About substring instances and periodic strings

Definition 1: A string N (the needle) *has an instance* as a substring at index i in string H (the haystack) iff $\forall k \in [0, |N|), N[k] = H[i + k]$. If $\forall k \in [0, p), N[k] = H[i + k]$ is only true for some $p < |N|$, then N has a *partial instance* of length p at index i.

Definition 2: Two instances of a string N at indices i, j in some string *overlap* iff the intervals $[i, i + |N|)$ and $[j, j + |N|)$ overlap (i.e. have a nonempty intersection).

Definition 3: A string S has a *repeated prefix* of length $L > 0$ at index $m > 0$, iff $(\forall k \in [0, L), S[k] = S[m + k]) \wedge (m + L = |S| \vee S[L] \neq S[m + L])$.

Definition 4: A string S is *periodic* with *periodic prefix* P, iff $|S| > |P| \wedge \forall k \in [0, |S|), S[k] = P[k \mod |P|]$. In that case, the number of repetitions of P in S, including any final partial one, is $\lceil |S|/|P| \rceil$. The length of the periodic prefix (i.e. $|P|$) is a *period* of S.

Theorem 1: To have overlapping instances in any string H, a string N must be periodic. Furthermore, all overlapping instances will be offset from each other by multiples of a period of N (i.e. if P is a periodic prefix and there are overlapping instances at indices i and j, then $(i - j)/|P| \in \mathbb{Z}$).

Proof: By Definition 2, overlap of two instances occurs iff an instance occurs at index i and another instance occurs at index $i + p$ for some $p \in [1, |N|)$. This means that $\forall k \in [0, |N| - p), N[k] = N[p + k]$ (since the instance at $i + p$ means that $H[(i + p) + k] = N[k]$ and the instance at i means that $H[i + (p + k)] = N[p + k]$), which is equivalent to saying that N consists of $\lceil |N|/p \rceil$ repetitions (the last one of which may be partial) of a prefix of length p (to see this, consider that if $p+k < |N|-p$, then $N[k] = N[p+k] = N[p+(p+k)]$, etc.).

About algorithms that find substring instances

To find all instances of a string N in another string H, it is by definition sufficient to test whether $\forall k \in [0, |N|), N[k] = H[i + k]$ for all i that are valid indices of H. Every instance has to start somewhere, and that "somewhere" can only be a valid index of H.

Assume that H is processed as a stream, one character at a time in ascending index order. Further assume that we have reached the end of a complete or partial instance that starts at index i, i.e. we have read and checked some number of characters $p \in [1, |N|]$ and found that $\forall k \in [0, p), N[k] = H[i + k]$ (but in the case of a partial instance (i.e. $p < |N|$), $N[p] \neq H[i + p]$). We may now have to consider some of those characters again, since there may also be instances that start somewhere in the interval $[i+1, i+p)$. A straightforward way to do this would be to maintain a look-behind buffer, containing all characters in H that have been read from the input stream during the substring match attempt but not yet tested as potential first characters of an instance of N.

However, since a substring match attempt only continues while the checked characters in H match the characters at the corresponding positions in N, we know that such a buffer would always contain a prefix of the known string N. This means that there can be an instance at any index $i + m$ in H, where $m \in [1, p)$, only if the substring of N that consists of the characters at indices $[m, p)$ matches a prefix of N (i.e. $\forall k \in [0, p-m), N[k] = N[m+k]$). In that case, we already have a new partial matching substring and may continue matching that substring from index $i + p$ in H and index $p - m$ in N. If we know all the places (if any) where N has a repeated prefix, then we know all the places where we may have to go back and look for another instance.

One approach to handling the need to re-check characters that have already been checked and read is to first generate a list where the start index m and length L of every repeated prefix in N are recorded and then, whenever the end of a complete or partial instance of length p has been reached, check whether any prefixes on the list satisfy the condition $m < p \wedge L \geq p - m$. If the prefix with the lowest start index among these prefixes is selected (this can easily be accomplished by using a prefix list that is sorted in ascending start index order) and the search then continues with a new matching substring of length $p - m$, then every instance of N in H will be found. This is because each instance that overlaps with another complete or partial instance must start at a repeated prefix, and if we always continue the search at the first repeated prefix that could be the start of an instance, then we'll never miss any instance.

Repeated prefixes that are contained in another repeated prefix (e.g. like "AA" at index 8 in "AABAAAABAACC" is contained in "AABAA" at index 5) may be omitted from the list of prefixes without breaking the algorithm. Contained prefixes will never be selected as continuation points by the algorithm, because the containing prefix always starts before the contained one and ends at or after it. This means that by the time $L < p - m$ for the containing prefix (making it disproven as a potential instance by subsequent non-matching characters and therefore no longer selectable), the same will be the case for the contained prefix.

Given the algorithm described above, applying overlapping or non-overlapping substring search is simply a matter of either consulting or not consulting the prefix list after a complete instance of the sought string has been found.

More on periodic strings

Theorem 2: No string has two distinct periods p, q that are not both multiples of a single period (a common factor of p and q, possibly p or q itself, or 1 (if the string is constant, i.e. just a repetition of a single character)).

FIXME: Theorem 2 needs to be restricted to be true. I have discovered that if a single complete repetition plus a partial repetition is enough to identify a period, then a string can indeed have several periods that are not multiples of any common factor period (e.g. "AABAAABAAA" has both period 4 and period 9 (but not period 1, which is the only common factor)).

TODO: Prove Theorem 2. The (restricted) theorem is implied by the statement that *no sequence* (finite or infinite, not even a constant one) has two periods that are not both multiples of a fundamental (i.e. shortest) period (if we exclude "periods" that are greater than half the length of a finite sequence). I believe this is true.

Corollary 2.1: If a string S is periodic, then it has a unique shortest periodic prefix P such that all periodic prefixes of S are multiples of P. This is just Theorem 2 stated as a universal positive instead of an existential negative (plus the facts that no string has more than one distinct prefix of the same length, that no shorter string is a (non-zero) multiple of a longer one and that "being a multiple of" is transitive).

Note that together, Theorem 1 and Corollary 2.1 imply that there's no need to backtrack in the haystack to find overlapping instances of the needle string N. If we are searching for a periodic N (and if we aren't, then by Theorem 1 there are no overlapping instances) with shortest periodic prefix P and find an instance that doesn't overlap with any preceding instance (i.e. an instance that begins at a lower index in the haystack), then we know that we potentially have the beginning of an overlapping instance (i.e. the found instance minus the first repetition of P). We may continue to check this potential overlapping instance against the suffix of N starting at index $|N| - |P|$. If we do find an overlapping instance, we may continue searching for additional overlapping instances in the same way.