Accelerating Time Series Subsequence Matching on the Intel Xeon Phi Many-core Coprocessor

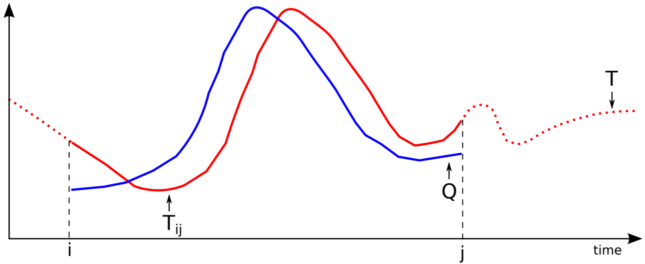
Ruslan Miniakhmetov
Alexander Movchan
Mikhail Zymbler

South Ural State University, Chelyabinsk, Russian Federation

MIPRO 2015

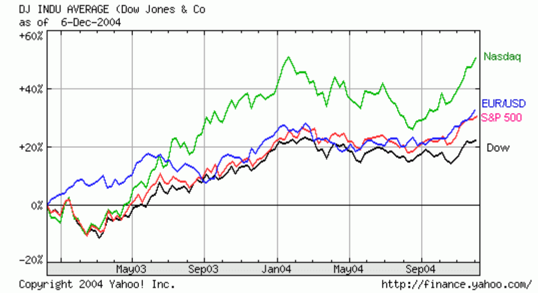
This work was financially supported by the Ministry of education and science of Russia (''Research and development on priority directions of scientific-technological complex of Russia for 2014-2020'' Federal Program, contract No. 14.574.21.0035).

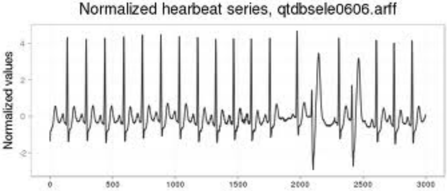
Background: Time Series



- *A time series* T is an ordered sequence $t_1, t_2, ..., t_N$ of real data points, measured chronologically, where N is a length of the sequence.
- *A subsequence* T_{ij} of time series T is its continuous subset starting at i position and ending at j position.
- *A query* Q is a certain subsequence to be found in T .

Background: Applications





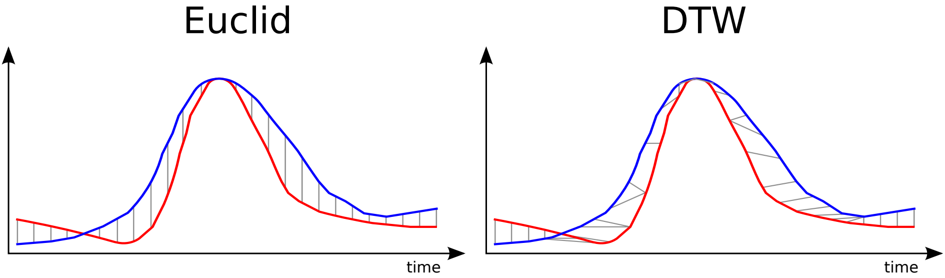
Background: Subsequence Matching

Subsequence matching problem aims to finding subsequence T_{ij} such that distance $D(T_{ij}, Q)$ is minimal.

Best-match search: find $T_{ij} \in T$, where
$\forall T_{mn}$

- $|T_{mn}| = |T_{ij}| = |Q|$
- $D(T_{ij}, Q) < D(T_{mn}, Q)$

Background: Dynamic Time Warping

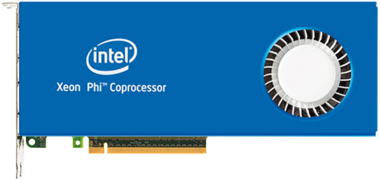


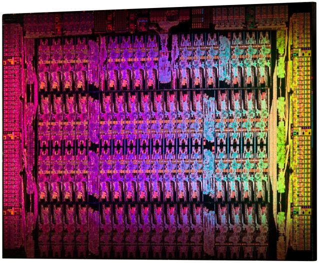
$$DTW(X,Y) = d(N,N),$$

$$d(i,j) = |x_i - y_j| + min \begin{cases} d(i-1,j) \\ d(i,j-1) \\ d(i-1,j-1), \end{cases}$$

