Approximate Regular Expressions: A Comparison of Exact and Approximate Matching Algorithms

Umme Salma Gadriwala, Tasnim Noshin, Rumsha Siddiqui
{gadriwau, noshint, siddiqur}@mcmaster.ca
April 2019

MOTIVATION, PROBLEM, SOLUTION

MOTIVATION

- DNA sequences are often represented by regular expressions to capture different variations of the same structure.
- Efficient approximate string matching would allow us to capture more longer sequences, and optimize time and cost of resources.

PROBLEM

- To evaluate the costs and benefits of replacing an implementation of exact regular expression matching with one of approximate matching for added functionality

SOLUTION

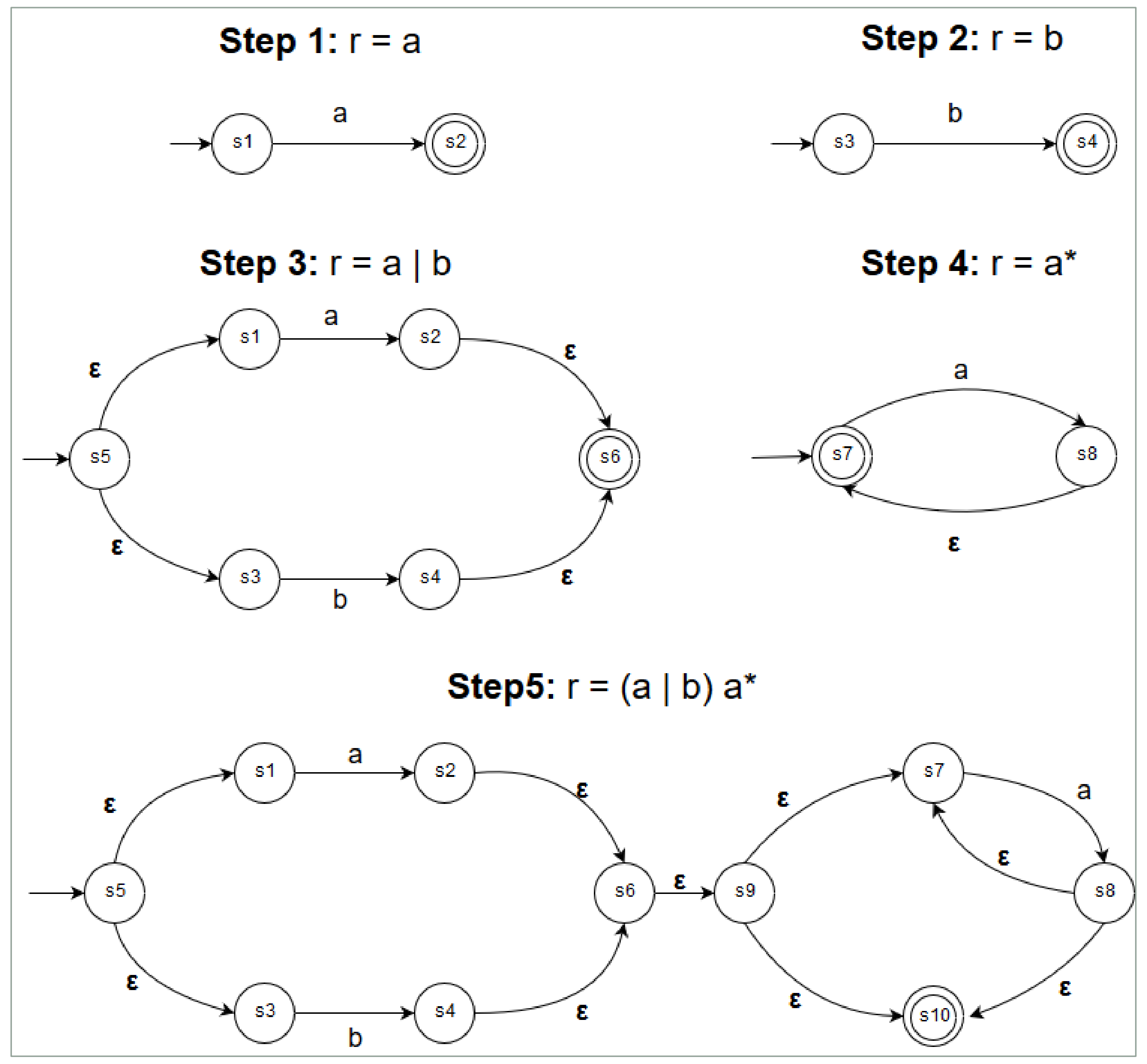
- A comparison of the running times of exact matching using Thompson's NFA to the Myers and Miller's approximate matching construction.

EXACT MATCHING: THOMPSON'S CONSTRUCTION

INPUT: Regular expression, *r* over Σ; and string, *s*

OUTPUT: True, if *s* satisfies *r*

METHOD: Traverse the NFA for s.
Return true if traversal ends at a terminal state.



