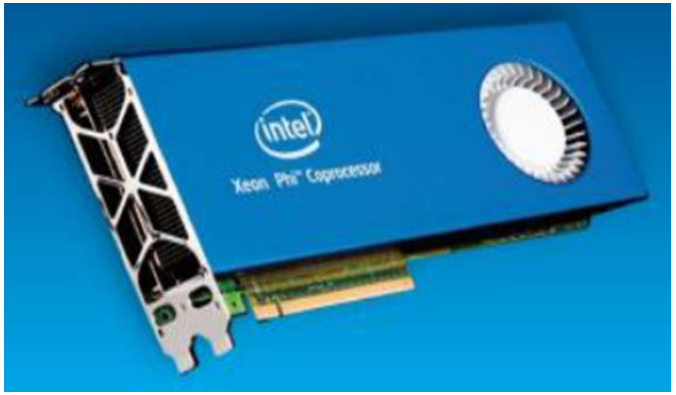
- Discovered sequences need to be analyzed/annotated
- Typical operations
  - Database Scanning
  - Multiple Sequence Alignment
  - Hidden Markov Model training and scoring
  - Computing Evolutionary Trees

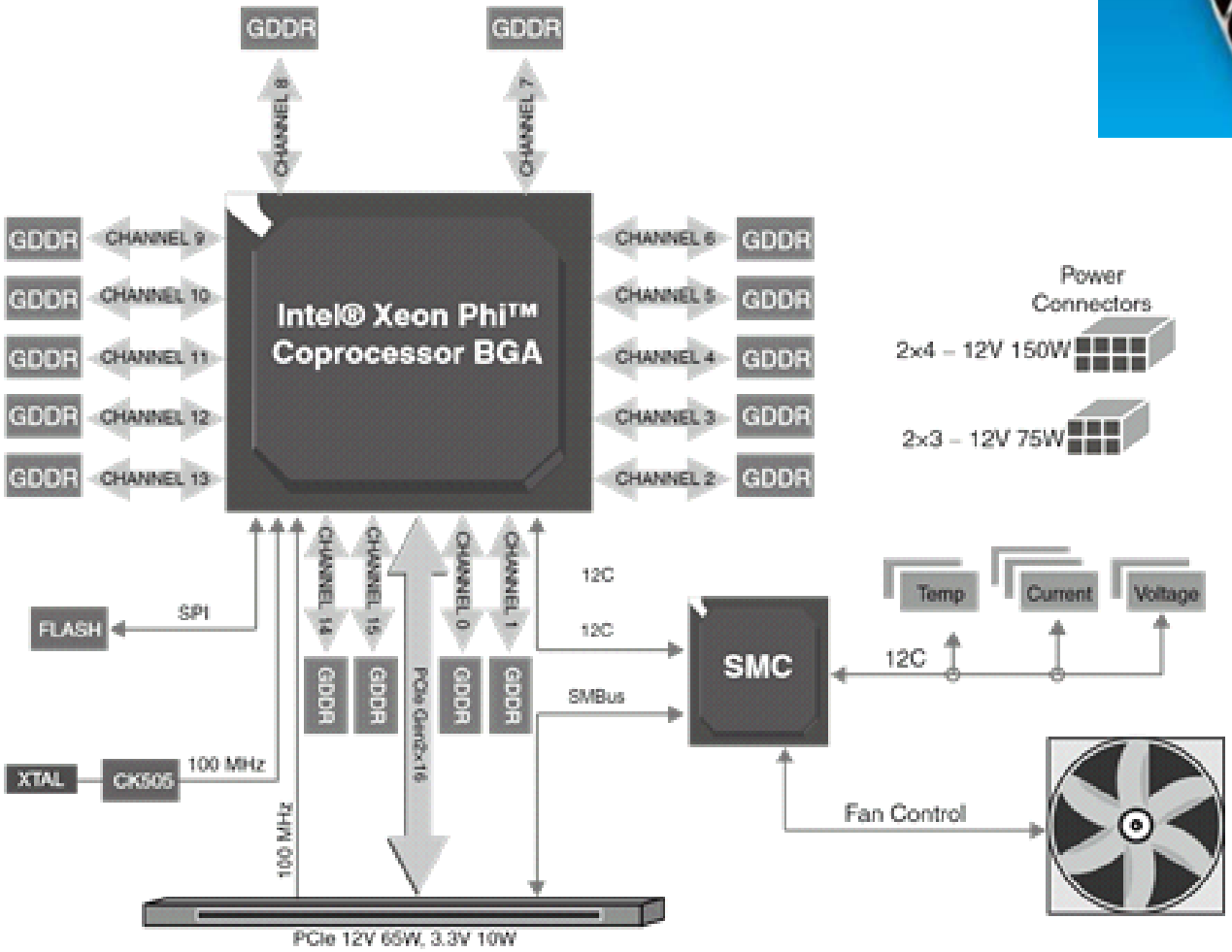
Type of data	Doubling time (year)
Genome databases	1.44
PC speed (number of transistors)	2.09
Supercomputer speed (LINPACK)	1.04

- Establishes the need for High Performance Computing (HPC)
- HPC Alternatives
  - –Coarse-grained (e.g. Clusters, Grids, Clouds)
  - –Fine-grained (e.g. FPGAs, GPUs)

Bio DB Scanning on Xeon Phi: Motivation(3/3)

- High performance/price ratio
- Easy programming





Smith-Waterman Algorithm

- Performs an exhaustive search for the optimal local alignment of two sequences.
- Aligning *S*1 and *S*2 of length *l*1 and *l*2 using Recurrences:

$$H(i,j) = \max \begin{cases} 0 \\ E(i,j) \\ F(i,j) \\ H(i-1,j-1) + Sbt(S1_i, S2_j) \end{cases} \quad ,1 \le i \le l1, 1 \le j \le l2$$

$$H(i,0) = E(i,0) = 0$$
$$H(0,j) = F(0,j) = 0$$

$$E(i,j) = \max \begin{cases} H(i,j-1) - \alpha \\ E(i,j-1) - \beta \end{cases}, \quad F(i,j) = \max \begin{cases} H(i-1,j) - \alpha \\ F(i-1,j) - \beta \end{cases}$$

