COL 380	**Apr 15, 2019**
Lab 4	
Instructor: Subodh Sharma	**Due: Apr 26, 23:55 hrs**

Parallel Periodic-Pattern Matching using MPI

For this lab you have to search for given periodic pattern(s) in a given text using MPI. You must use the Algorithm 7.3 'Text Analysis, Periodic Case', Chapter 7 of *An introduction to Parallel Algorithms* by Joseph JáJá. The relevant sections of the Chapter have been provided in 'PatternMatching.pdf' on Moodle. The technique has been briefly discussed in this document.

1 Preliminary setup

1.1 Periodicity in Strings

Let Y be a string of length m. Length of the shortest substring X of Y such that $Y = X^k X'$, where X' is a proper prefix of X, is called the *period* of Y. The period is represented by p.

For example, let $Y = abababab a$, $X = ab$ $(X' = a)$. The period of $Y = |X| = 2$.
The string Y is called *periodic* if its period $p \leq m/2$.

Let Z be an arbitrary string of length $n \geq m$, then the following two statements hold;

- If Y occurs in Z at positions i and j, then $|i - j| \geq p$.
- If Y occurs in Z at positions i and $i + d$, where $d \leq m - p - 1$, then d must be a multiple of p. If $0 < d < m/2$, then Y must occur at positions $i + kp$, where k is an integer such that $kp \leq d$.

As a consequence, Y can occur in Z at most n/p times.

1.2 Witness Array

For a string Y of length m and period p, let $\pi(Y) = min(p, \lceil m/2 \rceil)$. ($\pi(Y) = p$ if Y is periodic, otherwise $\pi(Y) = \lceil m/2 \rceil$).

A *witness function* Φ_Y is defined as:

- $\Phi_Y(0) = 0$,
- $\Phi_Y(i) = k$, where k is any index such that $Y(k) \neq Y(i + k)$, for $1 \leq i < \pi(Y)$.

For example, let $Y_1 = abcaabcab$ and $Y_2 = abcabcab$. $\pi(Y_1) = \lceil m_{Y_1}/2 \rceil = 5$ and $\pi(Y_2) = p_{Y_2} = 3$.
$\Phi_{Y_1} = [0, 0, 0, 1, 4]$
$\Phi_{Y_2} = [0, 1, 1]$
(there are other correct values of Φ_{Y_1} and Φ_{Y_2} possible.)

Given a string Z of length $n \geq m$, consider the problem of determining all positions where a string Y can occur in Z. Let i and j be two positions in Z such that $|i - j| < \pi(Y)$, Y cannot occur at positions i and j simultaneouly. $\Phi_Y(j - i)$ provides a position where two copies of Y starting at i and j would differ.

function DUEL($Z(0:n-1),\ Y(0:m-1),\ \Phi_Y,\ i,\ j$) ▷ $0 \le i < j < n$
  $k := \Phi_Y$[j-i]
  if $Z(j+k)$ is not a valid index **or** $Z(j+k) \neq Y(k)$ **then return** i
  else return j
  end if
end function

2 Pattern Matching

Consider a text T of length n and a pattern P of length m.
In this section we will discuss a $O(log\ m)$ time parallel string matching algorithm using $O(n+m)$ operations. The algorithm consists of two steps:

1. *Pattern Analysis*: This step involves processing of the pattern to extract information about the structure of pattern. In this phase we compute the witness function Φ as discussed above.

2. *Text Analysis*: This step involves processing of the text using the information gathered in step 1. This step has been explained in Algorithms 12.

Algorithm 1 Non-periodic Pattern Matching

function NP-TEXTANALYSIS($T(0:n-1),\ P(0:m-1),\ \Phi_P$) ▷ It is assumed that P is non-periodic
  Partition T into $\frac{n}{\lceil m/2 \rceil}$ blocks T_0, ..., T_b. with at most $\lceil m/2 \rceil$ consecutive characters of T.
  for $bi := 0$ **to** b **do**
    $i := |T_0| + ... + |T_{bi-1}|$
    for $j := i+1$ **to** $(|T_0| + ... + |T_{bi}|) - 1$ **do**
      $i :=$ DUEL($T,\ P,\ \Phi_P,\ i,\ j$)
    end for
    potential-positions[bi] := i
  end for
  match-positions := emptySet
  for each i **in** *potential-positions* **do**
    if P occurs at i in T **then** ▷ compute using brute-force
      add i to *match-positions*
    end if
  end for
  return *match-positions*
end function

Example: Consider $T = babaababaaba$ and $P = abaab$. Clearly, P is non-periodic. $\pi(P) = \lceil 5/2 \rceil = 3$, $\Phi(P) = [0,\ 0,\ 1]$.