Thompson's NFA with r = (a|b)a*

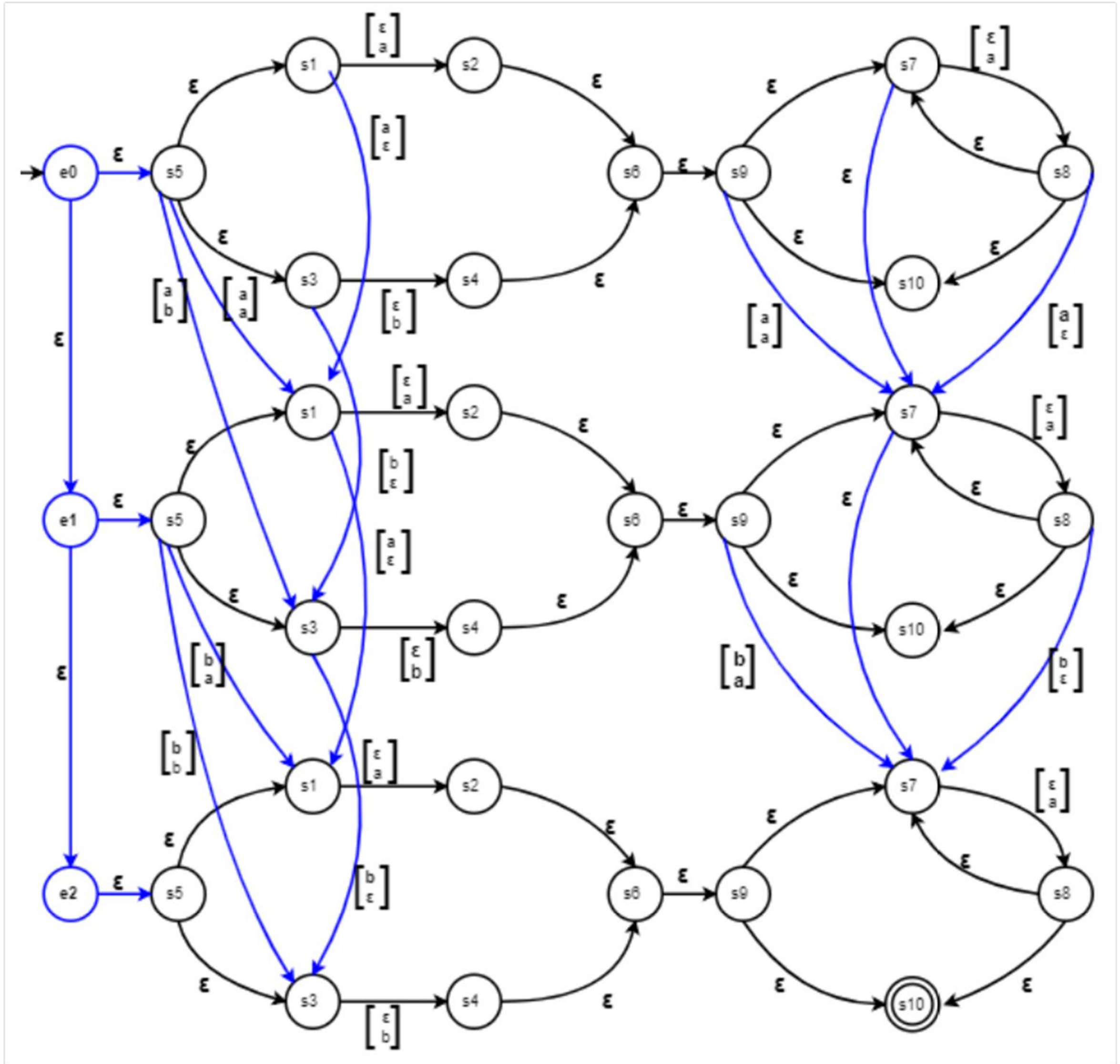
APPROXIMATE MATCHING: MYERS AND MILLER'S CONSTRUCTION

INPUT: Regular expression, *r* over Σ; string, *s*; and error value, *k*

OUTPUT: True, if *s* satisfies *r* after at most k errors

METHOD:

- Construct a Myers and Millers NFA by combining |*s*|+1 instances of Thompson's NFA construction of *r* by adding: deletion, insertion, and substitution edges based on *s*.
- Traverse the NFA for *s*, incrementing a counter for each error.
- Return *true* if *k* ≤ counter.



