Unipro UGENE Query Designer Manual

Version 1.30

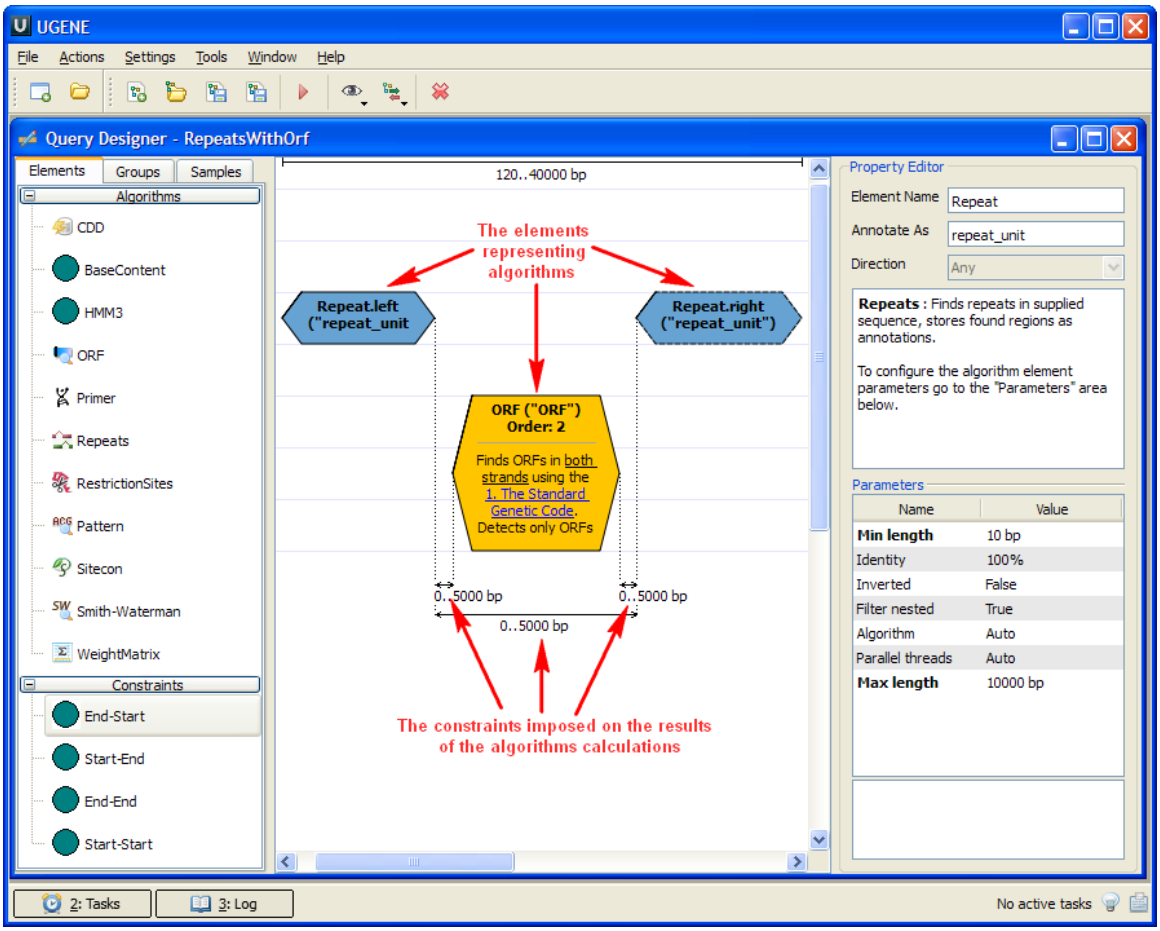
May 28, 2018

Query Designer Documentation

About the Query Designer

The *Query Designer* is a part of UGENE genome analysis suite that allows a molecular biologist to analyze a nucleotide sequence using different algorithms (Repeats finder, ORF finder, Weight matrix matching, etc.) at the same time imposing constraints on the positional relationship of the results obtained from the algorithms.

A user-friendly interface is used to create a schema of the algorithms and constraints.



Alternatively, you can create / edit a schema using a text editor.