- T is partitioned into $\frac{12}{\lceil 5/2 \rceil} = 4$ blocks, i.e., $T_0 = bab$, $T_1 = aab$, $T_2 = aba$ and $T_3 = aba$. (All these blocks are handled concurrently.)

- After 1st round of duels we get, `DUEL`(0,1) = 1, `DUEL`(3,4) = 4, `DUEL`(6,7) = 6 and `DUEL`(9,10) = 9. After 2nd round of duels we get, `DUEL`(1,2) = 1, `DUEL`(4,5) = 4, `DUEL`(6,8) = 6 and `DUEL`(9,11) = 9.

- We get the potential candidates, 1, 4, 6, 9.
- We can check that P occurs at locations 1 and 6.

Algorithm 2 Periodic Pattern Matching

```
function P-TextAnalysis(T(0 : n − 1), P(0 : m − 1))
    p := period of P
    P′ := P(0 : 2p − 2)
    Φ_P′ := WITNESS(P′)
    pos := NP-TextAnalysis(T, P′, Φ_P′)
    u := P(0 : p − 1)
    k := ⌊m/p⌋
    v := P(kp : m − 1)
    for i := 0 to n − 1 do
        M[i] := 0
        if i ∈ pos and u^2 v occurs at i then M[i] := 1
        end if
    end for
    for i := 0 to p − 1 do
        S[i] := (M[i], M[i + p], M[i + 2p], ...)
        for j := 0 to |S[i]| − 1 do
            C[i][j] := 0
            if there are k − 1 consequetive 1s starting at i then
                C[i][j] := 1
            end if
        end for
    end for
    for j := 0 to n − m do
        if ∃i, l such that 0 ≤ i < p, l ≥ 0 and j = i + lp then
            MATCH[j] := C[i][l]
        end if
    end for
end function
```

Examples: Consider text $T = bababababababaabab = (ba)^6(ab)^2$ and pattern $P = abababa = (ab)^3a$. Pattern P has a period $p = 2$ (ab).

- $P' = P(0:2) = aba$. (P' represents the longest non-periodic substring of P).
- $\Phi_{P'} = [0, 0]$. (This is one of the correct $\Phi_{P'}$ values.)
- Algorithm 1 identifies locations $pos = 1, 3, 5, 7, 9$ and 12 that P' occurs at in T.
- Occurances 1, 3, 5, 7 can be extended to $(ab^2)a$.
- Hence, $M = [0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]$.
- $S[0] = [0, 0, 0, 0, 0, 0, 0, 0]$, $S[1] = [1, 1, 1, 1, 0, 0, 0, 0]$.
- $C[0] = [0, 0, 0, 0, 0, 0, 0, 0]$, $C[1] = [1, 1, 1, 0, 0, 0, 0, 0]$.
- Therefore, $MATCH = [0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0]$.
- Pattern P occurs in text T at positions 1, 3 and 5.

3 Lab submission details

3.1 Problem Statement

- Your task is to implement parallel periodic pattern matching for single text, multiple patterns.
- You must adapt the Algorithm 2 for multiple patterns.
- You will be provided a text, $Text$, a set of patterns $PatternSet$, and a set of periods p_set corresponding to each pattern in $PatternSet$.
- You have to return a set of $MATCH$ arrays corresponding to the patterns in $PatternSet$.

3.2 Input format

The input file format and the main file to calculate the time taken by your implementation can be cloned from `https://github.com/dvynjli/col380_lab4_suite`. Read the `README` file for details.

3.3 Plagiarism policy

Make sure the code is not plagiarized. We have a repository of online codes. If you are found copying code from online sources or your peers, you will be penalized with a grade-drop + zero in the lab.

3.4 Late submission policy

10% penalty for each day over the deadline. Maximum of 2 days allowed after which summary zero will be awarded (unless there is a medical emergency – for which you will have to provide a letter from the doctor).

3.5 Evaluation scheme

1. Correct *parallel* implementation. **50 Marks**
2. Performance **50 Marks**

Note: The grading for component 2 (*Performance*) will be relative.