$$d(0,0) = 0; d(i,0) = d(0,j) = \infty; i = 1,2,\ldots,N; j = 1,2,\ldots,N.$$

Intel Xeon Phi Many-core Coprocessor

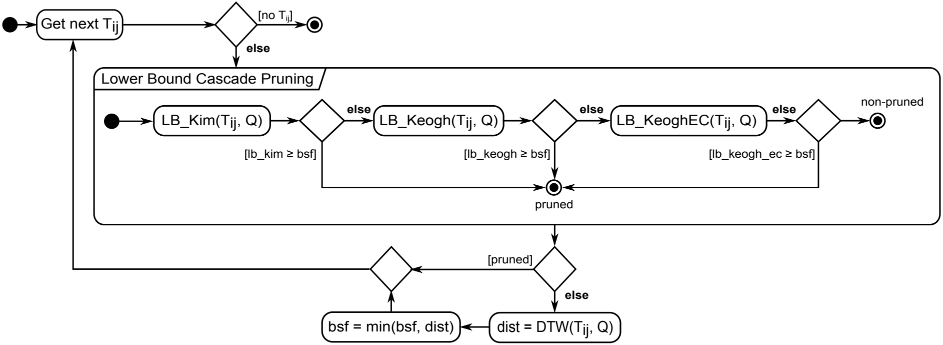




61 core, 244 threads, ≈1.2 TFLOPS, 512-bit SIMD

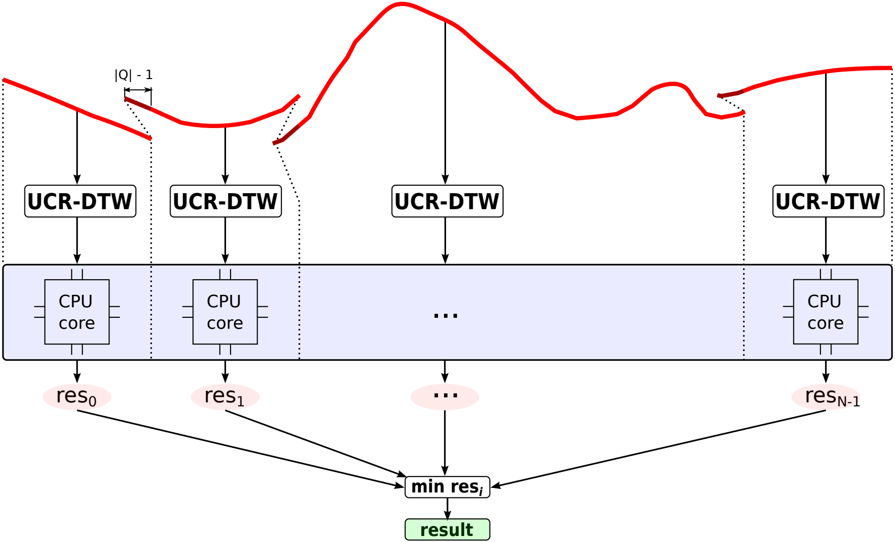
- Native Execution
  - independent execution on the coprocessor
- Offload Mode
  - execution on the host, offloading computationally intensive part of work to the coprocessor
- Symmetric Mode
  - execution on the coprocessor as MPI process

UCR-DTW Serial Algorithm



Rakthanmanon T., et al. Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping // The 18th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Beijing, China, 12-16 August, 2012. ACM, 2012. P. 262–270.