Smith-Waterman Algorithm

Align S1=*ATCTCGTATGATG* S2=*GTCTATCAC*

$$Sbt(x,y)=\begin{cases}2 \text{ if } (x=y)\\ -1 \text{ else}\end{cases}$$

α=1, β=1

	∅	A	T	C	T	C	G	T	A	T	G	A	T	G
∅	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	2	1	0	0	2	1	0	2
T	0	0	2	1	2	1	1	4	3	2	1	1	3	2
C	0	0	1	4	3	4	3	3	3	2	1	0	2	2
T	0	0	2	3	6	5	4	5	4	5	4	3	2	1
A	0	2	2	2	5	5	4	4	7	6	5	6	5	4
T	0	1	4	3	4	4	4	6	5	9	8	7	8	7
C	0	0	3	6	5	6	5	5	5	8	8	7	7	7
A	0	2	2	5	5	5	5	4	7	7	7	10	9	8
C	0	1	1	4	4	7	6	5	6	6	6	9	9	8

$$H(i,j)=\max\begin{cases}0\\ H(i-1,j)-1\\ H(i,j-1)-1\\ H(i-1,j-1)+Sbt(S1_i,S2_j)\end{cases}$$

```
A T C T C G T A T G A T G
| |   | | | ⋮ |
G T C - T A T C A C
```

AsHES 2014

Parallel SW on Multi-core CPU

A. Wozniak(1997)

- Using video-oriented instructions to speed up sequence comparison, Bioinformatics, Vol. 13 Issue 2, pages 145-150, 1997. (Impact Factor: 5.323)

T. Rognes, E. Seeberg(2000)

- Six-fold speed-up of Smith-Waterman sequence database searches using parallel processing on common microprocessors, Bioinformatics, Vol. 16 no. 8, pages 699-706, 2000. (Impact Factor: 5.323)

Michael Farrar(2007)

- Striped Smith-Waterman speeds database searches six times over other SIMD implementations, Bioinformatics, Vol. 23 no.2, pages 156-161, 2007. (Impact Factor: 5.323)

T. Rognes(2011)

- Faster Smith-Waterman database searches with inter-sequence SIMD parallelisation, BMC Bioinformatics, 12:221, 2011. (Impact Factor: 3.02)

Parallel SW on Coprocessors

T. Oliver, etc.(2005)

- Reconfigurable architectures for bio-sequence database scanning on FPGAs, IEEE Trans. Circuit Syst. II, vol. 52, no. 12, pp. 851-855, 2005. (Impact Factor: 1.327)

W. Liu, etc.(2007)

- Streaming Algorithms for Biological Sequence Alignment on GPUs, IEEE Transactions on Parallel and Distributed Systems, vol. 18, no. 9, pp. 1270-1281, 2007. (Impact Factor: 1.733)

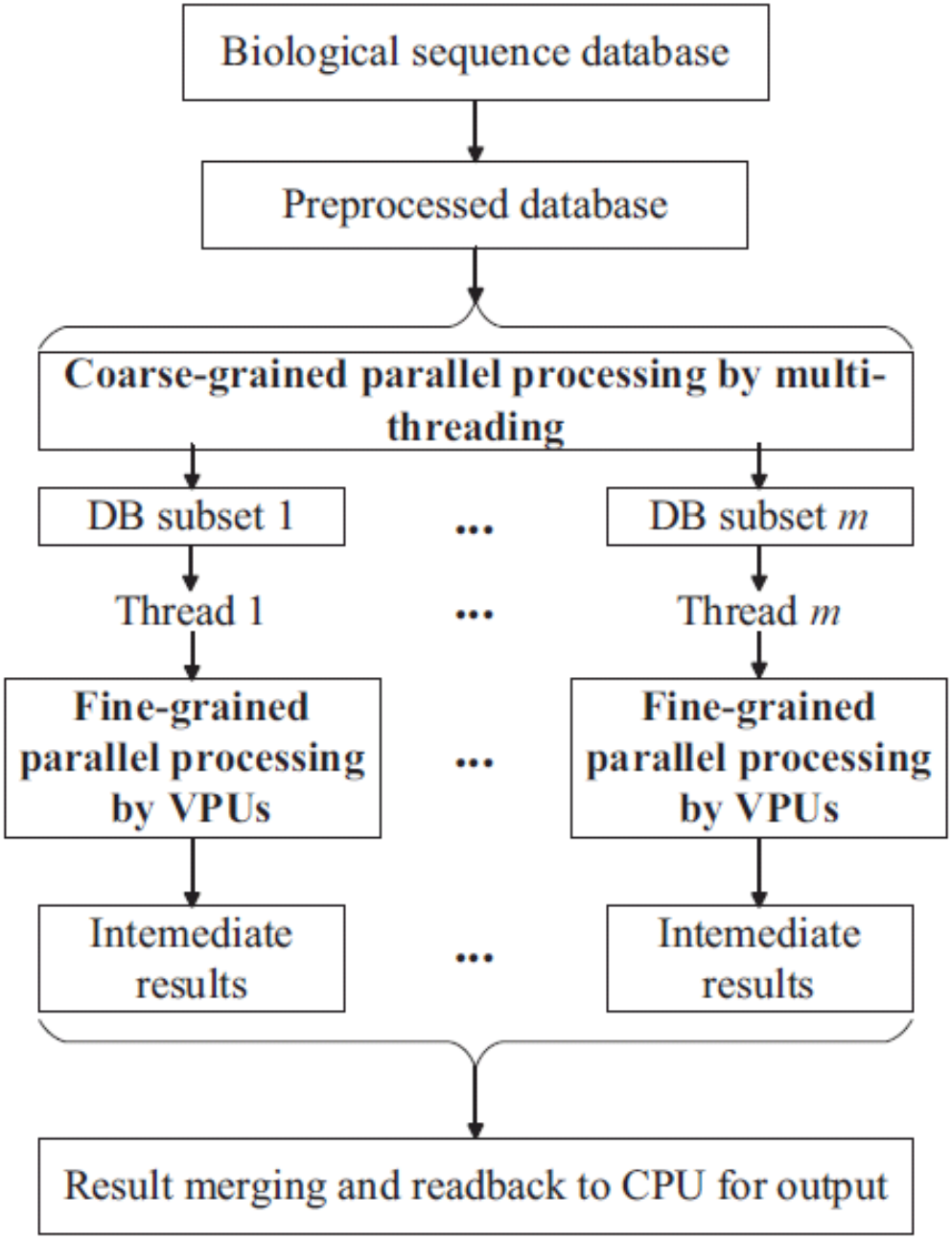
A. Wirawan, etc. (2008)

