

# Vectorizing the DNAStrng function (work in progress)

Hervé Pagès

December 1, 2007

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>DNAStrng vs BStringViews</b>	<b>1</b>
<b>3</b>	<b>The BStringViews generic function</b>	<b>2</b>
<b>4</b>	<b>Performance</b>	<b>2</b>
<b>5</b>	<b>Loading a FASTA file into a <i>BStringViews</i> object</b>	<b>3</b>
<b>6</b>	<b>Switching between DNA and RNA views</b>	<b>3</b>

## 1 Introduction

This is a short tour on the DNAStrng function vectorization feature.

Feel free to add your own comments.

## 2 DNAStrng vs BStringViews

The `Biostrings2Classes` vignette presents a proposal for 2 new classes (*BString* and *BStringViews*) as a replacement for the *BioString* class currently defined in the *Biostrings* 1 (*Biostrings* v 1.4.x) package.

It also shows how to use the DNAStrng function to create a *DNAStrng* object (a *DNAStrng* object is just a particular case of a *BString* object):

```
> d <- DNAStrng("TTGAAAA-CTC-N")
```

However this function is NOT vectorized: it always returns a *DNAStrng* object (which can only represent a *single* string).

In *Biostrings* 1, the DNAStrng function IS vectorized. Its vectorized form does the following: (1) concatenates the elements of its `src` argument into a single big string, (2) stores the offsets of all these elements in the `offsets` slot.

This behaviour is not immediately obvious to the user, until he looks at the `offsets` slot.

It always returns a *BioString* object (with has as many values as the number of elements passed in the `src` argument).

### 3 The BStringViews generic function

The feature described in the previous section (provided by the vectorized form of the `DNAStrng` function in *Biostrings 1*) is provided in *Biostrings 2* via the `BStringViews` generic function:

```
> v <- BStringViews(c("TTGAAAA-C", "TC-N"), "DNAStrng")
> v
```

Views on a 13-letter `DNAStrng` subject

Subject: TTGAAAA-CTC-N

Views:

	start	end	width	
[1]	1	9	9	[TTGAAAA-C]
[2]	10	13	4	[TC-N]

### 4 Performance

The following example was provided by Wolfgang:

```
> library(hgu95av2probe)
> system.time(z <- BStringViews(hgu95av2probe$sequence, "DNAStrng"))
```

user	system	elapsed
3.912	0.040	3.952

```
> z
```

Views on a 5045000-letter `DNAStrng` subject

Subject: TGGCTCCTGCTGAGGTCCCCTTTCCGCTGTGAA...AAGCCCTCGTGCTCCTTGCAACAGCGCACCCA

Views:

	start	end	width	
[1]	1	25	25	[TGGCTCCTGCTGAGGTCCCCTTTCC]
[2]	26	50	25	[GGCTGTGAATTCCTGTACATATTC]
[3]	51	75	25	[GCTTCAATTCATTATGTTTAAATG]
[4]	76	100	25	[GCCGTTTGACAGAGCATGCTCTGCG]
[5]	101	125	25	[TGACAGAGCATGCTCTGCGTTGTTG]
[6]	126	150	25	[CTCTGCGTTGTTGTTTACCAGCT]
[7]	151	175	25	[GGTTTACCAGCTTCTGCCCTCACA]
[8]	176	200	25	[TTCTGCCCTCACATGCACAGGGATT]
[9]	201	225	25	[CCTCACATGCACAGGGATTTAACAA]
...	...	...	...	...
[201792]	5044776	5044800	25	[GAGTGCCAATTCGATGATGAGTCAG]

```
[201793] 5044801 5044825 25 [ACACTGACACTTGTGCTCCTTGTCA]
[201794] 5044826 5044850 25 [CAATTCGATGATGAGTCAGCAACTG]
[201795] 5044851 5044875 25 [GACTTCTGAGGAGATGGATAGCCT]
[201796] 5044876 5044900 25 [AGATGGATAGCCTTCTGTCAAAGCA]
[201797] 5044901 5044925 25 [ATAGCCTTCTGTCAAAGCATCATCT]
[201798] 5044926 5044950 25 [TTCTGTCAAAGCATCATCTCAACAA]
[201799] 5044951 5044975 25 [CAAAGCATCATCTCAACAAGCCCTC]
[201800] 5044976 5045000 25 [GTGCTCCTTGTCAACAGCGACCCA]
```

With *Biostrings* 1, the call to `DNAStrng(hgu95av2probe$sequence)` takes about 20 minutes... (the implementation of the vectorization feature is quadratic in time, as reported by Wolfgang).

## 5 Loading a FASTA file into a *BStringViews* object

The `read.BStringViews` function can be used to load a FASTA file in a *BStringViews* object:

```
> file <- system.file("Exfiles", "someORF.fsa", package = "Biostrings")
> orf <- read.BStringViews(file, "fasta", "DNAStrng")
> orf
```

Views on a 26339-letter DNAStrng subject

Subject: ACTTGTAATATATATCTTTTATTTTCCGAGAGGAA...TATACATAGGGCTAAGGAAGAAAAAAAATCAC

Views:

	start	end	width	
[1]	1	5573	5573	[ACTTGTAATATATATCTTTTATTTTCC...ACGCTTATCGACCTTATTGTTGATAT]
[2]	5574	11398	5825	[TTCCAAGGCCGATGAATTCGACTCTT...CAGAGTAAATTTTTTCTATTCTCTT]
[3]	11399	14385	2987	[CTTCATGTCAGCCTGCACTTCTGGGT...CGATGGTACTCATGTAGCTGCCTCAT]
[4]	14386	18314	3929	[CACTCATATCGGGGTCTTACTTCCC...ACGTGTCCCGAAACACGAAAAAGTAC]
[5]	18315	20962	2648	[AGAGAAAGAGTTTCACTTCTTGATTA...AAAATATAATTTATGTGTGAACATAG]
[6]	20963	23559	2597	[GTGTCCGGCCCTCGCAGGCGTTCTAC...TTCAAGTTTTGGCAGAATGTACTTTT]
[7]	23560	26339	2780	[CAAGATAATGTCAAAGTTAGTGTCG...AGGGCTAAGGAAGAAAAAAAATCAC]

```
> desc(orf)
```

```
[1] "YAL001C TFC3 SGDID:S0000001, Chr I from 152168-146596, reverse complement, Verified ORF"
[2] "YAL002W VPS8 SGDID:S0000002, Chr I from 142709-148533, Verified ORF"
[3] "YAL003W EFB1 SGDID:S0000003, Chr I from 141176-144162, Verified ORF"
[4] "YAL005C SSA1 SGDID:S0000004, Chr I from 142433-138505, reverse complement, Verified ORF"
[5] "YAL007C ERP2 SGDID:S0000005, Chr I from 139347-136700, reverse complement, Verified ORF"
[6] "YAL008W FUN14 SGDID:S0000006, Chr I from 135916-138512, Verified ORF"
[7] "YAL009W SP07 SGDID:S0000007, Chr I from 134856-137635, Verified ORF"
```

## 6 Switching between DNA and RNA views

The `BStringViews` function can also be used to switch between “DNA” and “RNA” views on the same string:

```
> orf2 <- BStringViews(orf, "RNAString")
```

These conversions are very fast because no string data needs to be copied:

```
> orf[[0]]@data
```

26339-byte XRaw object (starting at address 0x3384298)

```
> orf2[[0]]@data
```

26339-byte XRaw object (starting at address 0x3384298)