Parallel Algorithm for CPU

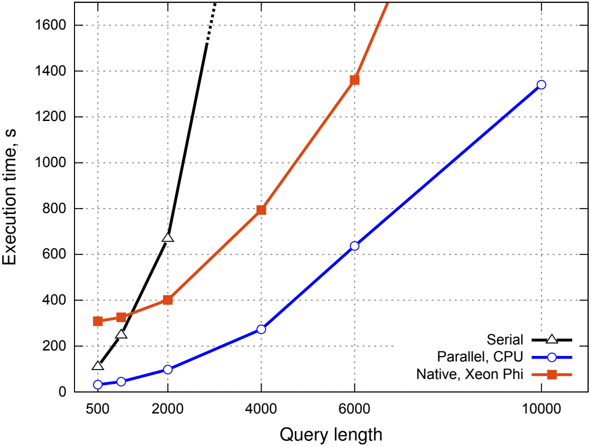


Performance of Parallel Algorithm for CPU

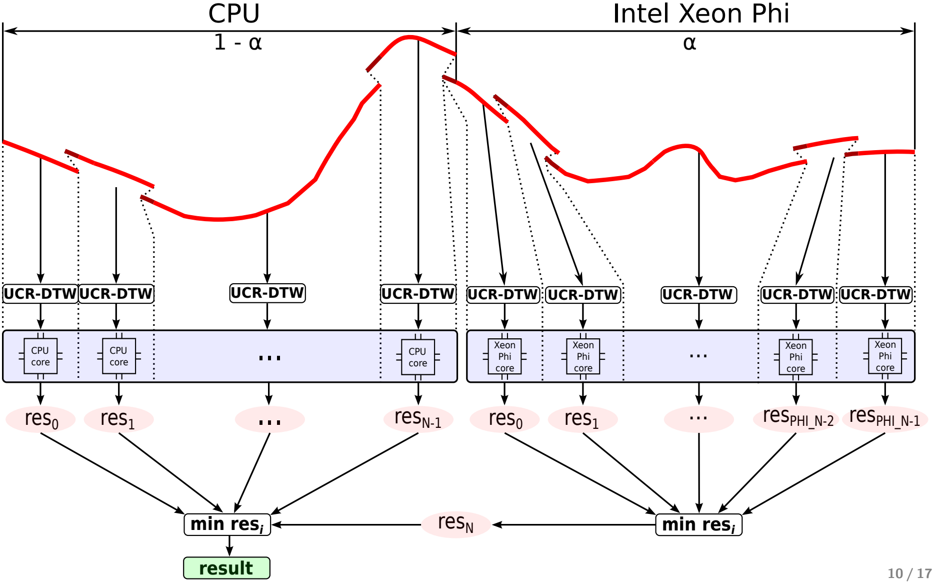
LB_Kim	$O(1)$
LB_Keogh	$O(n)$
LB_KeoghEC	$O(n)$
DTW	$O(n^2)$

