

Darn v1.0
A weighted constraint solver for RNA motif localization
User manual

INRA Toulouse, France

February 10, 2010

Contents

1	Presentation	2
2	Install	2
3	Run example	3
4	Arguments description	3
5	Motif description syntax	4
5.1	General shape	4
5.2	List of available cost functions	4
5.3	Changing the costs	7
5.4	The <code>model</code> parameter	7
5.5	Example	8
6	FAQ	8

1 Presentation

Darn is a RNA motif search tool. It finds all the subsequences that match a specified motif on some input genomic sequence(s).

Darn uses a weighted constraint solver to localize the portions of a genomic sequence that match a motif. The motif is expressed in a specific language (see section 5). Some motif descriptors are provided with the software for FMN, Lysine, RNaseP, SAM, snoRNA C/D-box and tRNA search.

The predictions could be available in GFF, FASTA and other formats (see section 4).

Darn can be use on line at this address: <http://carlit.toulouse.inra.fr/Darn>.

Darn was developped by Matthias Zytnicki during his PhD in the BIA lab at INRA Toulouse, France.

Darn is written in C++ for Linux systems and is distributed under the GNU GPL licence. The current version is 1.0 (February 10, 2010).

If you are satisfied to use it, please cite *Darn - A Weighted Constraint Solver for RNA Motif Localization*. M. Zytnicki, C. Gaspin, T. Schiex. Constraints, Volume 13 (2008).

2 Install

From archive package `darn-1.0.tar.gz` only **gcc**, **g++** and **makedepend** are required on the computer.

First, download the software.

Go to this Internet page https://mulcyber.toulouse.inra.fr/frs/?group_id=15.

Download the current file `darn-1.0.tar.gz`.

Then go to the directory containing the file.

Execute the following commands:

```
tar xzvf darn-1.0.tar.gz
cd darn
make
```

To be able to call the software from everywhere, define an environment variable called `DARN_PATH`.

If `<path>` is the absolute path of `darn` directory, execute the command:

```
setenv DARN_PATH <path>/src if you use tcsh shell
export DARN_PATH=<path>/src if you use bash shell
```

To avoid to type this command in each user session, add it to the login shell script in your home directory. Using `tcsh` script, the file is called `.tcshrc`. Using `bash` script, the file is called `.bashrc`.

3 Run example

After the installation, go to the `src` directory and type the following command:

```
darn -s ../example/sample.fa -d ../descriptor/trNA_Bacteria.des -n B -f P
```

You should have the following output.

```
Darn_Pred_1 DARN . 9979 10050 0 + .
GTCCCCTTCGTCTAGAGGCCAGGACACCGCCCTTTCACGGCGGTAACAGGGGTTCGAATCCCCTAGGGGAC
(((((((..(((.....))))).((((.....))))).(((.....))))).(((.....))))))
Darn_Pred_2 DARN . 17079 17159 0 + .
GGTGGGGTTCGCGAGCGCCAAAGGGAGCAGACTGTAAATCTGCCGTCACAGACTTCGAAGGTTCGAATCCTCCCCACC
(((((((..(((.....))))).((((.....))))).(((.....))))).(((.....))))))
Darn_Pred_3 DARN . 17280 17350 0 + .
GCGGGCATCGTATAATGGCTATTACCTCAGCCTTCCAAGCTGATGATGCGGGTTCGATTCCCGTGCCCGC
(((((((..(((.....))))).((((.....))))).(((.....))))).(((.....))))))
Darn_Pred_4 DARN . 51381 51452 0 + .
GTCCCCTTCGTCTAGAGGCCAGGACACCGCCCTTTCACGGCGGTAACAGGGGTTCGAATCCCCTAGGGGAC
(((((((..(((.....))))).((((.....))))).(((.....))))).(((.....))))))
```

4 Arguments description

The syntax to call Darn is the following.

```
darn -d file1 -s file2 [-o outputfile] [-n F|R|B]
      [-x file3 [-v]]
      [-f S|G|P|F] [-t] [-c] [-u] [-e]
      [[-w] [-y drawdirectory] [-z webdrawdirectory]]
      [-V[V[V]]] [-a time]
```

Main options

- d file1: the file that contains the descriptor
- s file2: the file that contains the main stem
- o outputfile: a file to write the solutions (default: standard output)
- n F (default) |R|B: search on the forward(F)/reverse(R)/both(B) strand(s) on the main sequence
- x file3: the file that contains the duplex stem sequence
- v: search from 3' to 5' on the duplex sequence

Output options

- f S (default) |G|P|F: print solutions with specific(S), GFF(G), easy parsable(P), FASTA(F) format
- t: print the secondary structure (only in specific output format)
- c: print solutions with lower cost first (solution score not given in FASTA output format)
- u: print all dominated solutions
- e: print the costs given by the constraints (only in specific output format)

Web output options

- w: output solutions using HTML markups
- y drawdirectory: draw the solution in the given directory

-z webdrawdirectory: WWW address to the directory containing the drawings

Verbosity

- V: verbose search
- VV: very verbose search
- VVV: very, very verbose search
- VVVV: verbose search for software development

Miscellaneous

- a n: alarm, stop after 'n' seconds

5 Motif description syntax

Darn searches in the input sequence(s) all the subsequences that match a motif. This motif has to be described in the following specific language allowing to describe the general shape of a non-protein-coding RNA through constraints.