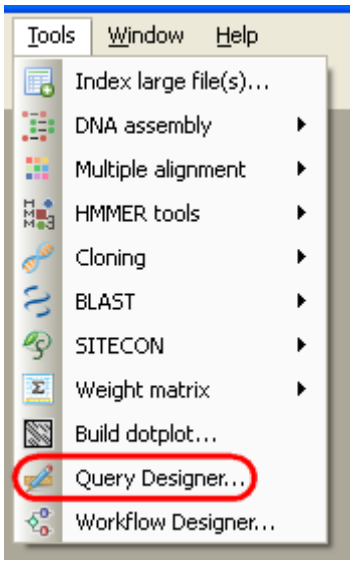
When the schema has been created and all its parameters have been set you can run it for a nucleotide sequence. The results are saved as a set of annotations to the specified file in the GenBank format.

Introduction

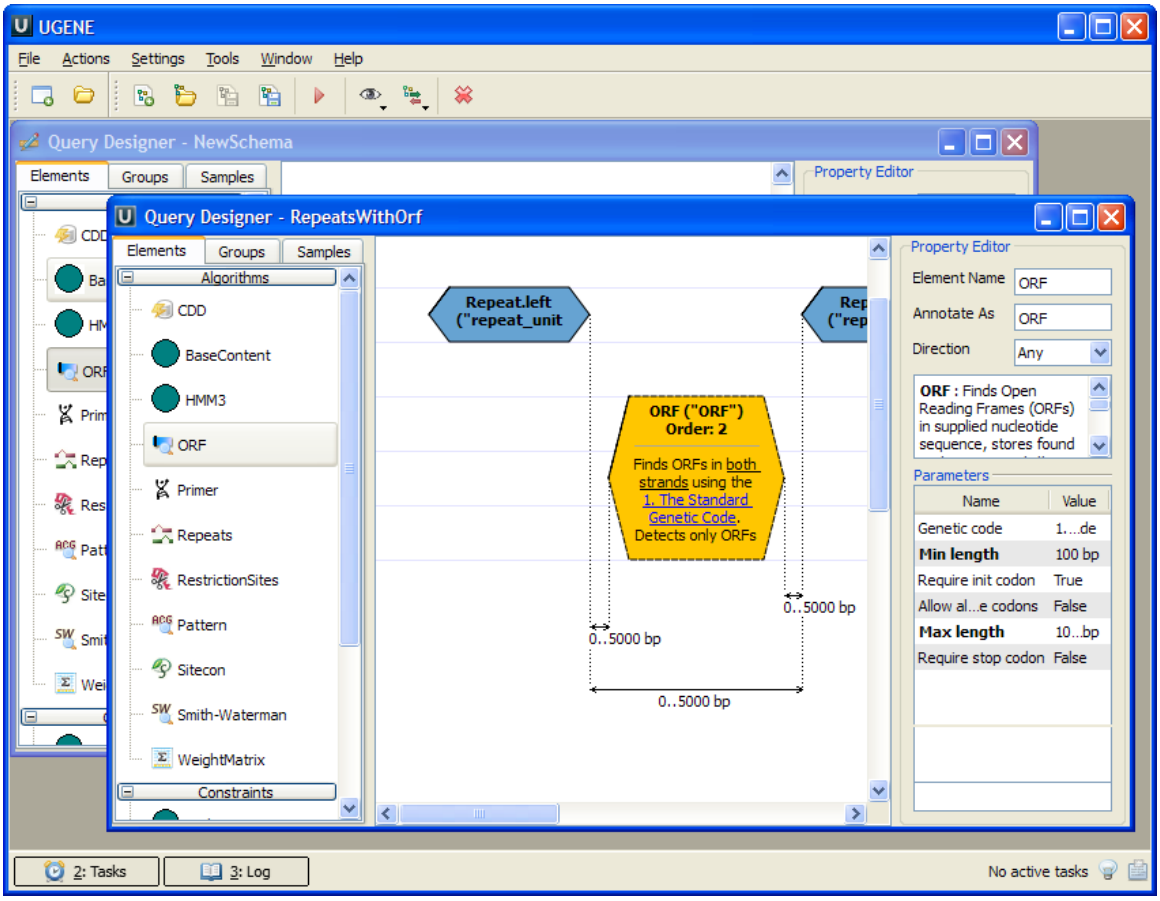
This chapter describes the *Query Designer* key elements and provides an example on how to create and run a simple *schema*.