- CBESW: Sequence Alignment on the Playstation 3, BMC Bioinformatics, 9:377, 2008. (Impact Factor: 3.02)

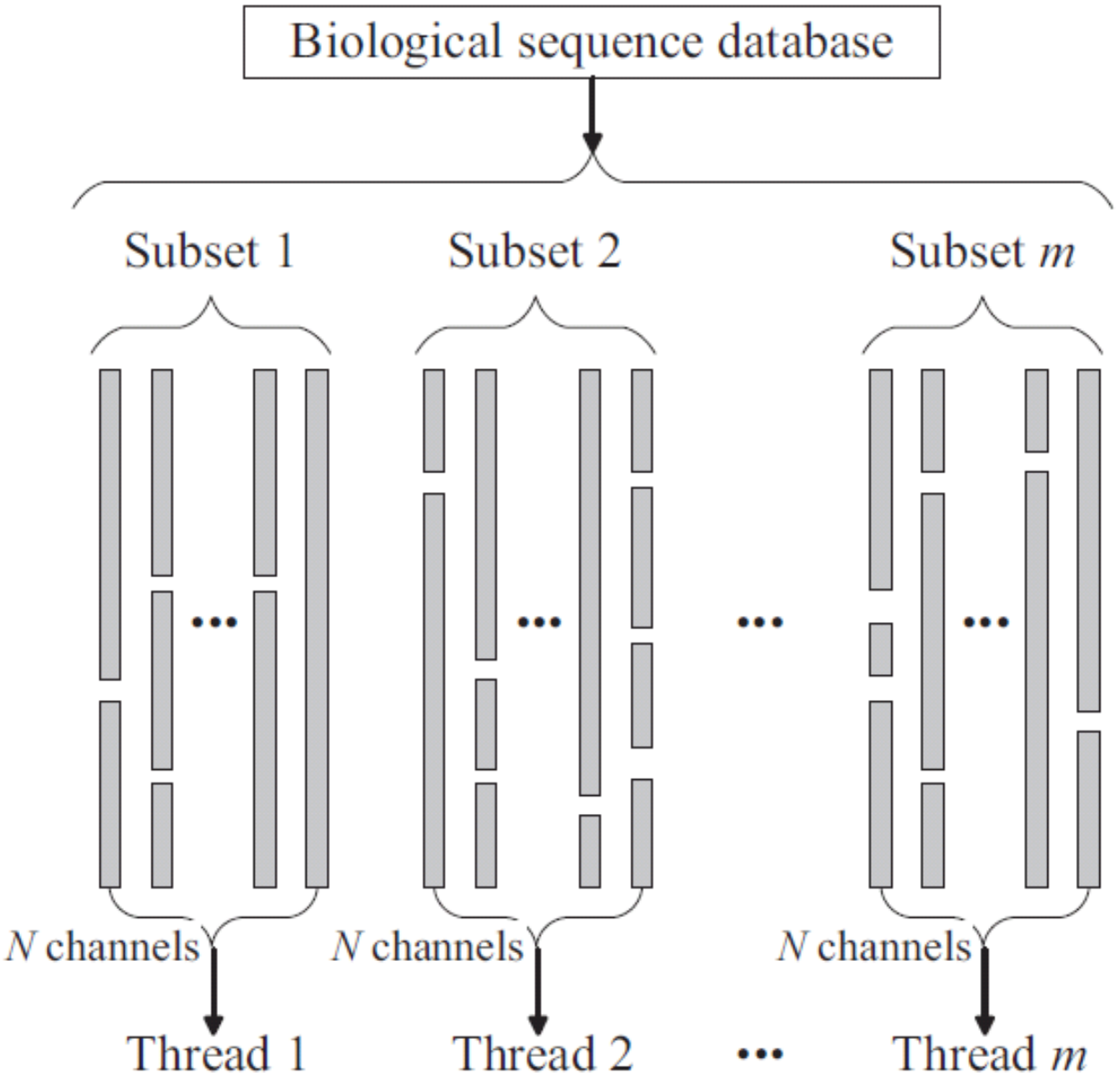
Y. Liu, etc. (2013)

- CUDASW++ 3.0: accelerating Smith-Waterman protein database search by coupling CPU and GPU SIMD instructions, BMC Bioinformatics, 14:117, 2013. (Impact Factor: 3.02)

