T-REx (Tandem Repeat Exciser)

This document describes a hopefully more reliable method of screening out tandem repeats from SSAHA matches. It is not too hard to find repeats in the query sequence by looking at successive hash words, but here we address the more difficult problem of also deducing repeats in the *subject* sequence by looking at the pattern of hits we obtain.

We begin by introducing some important variables:

The motif length *m*: this is the length in bases of the repeating motif.
The word length *w*: this is the length in bases of each hash word.
The step length *s*: this is the length in bases between successive hash words, and is less than or equal to *w*. Perhaps surprisingly, *w* itself is not that important.

The algorithm is probably best illustrated by looking at a concrete example. Let *w=s=5* and suppose we have a single subject sequence

```
    1     6     11    16    21    26    31    36    41
S = AGCAG CAGCA GCAGC AGCAG CAGCA GCAGC AGCAG CAGCA GCAGC
```

and that

```
Q = AGCAG CAGCA GCAGC AGCAG CAGCA GCAGC AGCAG CAGCA GCAGC
```

is our query sequence. We can see that both sequences consist entirely of a repeating motif of length 3, i.e. *m=3*.

The standard SSAHA algorithm begins by progressing base-by-base along the query sequence, forming a hash word at each position. In our example, we proceed from position 1 to 41 of Q and obtain a table of hits as follows

Query Position	Word	Subject Position
1	AGCAG	1, 16, 31
2	GCAGC	11, 26, 41
3	CAGCA	6, 21, 36
4	AGCAG	1, 16, 31
5	GCAGC	11, 26, 41
6	CAGCA	6, 21, 36
....	...	...
39	CAGCA	6, 21, 36
40	AGCAG	1, 16, 31
41	GCAGC	11, 26, 41

The problem is two-fold: each hash word is queried multiple times (because Q is repetitive) and multiple matches are found for each hash word (because S is repetitive).

Our approach is to preprocess the list of hits to remove this repetition.

1. We denote by *h(i)* the hash word obtained by starting at base i of the query sequence Q. We compare *h(i)* with the hash words *h(i+1)* to *h(i+s)*. If it is not equal to any of them, no tandem repeats are found and we proceed as for the original SSAHA algorithm. Else *h(i)=h(i+m)* for some *m£s*, and we have found a tandem repeat of motif length *m*. In our example above, we find straightaway that *h(1) = h(1+3)*.

2. Next we find the number of tandem repeats in the sequence: we track forward looking at hash words *h(i+2m), h(i+3m),* until either we find a word *h(i+rm)* that does not equal *h(i)* or *i+rm* lies off the edge of the sequence (in which case the repeated region has lasted to the edge of the query sequence). In our example, we find *r=15*.

3. We will say that the hash word *h(i+c)* of the repetitive region has a *cycle position* of *c*, where *c* varies from 0 to *m-1* inclusive. We query the database for the hits for each of the words *h(c)*, where *c = 0, ... m-1*. For our example we obtain a table of hits as follows

Cycle Position	Word	Subject Position
0	AGCAG	1, 16, 31
1	GCAGC	11, 26, 41
2	CAGCA	6, 21, 36

1. We merge the lists of hits, storing both the hit position *p* and the cycle number *c* of each hit, then sort the list in ascending order of *p* only. In our example, the list of hits is:

   (1, 0), (6, 2), (11, 1), (16, 0), (21, 2), (26, 1), (31, 0), (36, 2), (41, 1)

2. We go through the list of hits looking for runs of hits for which

   $$p(j+1) = p(j)+m$$
   and
   $$c(j+1) = c(j)+s \bmod m.$$

   All hits in each such run are passed on to be processed as per the original SSAHA algorithm. In our example, the whole list of hits forms a single run.
   The reason we check the cycle position as well as the subject position is so we can distinguish between the hits obtained from S and the hits obtained from what we might call an 'anagram' of S, that is, a sequence with the same words as S, but in a different order. An example of an anagram of S is

```
    1     6     11    16    21    26    31    36    41
S'= AGCAG GCAGC CAGCA AGCAG GCAGC CAGCA AGCAG GCAGC CAGCA
```

The corresponding list of hits for S′ is

(1, 0), (6, 1), (11, 2), (16, 0), (21, 1), (26, 2), (31, 0), (36, 1), (41, 2)

which the algorithm recognises as not being a match as the cycle positions do not change in the correct manner.

3. Given a list of hits in the run, we only pass the next x of them to the next stage of the SSAHA algorithm, where $w + (x-1)s < (r-1)m$. This avoids the pitfall that the repetitive region in the subject sequence may have been longer than the repetitive region in Q.

4. If we have not reached the end of the query sequence, we set $i=i+rm$ and continue.