Time of loading data from disk into memory of Intel Xeon Phi: $\approx 300\ c$

Data Set: random walk, 10^8 datapoints

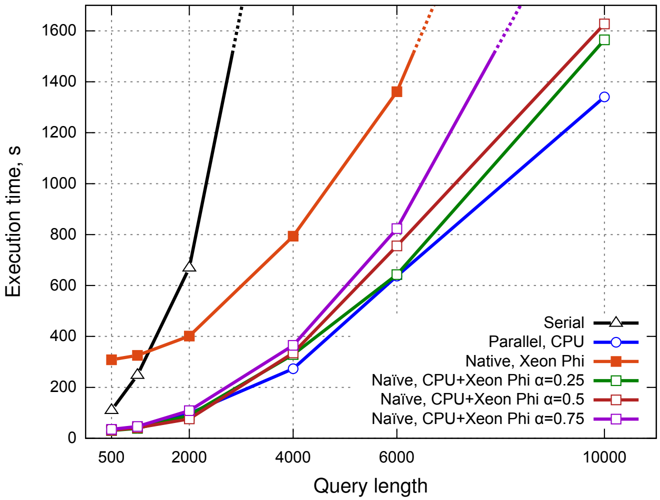


Naïve Parallel Algorithm for CPU and Intel Xeon Phi

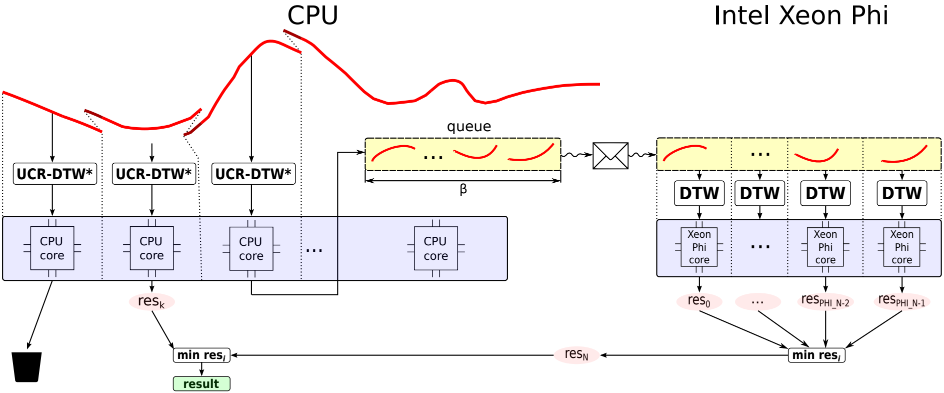


Performance of Naïve Parallel Algorithm for CPU and Intel Xeon Phi

Data Set: random walk, 10^8 datapoints



Advanced Parallel Algorithm for CPU and Intel Xeon Phi

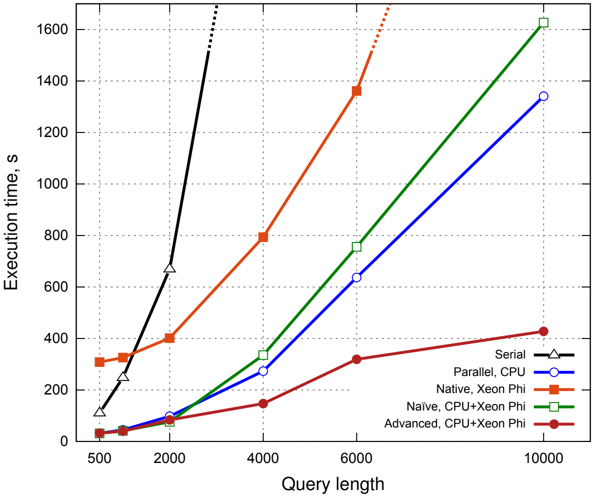


Experiments

- Hardware (Tornado SUSU supercomputer's node)
  - CPU
    - Intel Xeon X5680
    - 6 cores on 3.33 GHz
    - 0.371 TFLOPS
  - Coprocessor
    - Intel Xeon Phi SE10X
    - 61 cores on 1.1 GHz
    - 1.076 TFLOPS
- Data Sets
  - Synthetic
    - random walk, 10^8 datapoints
  - Real
    - ECG, 2×10^7 datapoints (approximately 22 hours of ECG sampled at 250 Hz)