Our Algorithm Framework

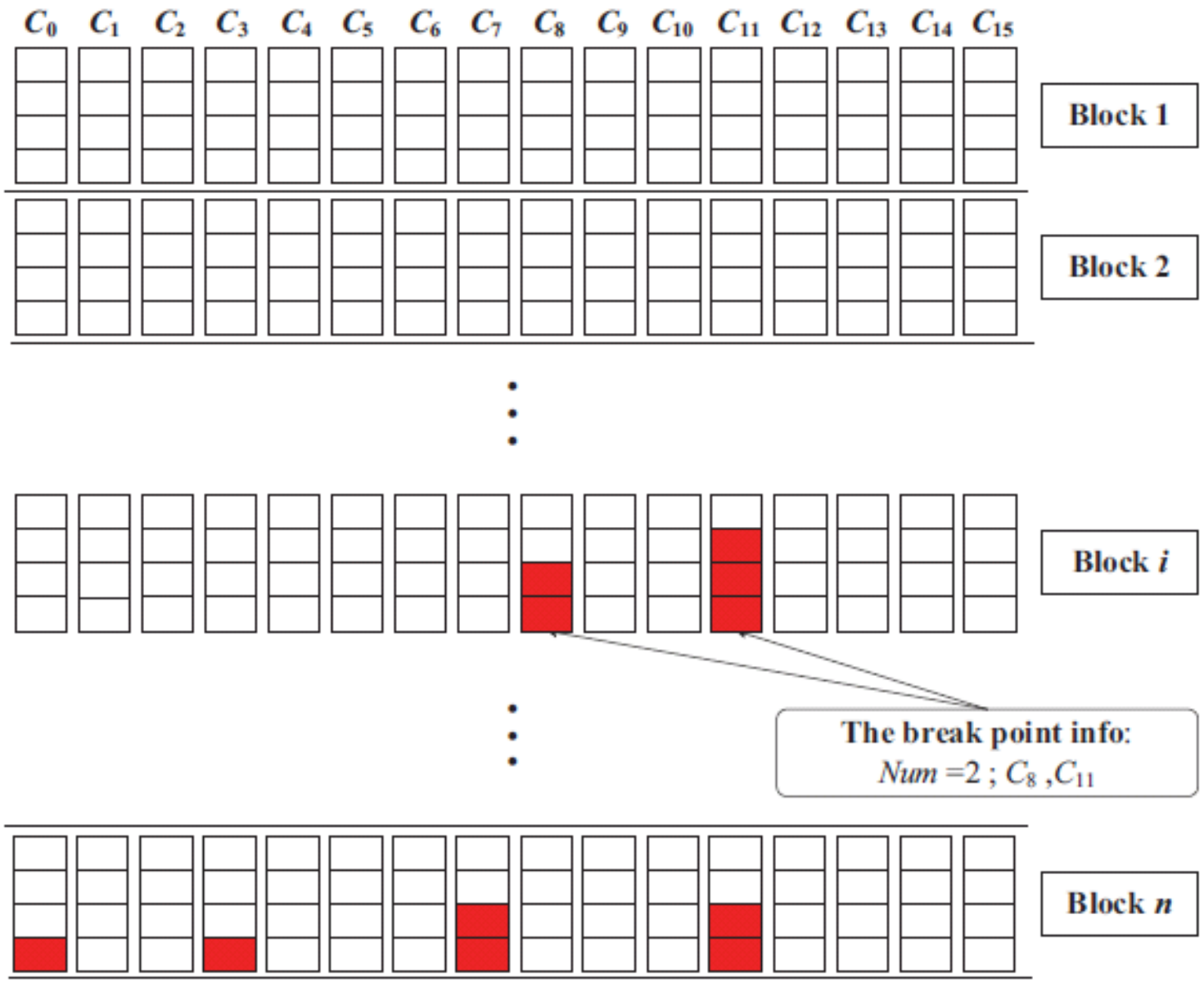


Coarse-grained Parallelism

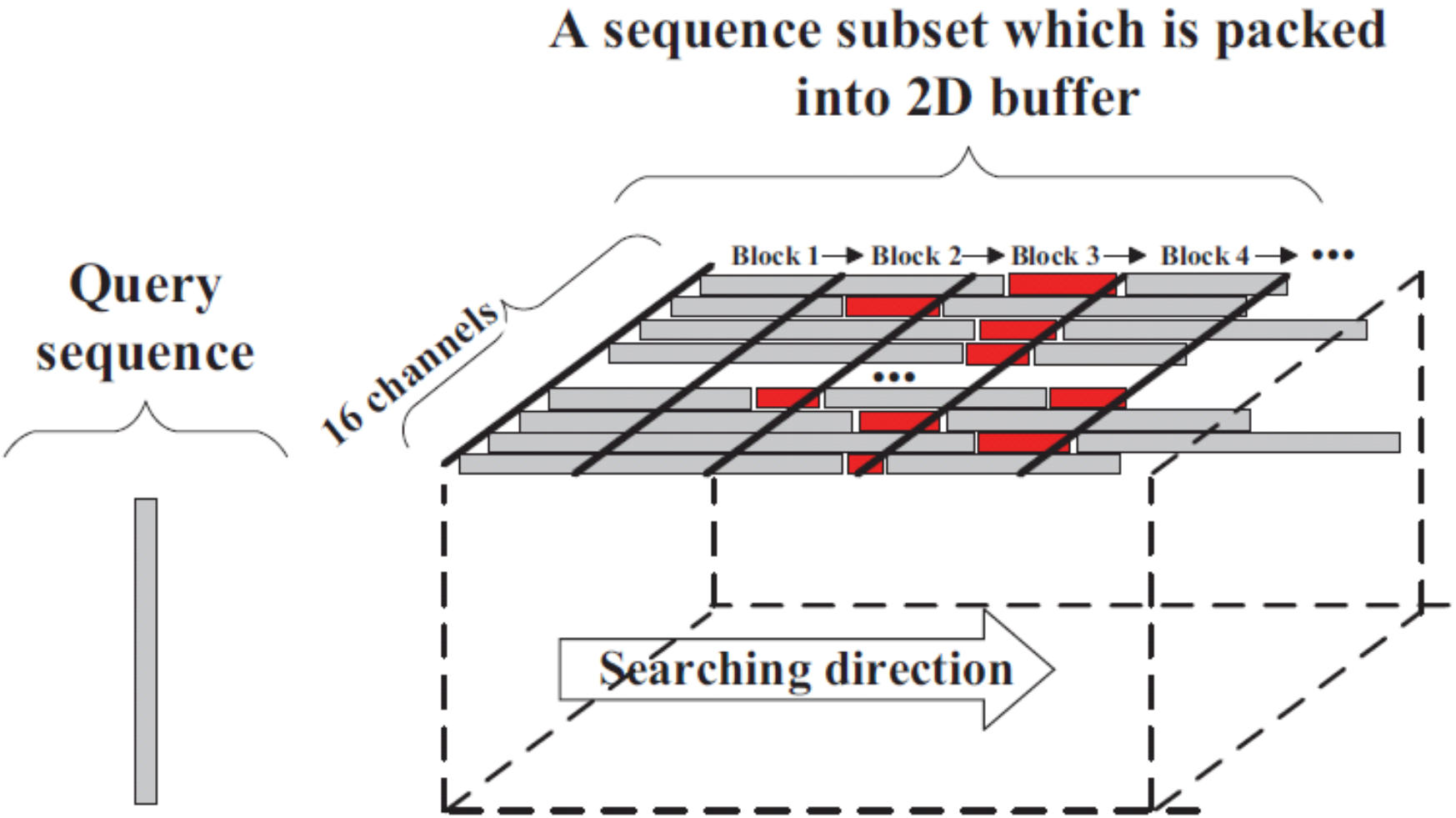


- Database is partitioned into small subsets
  - Reduce the superfluous computation
  - Achieve better load balancing