Launching the Query Designer

To launch the *Query Designer* select the *Tools Query Designer* item in the main menu:



The tool provides multi-window user interface, so you can open and use at the same time as many *Query Designer* windows as you need.



Terminology

Query Designer Window Components

Each *Query Designer* window consists of:

Palette

The palette of the *elements*.

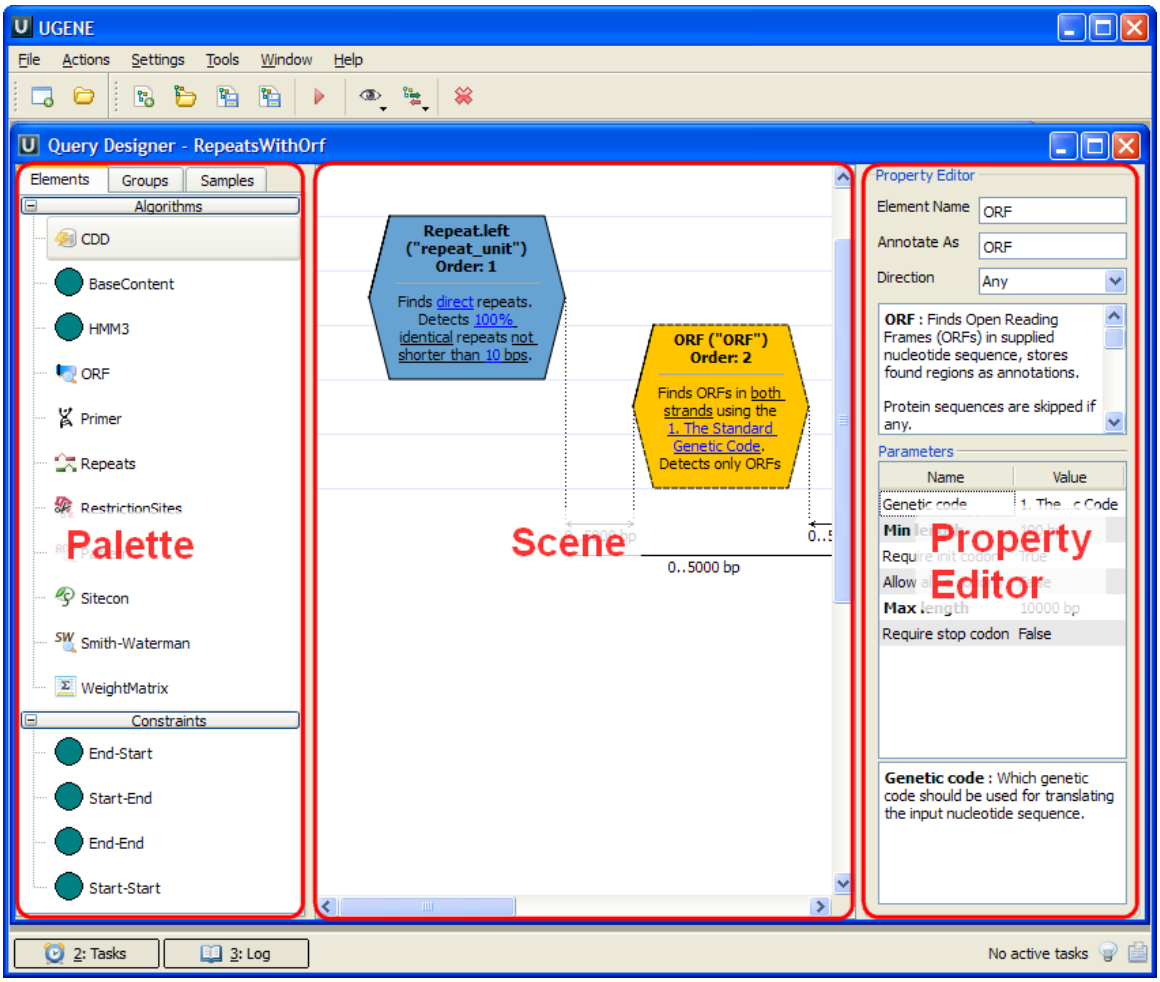
Scene

The main drawing scene is the place where *elements* are constructed into a *schema*.