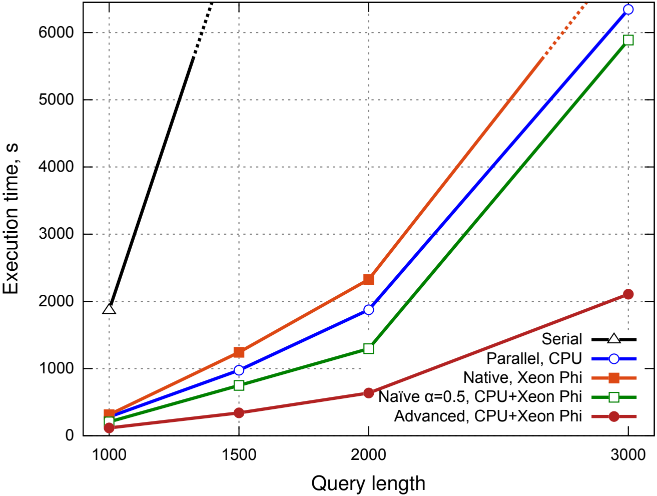
Performance on Synthetic Data Set

Data Set: random walk, 10^8 datapoints



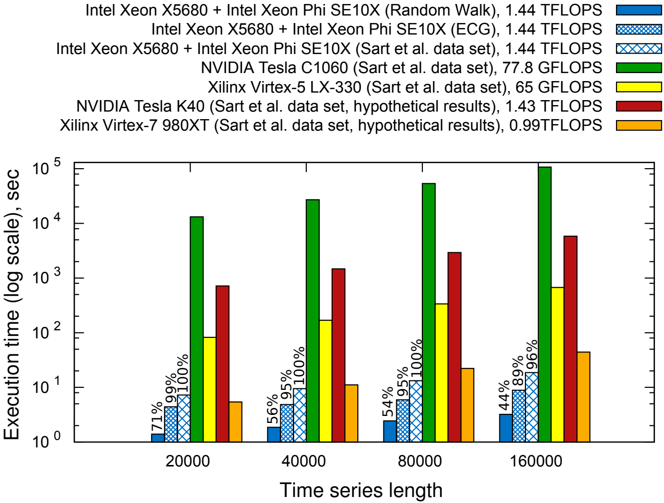
Performance on Real Data Set

Data Set: ECG, 2×10^7 datapoints



Comparison with Algorithms for GPU and FPGA

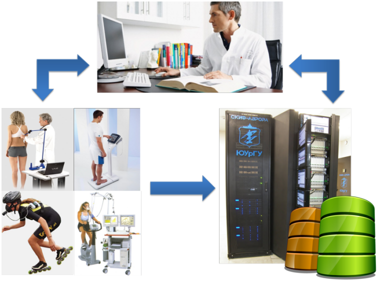
Sart et al. Accelerating dynamic time warping subsequence search with GPUs and FPGAs // ICDM, 2010.
Query length: 1024

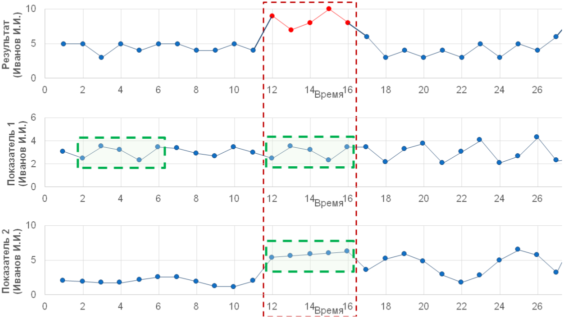


Conclusion

- A parallel algorithm of subsequence matching for the Intel Xeon Phi coprocessors was developed.
- Experiments on synthetic and real data sets have shown that our algorithm are effective on the long queries.
- Future work:
  - extend our algorithm for the case of a cluster system based on nodes equipped with the Intel Xeon Phi coprocessors;
  - extend our algorithm for the case of a node equipped with a few Intel Xeon Phi coprocessors.