Fine-grained Parallelism



Fine-grained Parallelism

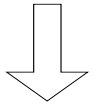


Fine-grained Parallelism

$$H(i,j) = \max\begin{cases} 0 \\ E(i,j) \\ F(i,j) \\ H(i-1,j-1) + Sbt(S1_i, S2_j) \end{cases} \quad ,1 \le i \le l1, 1 \le j \le l2$$

$$H(i,0) = E(i,0) = 0$$
$$H(0,j) = F(0,j) = 0$$

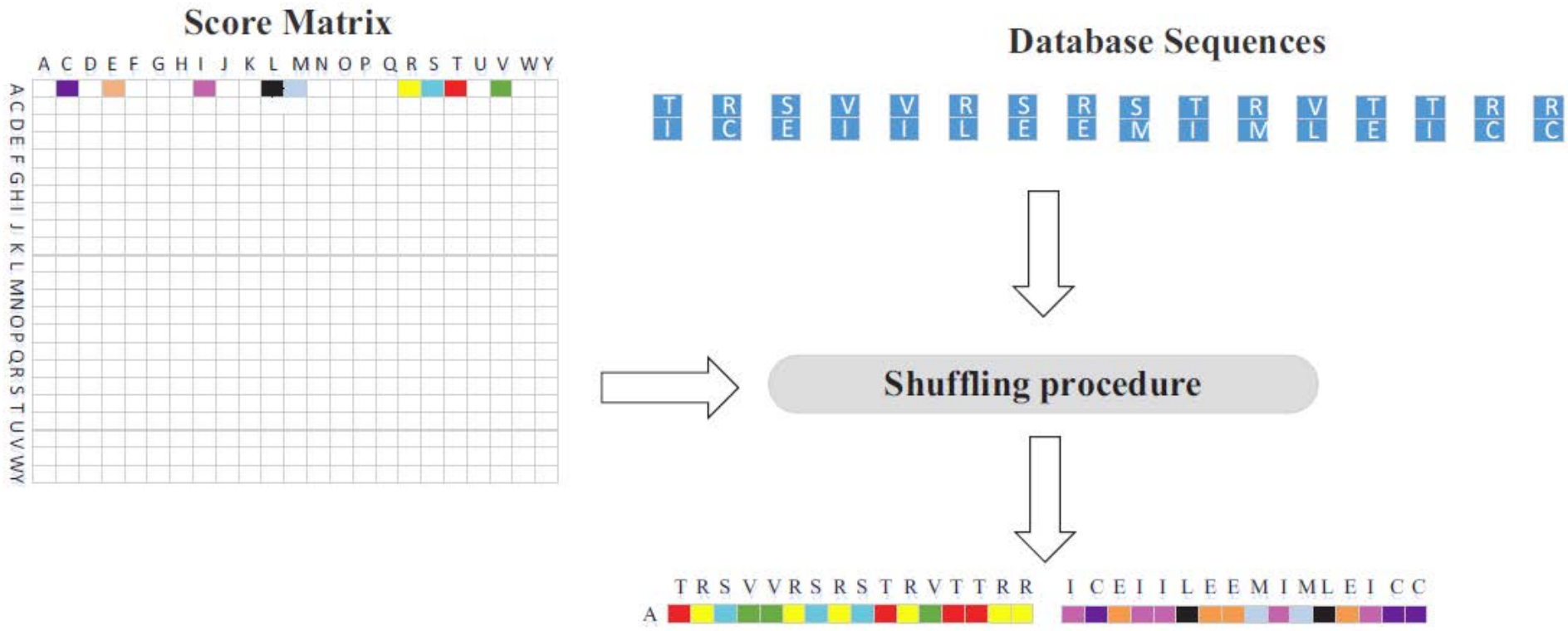
$$E(i,j) = \max\begin{cases} H(i,j-1) - \alpha \\ E(i,j-1) - \beta \end{cases}, \quad F(i,j) = \max\begin{cases} H(i-1,j) - \alpha \\ F(i-1,j) - \beta \end{cases}$$



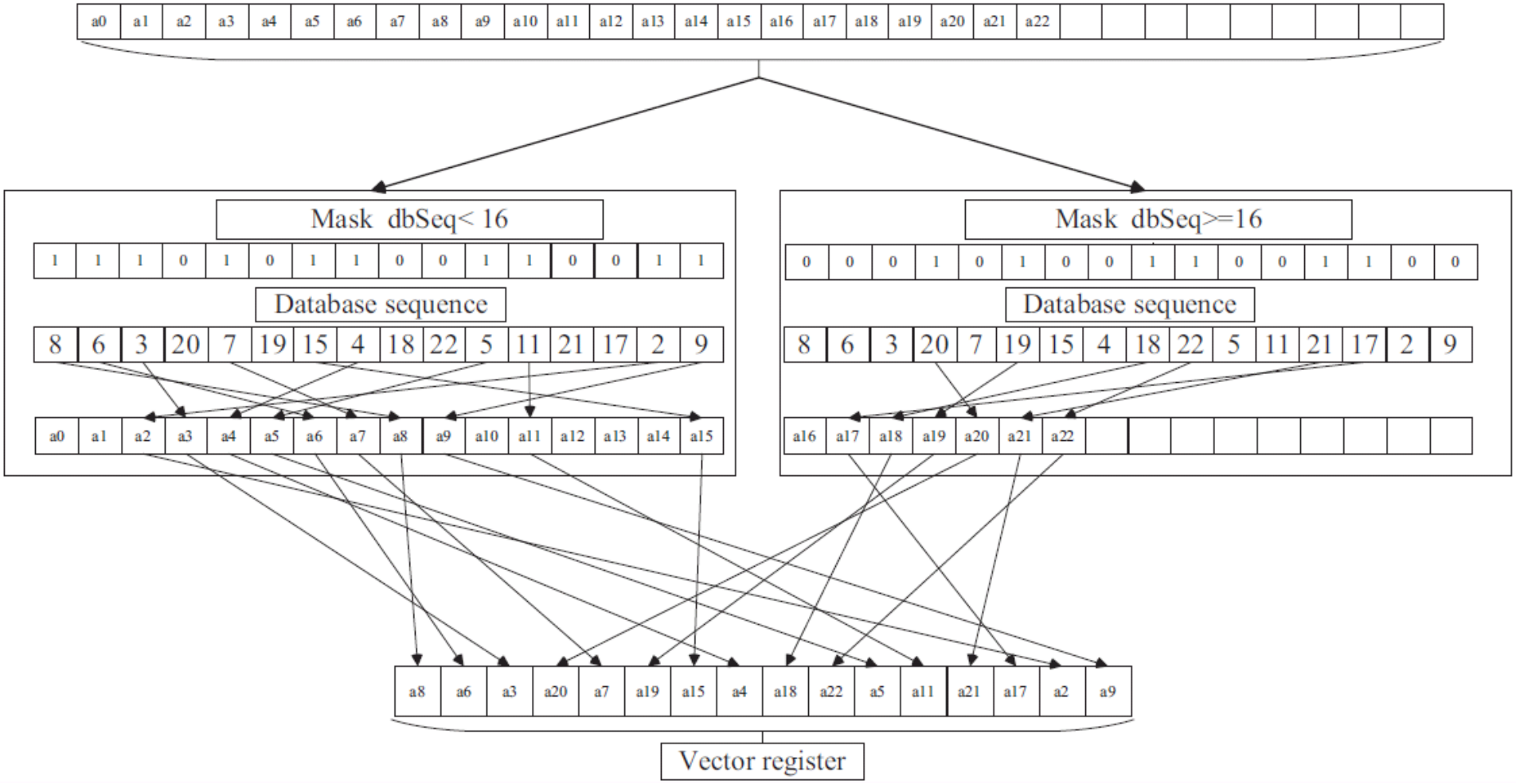
```
1.  vH[i] = _mm512_add_epi32(vH[i], vSbt);                  //H = H + sbt[q]
2.  vH[i] = _mm512_max_epi32(vH[i], vF[i]);                 //H = max(H, F)
3.  vH[i] = _mm512_max_epi32(vH[i], vE);                    //H = max(H, E)
4.  vH[i] = _mm512_max_epi32(vH[i], vZero);                 //H = max(H,0)
5.  vS = _mm512_max_epi32(vS, vH[i]);                       //S = max(S, H)
6.  vF[i] = _mm512_sub_epi32(vF[i], β);                     //F = F - β
7.  vE = _mm512_sub_epi32(vE, β);                           //E = E - β
8.  vN[i] = _mm512_mask_mov_epi32(vN[i], 0xffff, vH[i]); // N = H
9.  vH[i] = _mm512_sub_epi32(vH[i], α);                     //H = H - α
10. vE = _mm512_max_epi32(vH[i], vE);                       // E = max(H, E)
11. vF[i] = _mm512_max_epi32(vH[i], vF[i]);                 //F = max(H, F)
```