5.1 General shape

Here is the skeleton of a descriptor:

```
TOP_VALUE = n
POSITIONS X_VAR = Xmin..Xmax, Y_VAR = Ymin..Ymax
# cost functions here
```

The first line gives the overall number of errors the candidate may have, with respect to the signature. This number is n .

On the second line, the X positions refer to positions in the main sequence, whereas the Y positions refer to positions in the target sequence, if any. Thus, if you do not need any target sequence, drop the $Y_VAR = Y_{min}..Y_{max}$ part. This line creates $X_{max} - X_{min} + 1$ (ranging from $X_{X_{min}}$ to $X_{X_{max}}$) positions on the main sequence, and $Y_{max} - Y_{min} + 1$ on the target sequence.

5.2 List of available cost functions

Here are some examples involving every supported cost function. All these functions use a parameter model, which is set to soft by default. We will detail its meaning in subsection 5.4.

PATTERN cost function

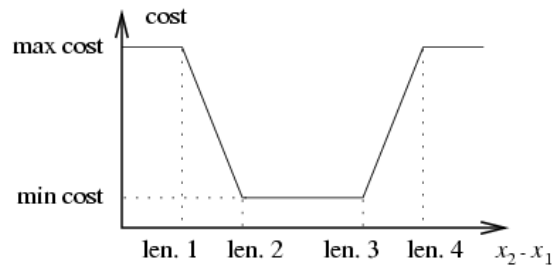
```
PATTERN[word="ACGU",errors=1,model=soft] (X1,X2)
```

Look for the word ACGU starting at position $X1$ and ending at position $X2$, with one possible error. You may use the letters A, C, G, U (or T), N or any ambiguous nucleotide (namely B, D, etc.) to describe the word. Possible errors are insertions, deletions and substitutions.

SPACER cost function

```
SPACER[length=15..20..30..35, costs=1..5, model=soft] (X1, X2)
```

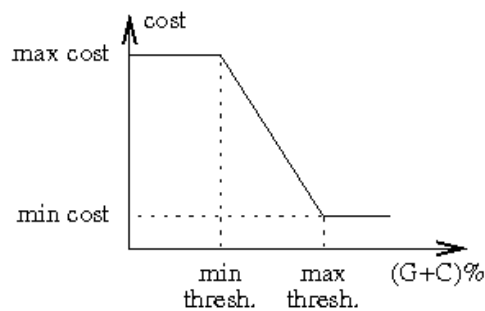
If the distance between positions X1 and X2 is not less than 20, and not greater than 30, then give a cost of 1. If the distance between positions X1 and X2 is less than 15, or greater than 35, then give a cost of 5. Otherwise, it gives a cost between 1 and 5. The spacer cost profile looks like this:



COMPOSITION cost function

```
COMPOSITION[nucleotides="CG", threshold>=60..80, costs=1..5, model=soft] (X1, X2)
```

If the (G+C)% between positions X1 and X2 is not less than 80%, then the given cost is 1. If it is not greater than 60%, then the given cost is 5. Otherwise, the cost is between 1 and 5. The composition cost profile looks like this:



HELIX cost function

```
HELIX[stem=5..7, loop=4..10, errors=1, model=soft] (X1, X2, X3, X4)
```

Set the presence of an helix formed by a stem between positions X1 and X2 on the one hand, and a stem between positions X3 and X4 on the other hand. The stem should contain not less than 5 base pairings, and not more than 7 base pairings. The loop should contain not less than 4 nucleotides, and not more than 10 nucleotides. One substitution is allowed in the stem. If you also want to allow insertions and deletions, add `indels=yes` to the list of parameters. Only canonic interactions (i.e. A-U and G-C) are allowed. If you want to allow wobble interactions (i.e. G-U), add `wobble=yes` to the list of parameters.

REPETITION cost function

```
REPETITION[size=5..7, distance=4..10, errors=1, model=soft] (X1, X2, X3, X4)
```

Set that the subsequence between positions X1 and X2 is repeated between positions X3 and X4. The size parameter give the size of the repeated subsequence, and the

`distance` parameter gives the distance between the two subsequences. The number of allowed differences between the subsequences is given by the parameter `error`, and the optional parameter `indels` states whether indels are allowed.