Property Editor

Provides information about a currently selected *element* and allows configuring it.

On the image below you can see these components in a maximized *Query Designer* window:



All these components are resizable and can be adjusted to individual needs.

Schema Elements

The *Scene* is initially empty and you start with creating a *schema* on it:

schema

A schema is a visual representation of the query that would be used to analyze a nucleotide sequence. It consists of *elements*.

element

An element of a *schema*. There are two types of elements: *algorithm elements* and *constraint elements*.

algorithm element

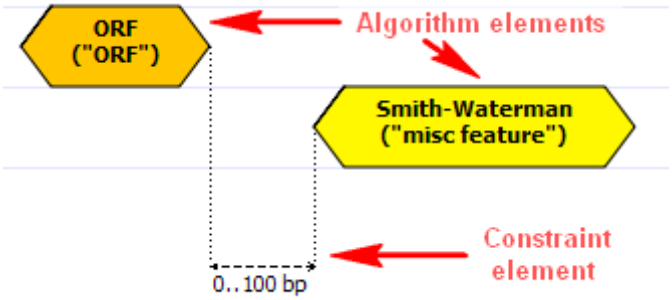
An element of a *schema* that represents an algorithm used to analyze a nucleotide sequence.

constraint element

An element of a *schema* used to impose constraints on the results obtained from *algorithm elements*.

Your first step is to *add necessary algorithm elements* to the *Scene* by dragging them to the *Palette*.

The next step is to *add constraints* on the algorithms results.