AsHES 2014

Fine-grained Parallelism



Fine-grained Parallelism

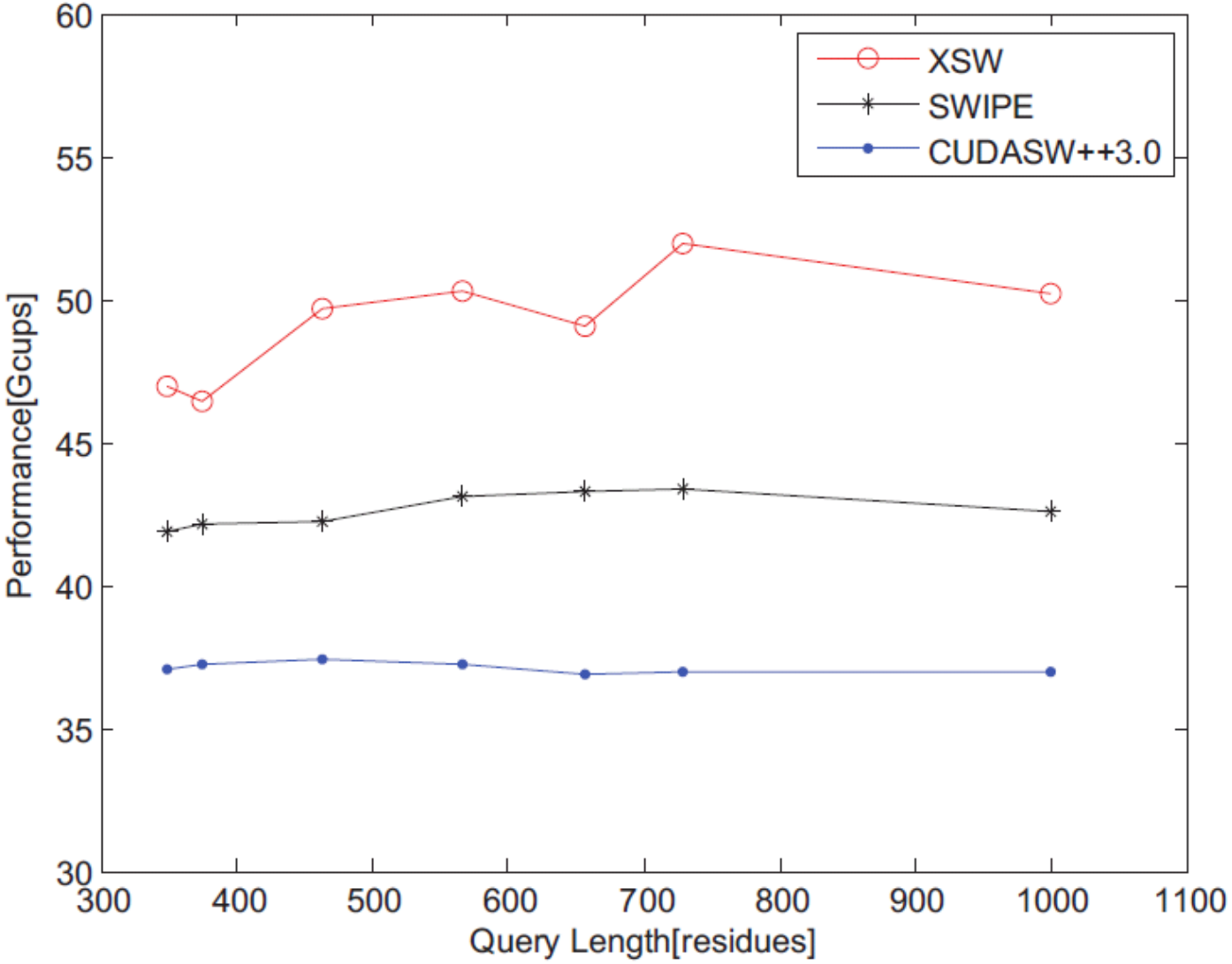


Performance Evaluation

- XSW: Implemented using C, Pthreads, and KCI.
- Performance evaluation on a PC server with an Intel E5-2620 six-core 2.0GHz CPU and an Intel Xeon Phi 7110P card. The server has 16GB RAM and runs Linux Red Hat 6.3.
- Performance comparison to SWIPE and CUDASW++ 3.0 running on a K20 GPU which is installed on the same PC server.
- Two biological databases are used: Swiss-Prot (541,954 sequences) and Environmental NR (6,165,520 sequences).

