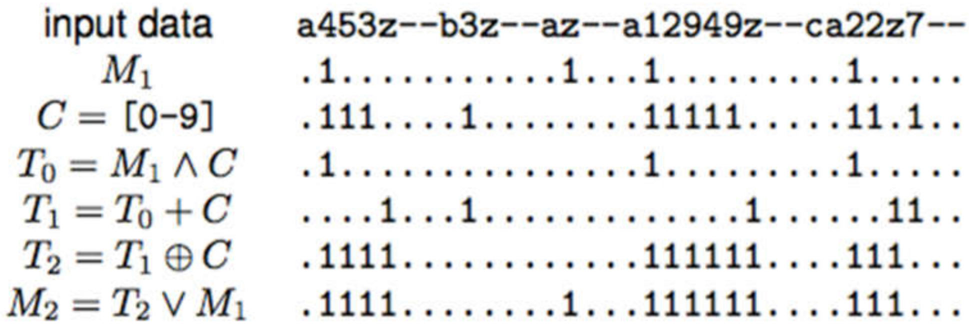
Regular Expression Improvements to be made in the Parabix Framework

By Fahad Aldebeyan and Duji Tufail

The MatchStar Function

In *icGrep*, MatchStar: m(M,C) is an operation that returns all positions that can be reached by scanning from the initial position marked in M along the span of ones in the stream C for zero or more steps. MatchStar gets its name from the star operator (*) in the regular expression [Meng Lin, 2014].

Figure 1 shows an example of it matching [0-9]* in the regular expression '[a][0-9]*[z9]' where M1 marks positions after occurrences of [a]. [Taken from Bitwise Data Parallelism in Regular Expression Matching slides, Rob Cameron].



```
input data        a453z--b3z--az--a12949z--ca22z7--
M1                .1...........1...1.........1.....
C = [0-9]         .111....1........11111.....11.1..
T0 = M1 ∧ C       .1...............1.........1.....
T1 = T0 + C       ....1...1.............1......11..
T2 = T1 ⊕ C       .1111............111111....111...
M2 = T2 ∨ M1      .1111........1...111111....111...
```

Figure 1: MatchStar primitive, where M2 = MatchStar(M1, C).

MatchStar Limitations

Currently, if the star (*) operator contains more than one character class and it occurs in the middle of the regular expression, the program generates a while loop. So a regular expression like `'a(bc)*d'` (without the quotes) would enter a while loop after matching the character '`a`' to iterate through all occurrences of `(bc)`. **Figure 2** is a part of the Pablo code generated printed out using the option `-print-pablo` with the command `./icgrep 'a(bc)*d'` demonstrating this limitation:

```
while Next(pending):
  CC_621 = (CC_62 & pending)
  ipp1 = pablo.Advance(CC_621, 1)
  CC_631 = (CC_63 & ipp1)
  ipp2 = pablo.Advance(CC_631, 1)
  not_27 = (~accum)
  and_71 = (ipp2 & not_27)
  Next(pending) = and_71
  or_35 = (ipp2 | accum)
  Next(accum) = or_35
CC_641 = (Next(accum) & CC_64)
matchstar = pablo.MatchStar(CC_641, any)
and_72 = (matchstar & and_56)
matches = and_72
```

Figure 2: Snippet of Pablo code generated to find matches of `(bc)*` in the regular expression '`a(bc)*d`'.

Ideally, the cases that generate a while loop should instead generate a fixed amount of operations to match what's inside the star (*) structure, similar to the current behavior of MatchStar in the scenario of one character class.