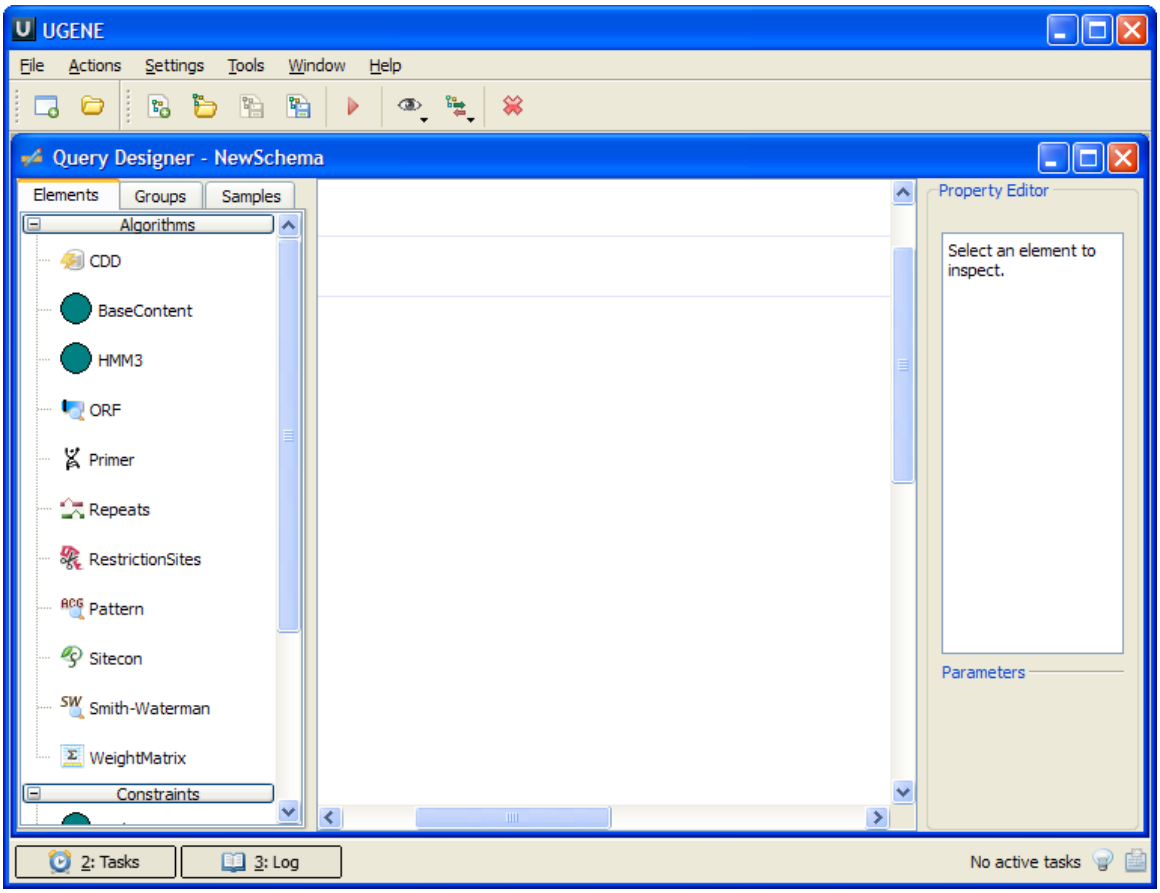


Read *Manipulating Element* chapter to learn the details or check the section below on *how to create and run a schema*.

How to Create and Run Schema

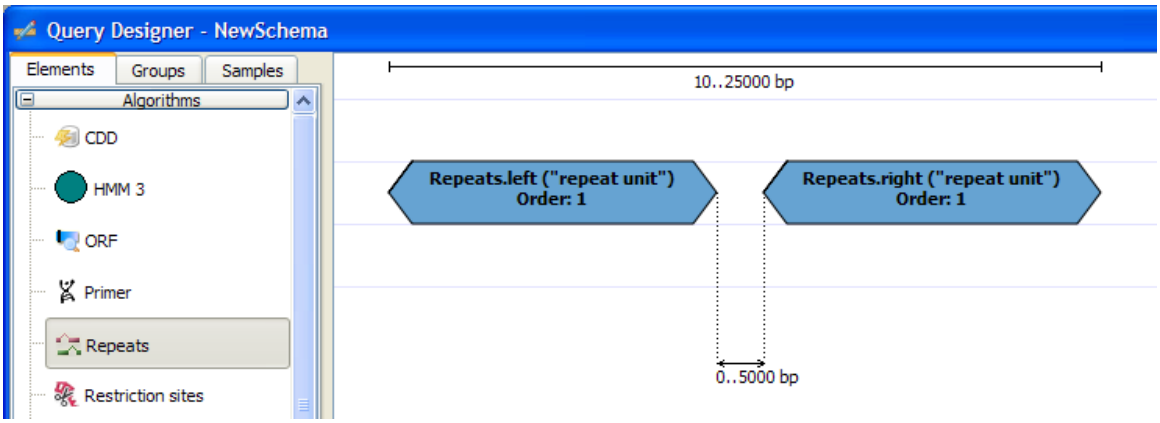
- Select the *Tools Query Designer* item in the main menu.

  Result: The Query Designer window appears:



- Drag the *Repeats* element from the *Palette* to the *Scene*.

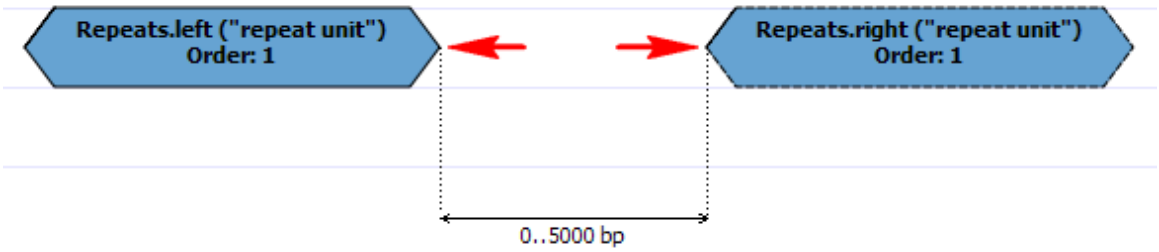
Result: The *Repeats* element subunits are presented on the *Scene*:



Note, that by default minimum distance between left and right repeats is 0bp, the maximum distance is 5000bp.