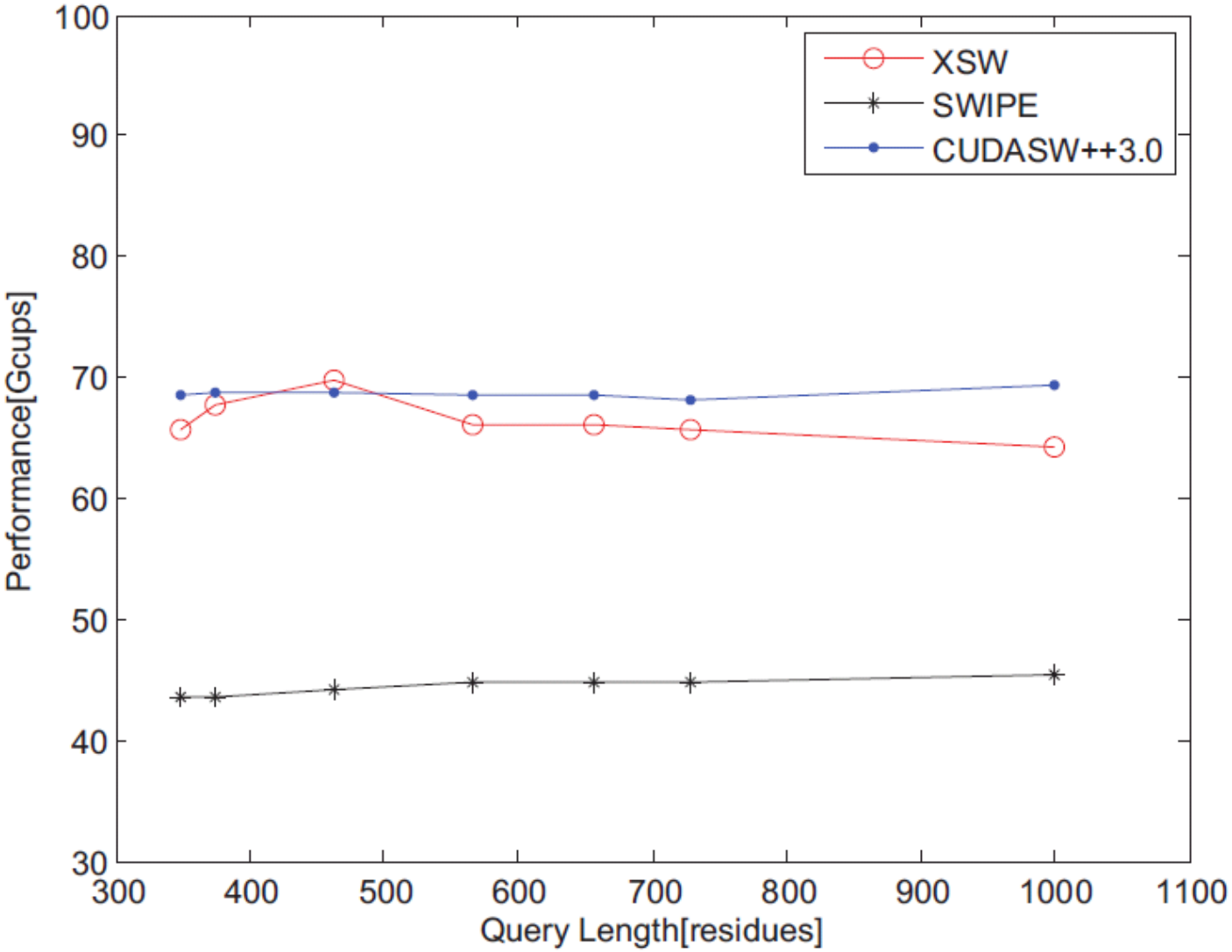
Performance Evaluation

- Performance comparison for scanning the Swiss-Prot.



Performance Evaluation

- Performance comparison for scanning the Environmental NR.



Conclusion

- Xeon Phi offers a flexible solution with a very good price/performance ratio for the SW algorithm (http://sdu-hpcl.github.io/XSW/)
- Achieved better performance than SWIPE and CUDASW++ 3.0 on an Xeon Phi 7110P
- Since the performance of many-core architectures grows faster than multi-core CPU, Xeon Phi-centric HPC will become even more important in the future