Myers' & Miller's NFA with r = (a|b)a*, s = "ab"

ANALYSIS

THOMPSON'S:

NFA construction: $O(|r|)$ steps, $O(|r|)$ space;

String traversal: $O(|r| \cdot |s|)$ steps;

Lines of code: 235

MYERS' & MILLER'S:

NFA construction: $O(|r| \cdot |s| + |s|)$ steps, $O(|r| \cdot |s| + 2^k)$ space;

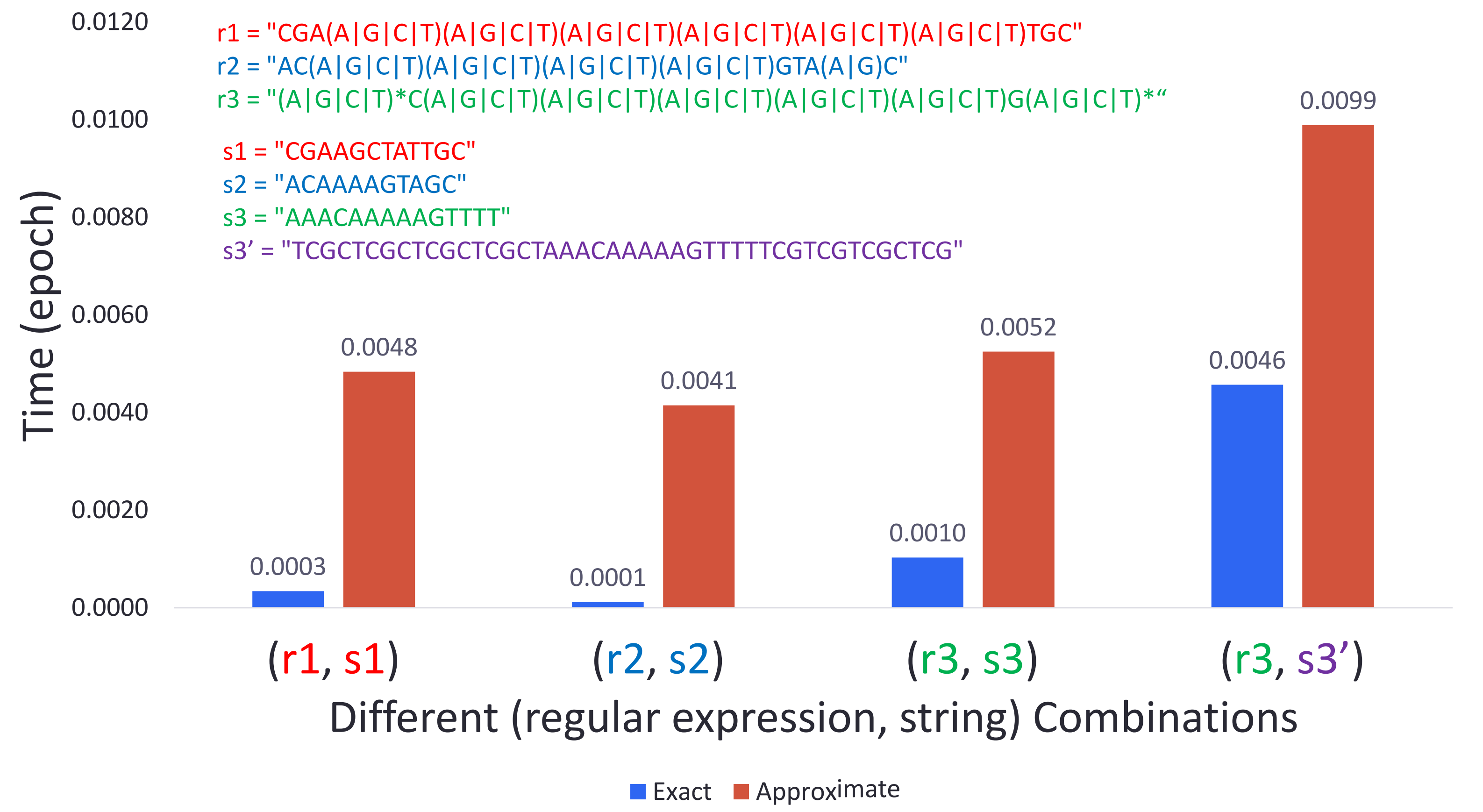
String traversal: $O(|r| \cdot |s| \cdot |s|)$ steps;

Lines of code: 295

PERFORMANCE TESTS

```
# Sample: compute runtime of exact matching algorithm
    for i in  range(numIterations):
        exact_nfa.match(string)
    end = time.time()
    # Compute average time to output
    exactTimes = exactTimes + [(end-start)/numIterations]
```

Runtimes of Various Regular Expressions and Strings for Exact and Approximate Matching Algorithms



HARDWARE SPECIFICATIONS

Processor	Intel Core™ i7-5500U CPU @ 2.40GHz 2.39 GHz	OS	Windows 10 (64-bit)
RAM	8.00 GB	Software	Python 3.5 (32-bit)

CONCLUSION

- Myers and Miller's approximate string matching takes more than 50% of the time Thompson's exact matching does for the given test cases.
- It is not worth the cost to use approximate matching where k = 0.
- This is inline with the algorithmic time complexities of string traversal.

REFERENCES

[1] D. Belazzougui, M. Raffinot, Approximate regular expression matching with multi-strings, Journal of Discrete Algorithms, Volume 18, Pages 14-21, 2013.

[2] E. W. Myers, W. Miller, Approximate Matching of Regular Expressions, Bulletin of Mathematical Biology, 1989.