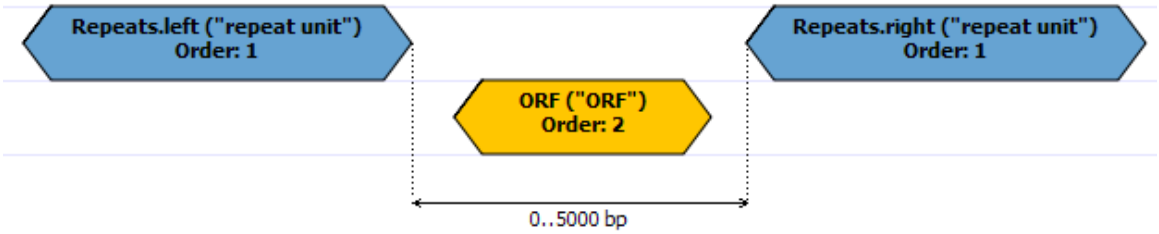
- Slide the *Repeats* element subunits apart.

Result:

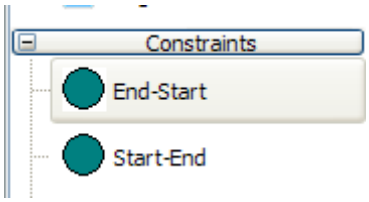


- Drag the *ORF* element from the *Palette* and drop it between the repeats.

Result: The *ORF* element is presented on the *Scene*:

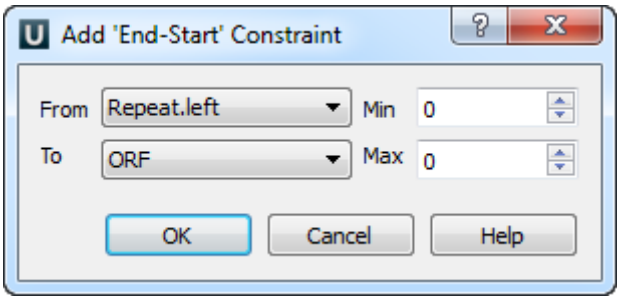


- Find the *End-Start* constraint on the *Palette*:



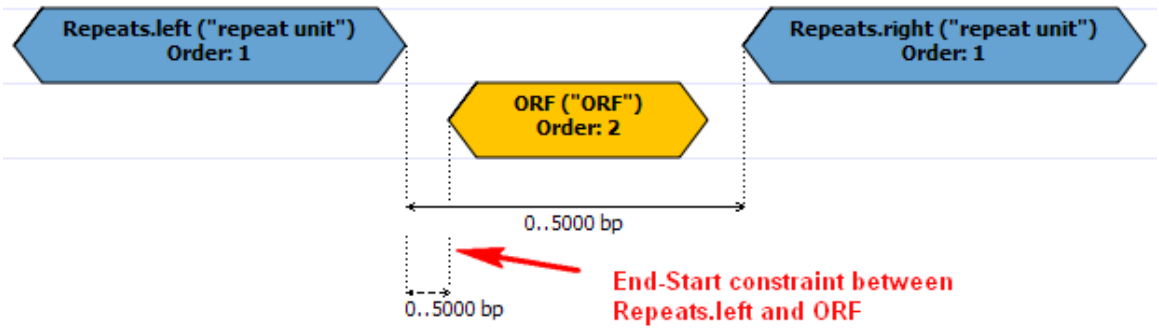
And drag it between the left *Repeats* element and *ORF*.

Result: The dialog appears:



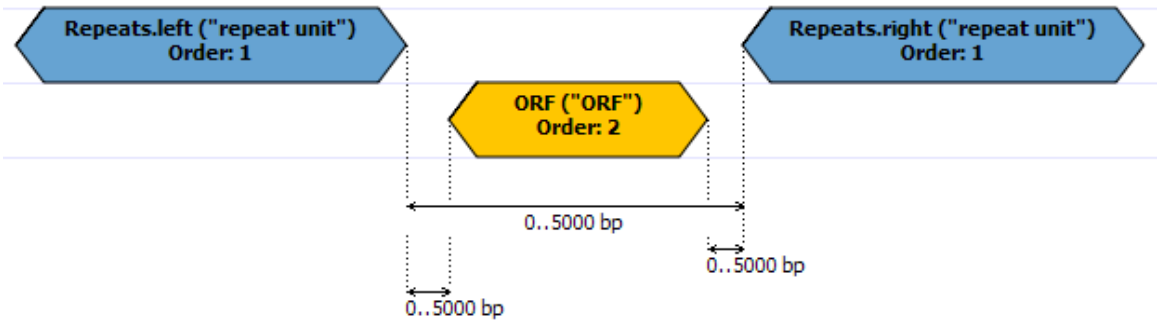
- Check that *From* equals to *Repeats.left* and *To* equals to *ORF*. Set *Max* to 5000. Press the *OK* button.