Future Work

XOmics

- Design and develop Omics-related algorithms on Xeon Phi

XFILE

- an Efficient File System for Processing Large-scale Data using Xeon Phi

XMR

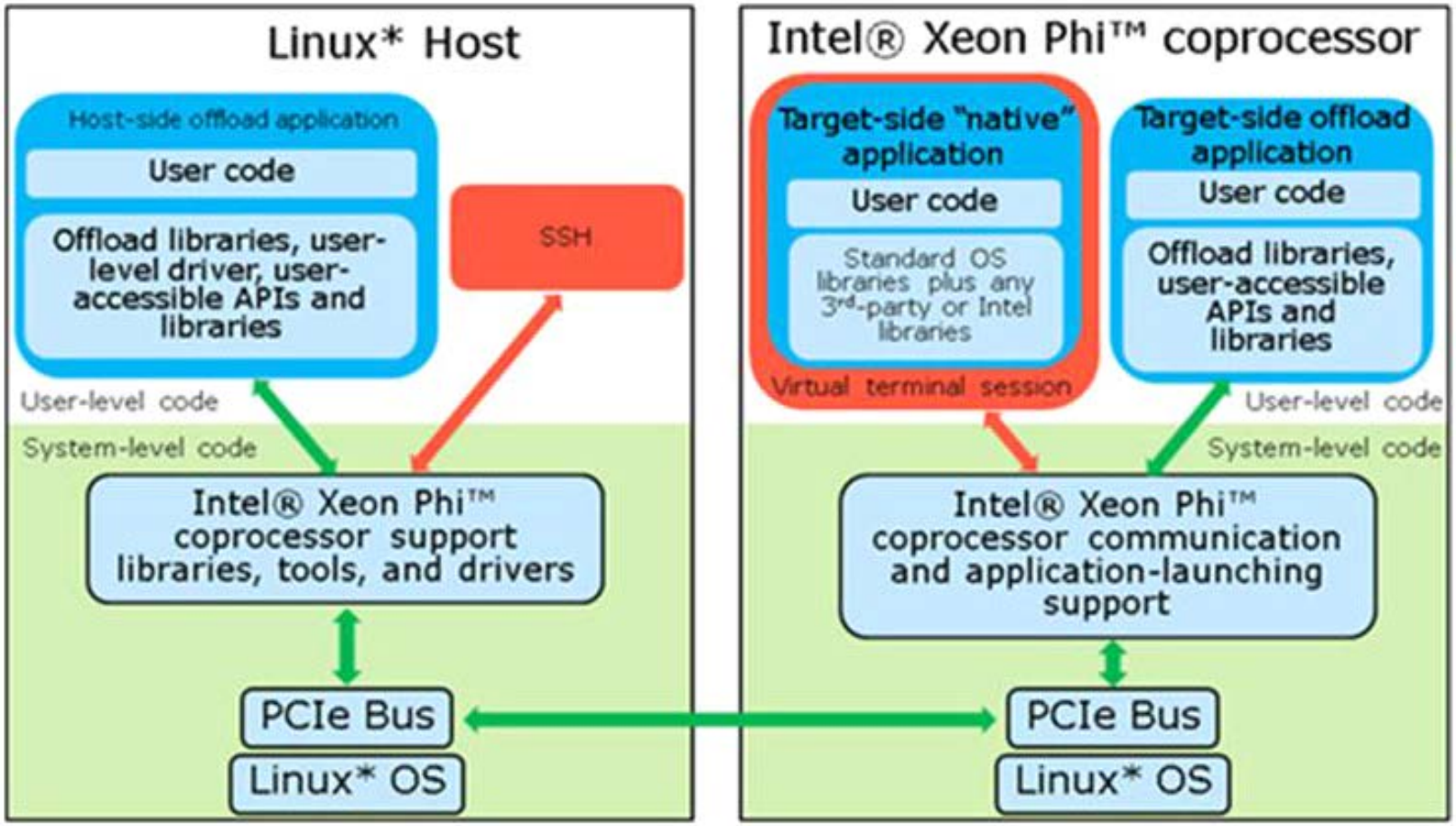
- A Heterogeneous Architecture-based MapReduce Framework for Large-scale Data Processing

XDC

- Xeon Phi Accelerated Compression Framework for Large-scale Data

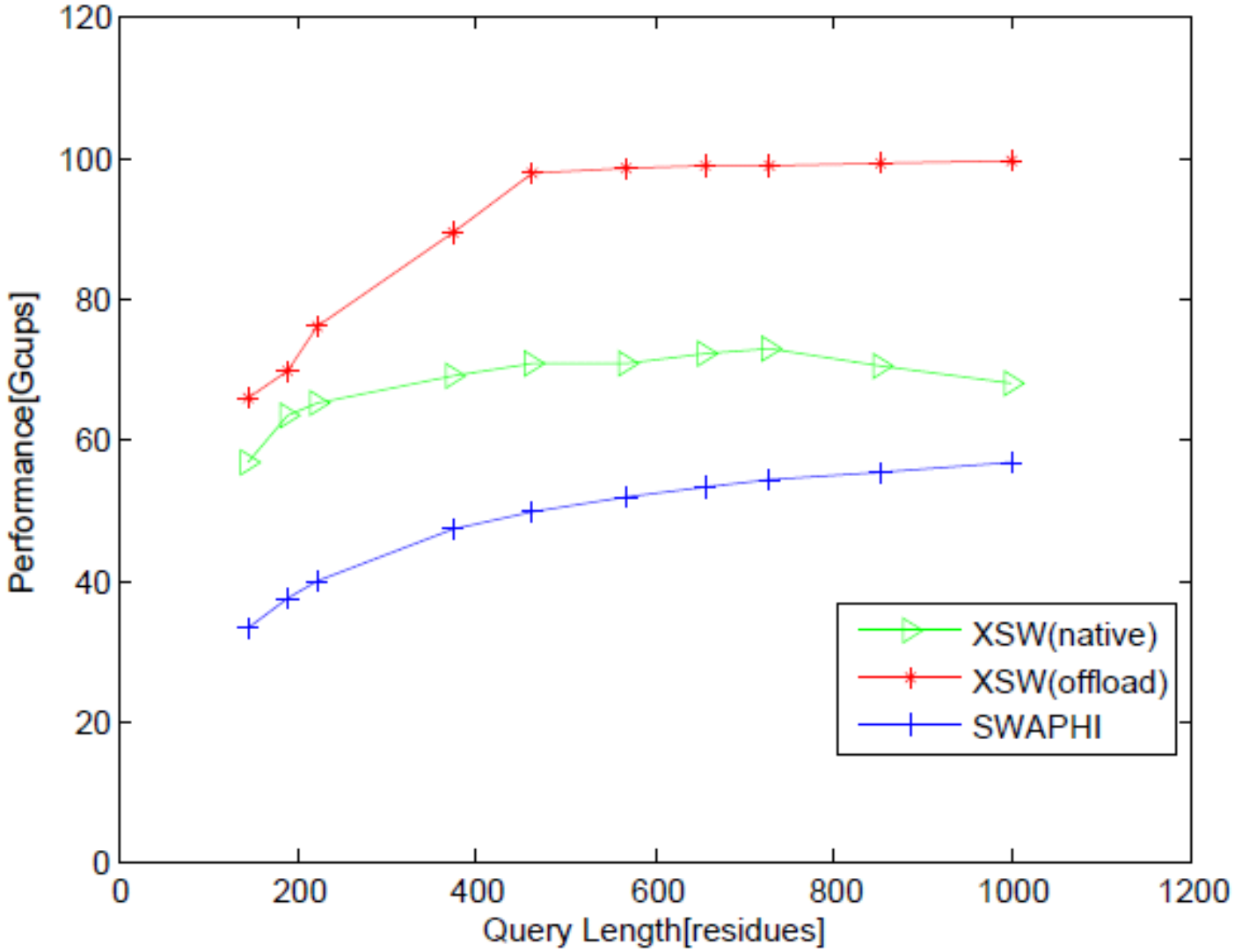
New Results: XSW 2.0

- Scanning large-scale databases using the offload programming model.



New Results: XSW 2.0

- Performance comparison to SWAPHI for scanning the Environmental NR.



New Results: XSW 2.0

- Performance for scanning large-scale DB (NR + TrEMBL, totally 36GB).