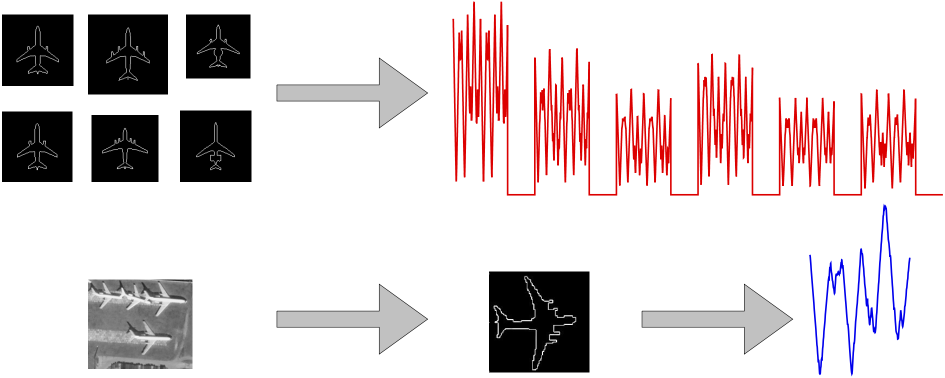
MedMining Project



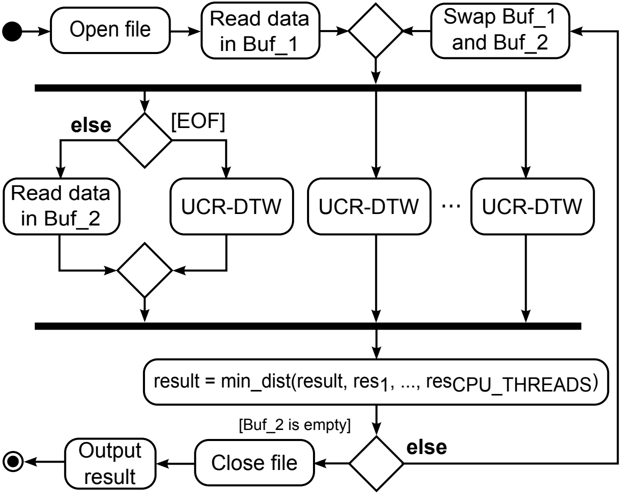


Data mining of physiological studies of professional athletes

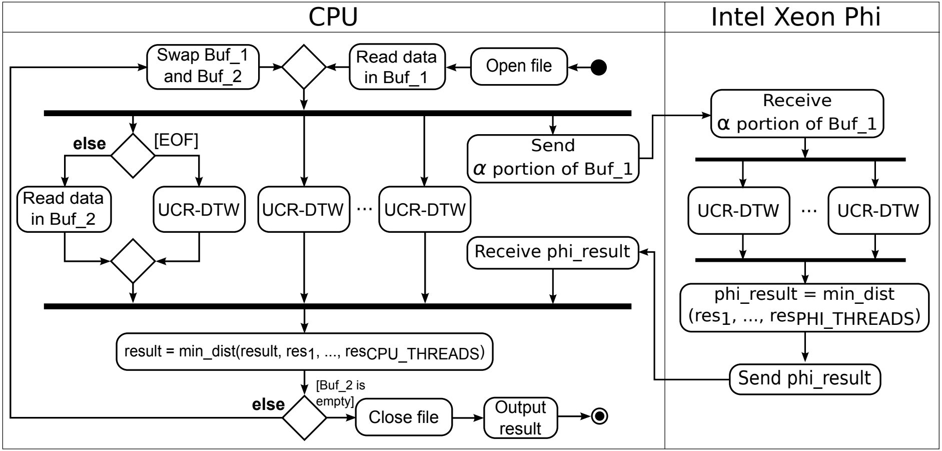
Classification of contour shapes



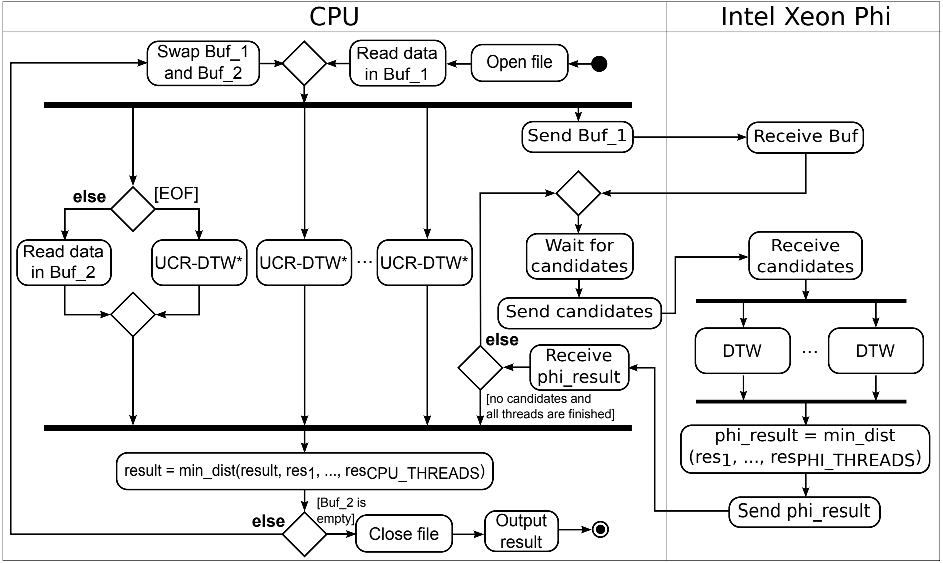
Parallel Algorithm for CPU



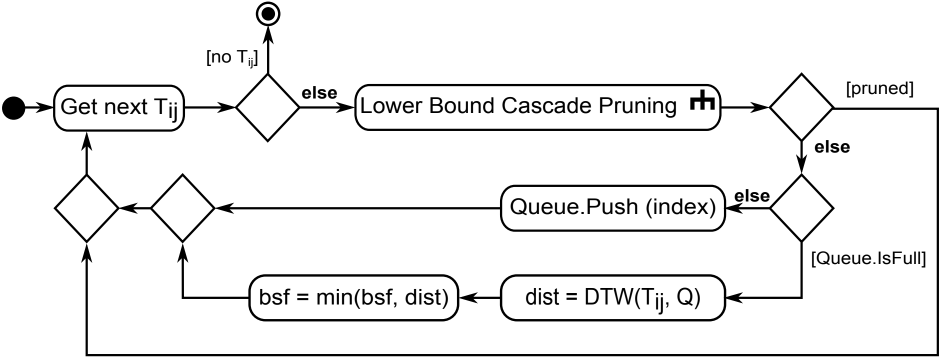
Naïve Parallel Algorithm for CPU and Intel Xeon Phi