PAIR cost function

```
PAIR[interaction=WATSON_CRICK,interaction=SUGAR,orientation=TRANS,  
family=2,model=soft] (X1,X2)
```

Set a non canonic pairing between the nucleotides at positions X1 and X2. The nucleotide at position X1 interacts with its Watson-Crick side ; the nucleotide at position X2 interacts with its Sugar side. This is a *trans* interaction, in the geometric family 2, as defined by Leontis *et al.* If you wish to accept all geometric families, given two sides and an orientation, write `family=all`.

DUPLEX cost function

```
DUPLEX[errors=1,model=soft] (X1,X2,Y1,Y2)
```

Set the presence of a duplex formed by a stem between positions X1 and X2 on the main sequence, and a stem between positions Y1 and Y2 on the target sequence. The duplex may contain a substitution, an insertion, or a deletion. Notice that it is a time consuming constraint, so set the distance between X1 and X2, and between Y1 and Y2, and do not use high numbers of errors. If you want to allow wobble interactions, add `wobble=yes` to the list of parameters.

5.3 Changing the costs

For the PATTERN, HELIX, DUPLEX and REPETITION cost functions, you can also change the costs given by the functions. For example, if you write:

```
PATTERN[word="ACGU",errors=1,model=soft] (X1,X2) {0,2}
```

then the cost given for no error is 0 (as usual). If one error is found, a cost of 2 is given.

5.4 The model parameter

The PATTERN, HELIX, DUPLEX and REPETITION functions all have an `errors` parameter. If the number of errors found is greater than the value of `errors`, then the element of structure is not accepted.

However, if you use `model=optional`, then every cost greater than `errors` will be set to `errors`.

If you use `model=hard`, then every cost greater than `errors` will not be accepted (like the `soft` option), but any cost not greater than `errors` will be set to zero.

The syntax of the functions SPACER and COMPOSITION with `model=hard` changes a bit. The SPACER only uses two lengths (the minimum and the maximum lengths), and the COMPOSITION only uses one threshold. Both of them do not use the `costs` parameter.

5.5 Example

Here is a little example, describing an H/ACA box sRNA

```
# an H/ACA box sRNA
TOP_VALUE = 3
VARIABLES X_VAR = 1...18, Y_VAR = 1...4

PATTERN[word="RA",err=0,model=hard] (X7,X8)
PATTERN[word="GA",err=1,model=optional] (X7,X8)
PATTERN[word="RUGA",err=0,model=hard] (X9,X10)
PATTERN[word="ANA",err=0,model=hard] (X17,X18)
PATTERN[word="NUNN",err=0,model=hard] (Y2,Y3)

HELIX[stem=7..10,loop=30..65,err=1,model=soft] (X1,X2,X15,X16)

DUPLEX[err=1,model=soft] (X3,X4,Y1,Y2)
DUPLEX[err=1,model=soft] (X13,X14,Y3,Y4)
COMPOSITION[nucleotides="GC",threshold>=75..100%,costs=0..2,model=soft] (X5,X6)
COMPOSITION[nucleotides="GC",threshold>=70..100%,costs=0..2,model=soft] (X5,X6)
COMPOSITION[nucleotides="GC",threshold>=65..100%,costs=0..2,model=soft] (X11,X12)

SPACER[length=0..8,model=hard] (X2,X3)
SPACER[length=3..6,model=hard] (X3,X4)
SPACER[length=1..1,model=hard] (X4,X5)
SPACER[length=6..8,model=hard] (X5,X6)
SPACER[length=1..1,model=hard] (X6,X7)
SPACER[length=3..30,model=hard] (X8,X9)
SPACER[length=1..1,model=hard] (X10,X11)
SPACER[length=4..8,model=hard] (X10,X12)
SPACER[length=1..1,model=hard] (X12,X13)
SPACER[length=3..6,model=hard] (X13,X14)
SPACER[length=0..8,model=hard] (X14,X15)
SPACER[length=0..1,model=hard] (X16,X17)
SPACER[length=3..6,model=hard] (Y1,Y2)
SPACER[length=3..6,model=hard] (Y3,Y4)
SPACER[length=10..12,model=hard] (Y1,Y4)
```

6 FAQ

How to report a bug or a feature required ?

Use the Bug and Feature Tracker of the project.

So go to the following page

https://mulcyber.toulouse.inra.fr/tracker/?atid=528&group_id=15&func=browse .

Click on the 'New' link, fill the description frame and submit the request.

I can't access directly pages related to Darn.

Darn is hosted by a software Forge `mulcyber.toulouse.inra.fr` with secure access. It is necessary to accept the certificate to access this Forge pages, for example the Download page.

Error message: Error! Cannot open file "" for cost values!

By default, Darn is looking for a file `costs.dat` in the current directory.

To be able to call Darn from everywhere, define an environment variable `DARN_PATH`.

Using tcsh shell, type: `setenv DARN_PATH src_directory`

Using bash shell, type: `export DARN_PATH=src_directory`

To avoid to type this command in each session, add it to the appropriate script.

Using tcsh shell, add the line in the `~/ .tcshrc` file.

Using bash shell, add the line in the `~/ .bashrc` file.