Iris Matching Algorithm on the Intel Xeon Phi Coprocessor

Guthrie A. Cordone
Mentor: Dr. Chen Liu
Department of Electrical and Computer Engineering, Clarkson University

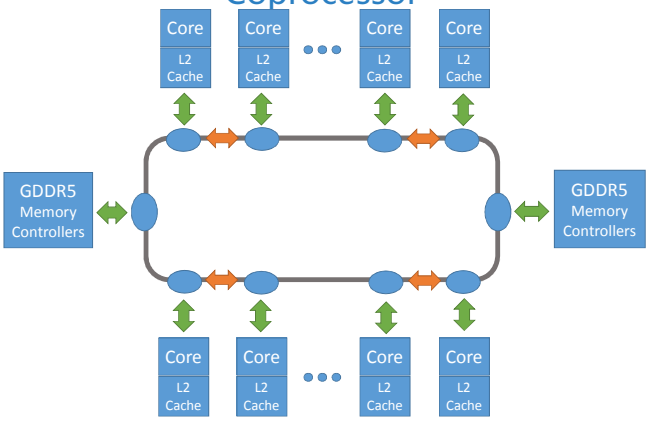
Introduction

- Biometric identification methods such as iris matching require repetitive computations on large sets of data. This inherent parallelism makes them inefficient to run on traditional serial processors.
- These applications are most efficiently run on highly parallel hardware platforms, which are capable of performing many computations simultaneously through the use of multithreading and multiple cores.

Goals

- Port an iris matching algorithm onto the Intel Xeon Phi Coprocessor
- Explore the performance of the algorithm on the coprocessor while varying the parallelism of the computation

Architecture of Intel Xeon Phi Coprocessor



- 60 cores in a ring topology
- Total of 240 simultaneous threads with 4 logical threads per core

Iris Matching Algorithm

Pseudocode for the Iris Matching Algorithm[1]:

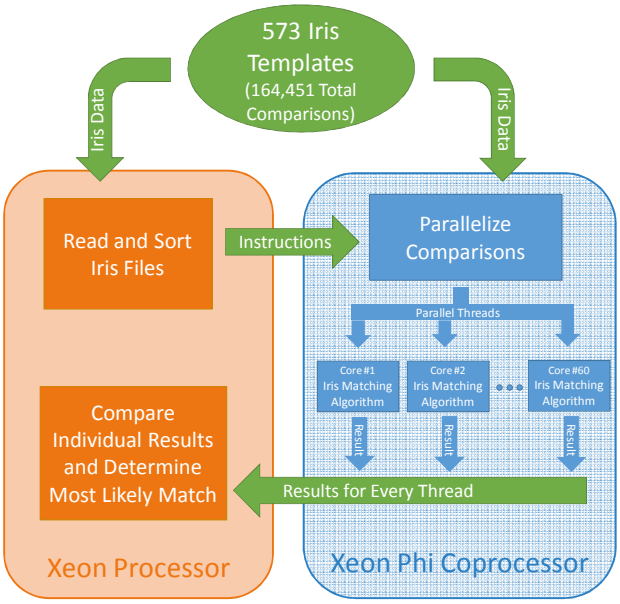
Algorithm 1 Iris_Matching Algorithm

```
template1, mask1, template2, mask2;
{Input 2-D Matrix with ROW rows and COL columns}
Hamming_Distance;            {Return value}
template1s, mask1s, temp, mask, result;  {Intermediate 2D Matrix}
hd=0, hd_min=1;                          {Intermediate variable}

for shift=-8 to 8 do
    template1s ← Rotate_Column(template1, shift);
    mask1s ← Rotate_Column(mask1, shift);
    temp←Matrix_XOR(template1s, template2);
        mask←Matrix_OR(mask1s, mask2);
    result←Matrix_AND(temp,mask);

    hd ← (number of 1s in Matrix result) / (ROWxCOL –number
                                            of 1s in Matrix temp)
    if hd < hd_min then
        hd_min ← hd
    end if
end for
Hamming_Distance ← hd_min;
return Hamming_Distance;
```

Algorithm 2 Rotate_Column

```
{Two Input Parameters: Matrix MAT and variable shift}
{Return Matrix MAT}
if shift ≤ 0 then
  rotate left MAT for 2 × abs(shift) columns;
else
  rotate right MAT for 2 × abs(shift) columns;
end if
return MAT;
```

Programming Model

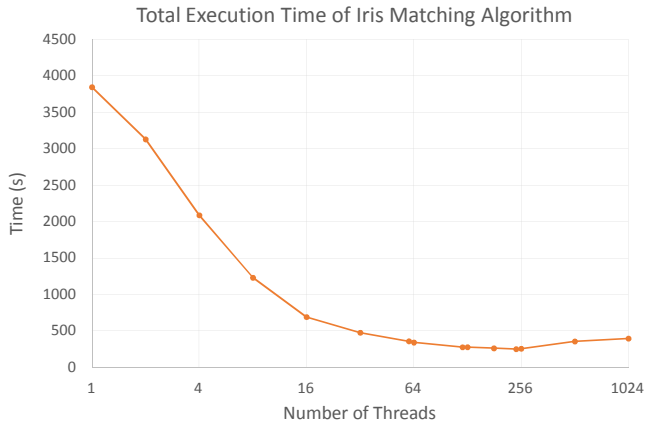


Method for Experiment

- Identify the two templates in the database with the highest likelihood of being a match
- Vary the number of threads used in the computation from 1 to 1024 and record completion time

Results

- Execution time decreased as the number of threads increased to 240
- Execution time increased as the number of threads went above 240
- Computation with 240 threads is 15 times faster than computation with one thread



- Future work includes scaling down to explore internal parallelism within each comparison and scaling up to a Cloud-Computing platform

Acknowledgements

This work is supported by NSF Grants No. ECCS-1301953, IIP-1332046, and NICS BEACON project titled "Design Space Exploration of Biometrics on Intel Many Integrated Core Architecture".

References

1. G. Torres, J. Chang, F. Hua, C. Liu, and S. Shuckers, "A Power-Aware Study of Iris Matching Algorithms on Intel's SCC" in The 2013 International Workshop on Embedded Multicore Systems (ICPP-EMS 2013)