Advanced Parallel Algorithm for CPU and Intel Xeon Phi

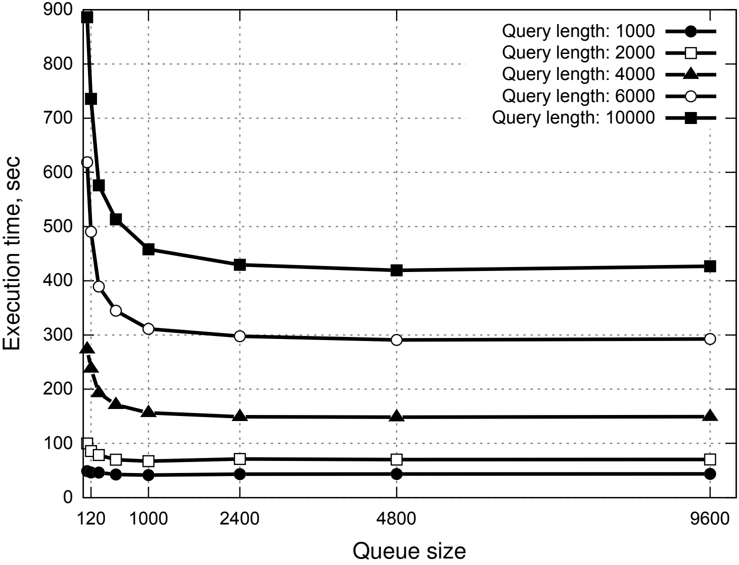


UCR-DTW*

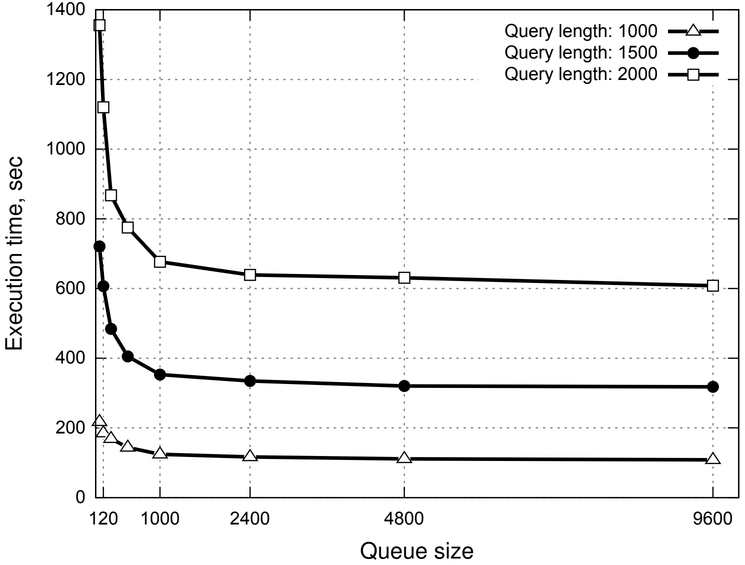


[no T_{ij}]
Get next T_{ij}
else
Lower Bound Cascade Pruning
[pruned]
else
Queue.Push (index)
else
[Queue.IsFull]
bsf = min(bsf, dist)
dist = DTW(T_{ij}, Q)

Queue Size: random walk



Queue Size: ECG


Query length: 1000
Query length: 1500
Query length: 2000
Execution time, sec
Queue size