Result: The constraint has been added to the elements:

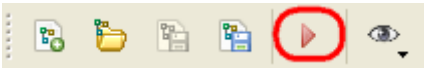


- Repeat steps 5-6 for *ORF* element and the right *Repeats* subunit.

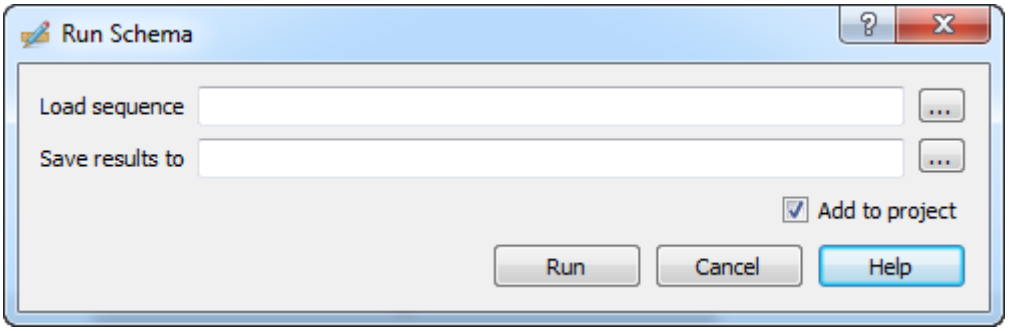
Result: The schema looks as follows:



- Press the *Run Schema* button on the toolbar:

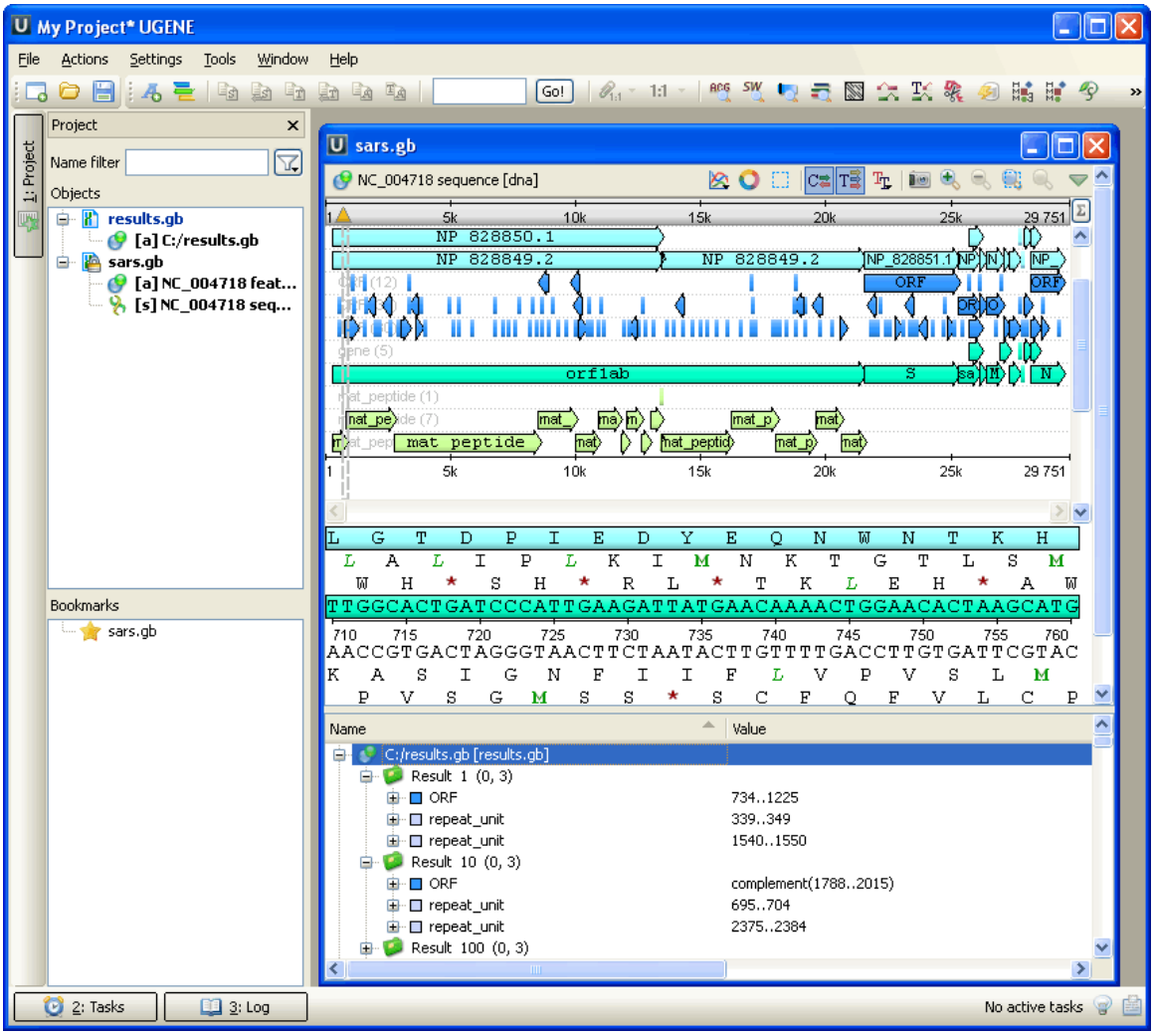


Result: The *Run schema* dialog appears:



- Browse for the sequence to analyze (the *Load sequence* field) and for a GenBank file to save results to (the *Save results to* field). Click the *Run* button.

Result: Both the sequence and the file with annotations are added to the current project. The sequence appears in the Sequence View window, for example:



To learn more about the Sequence View read main UGENE User Manual.

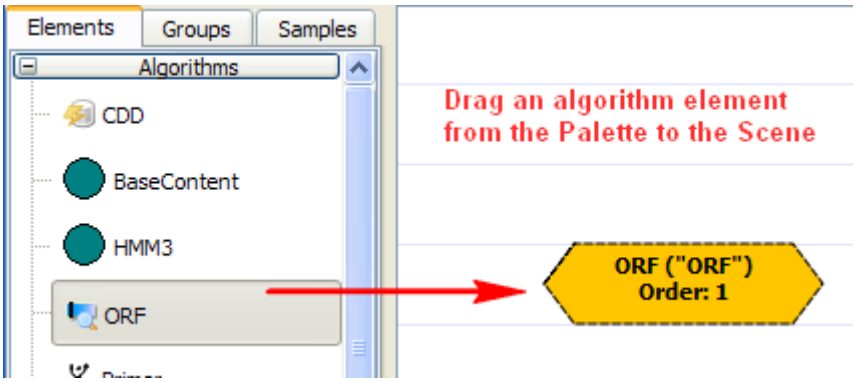
The schema described in this example is also available as sample schema. Select the *Samples* tab on the *Palette* and double-click the *ORF-Repeats* to open the sample schema.

Manipulating Element

This chapter describes in more details how you can manipulate *elements* using the GUI.

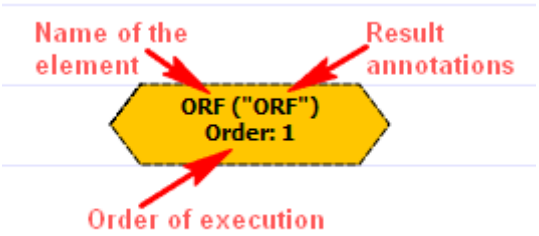
Adding Algorithm Element

To add an *algorithm element* drag it from the *Palette* to the *Scene*. Or you can just click on the element on the *Palette* and then click somewhere on the *Scene*.

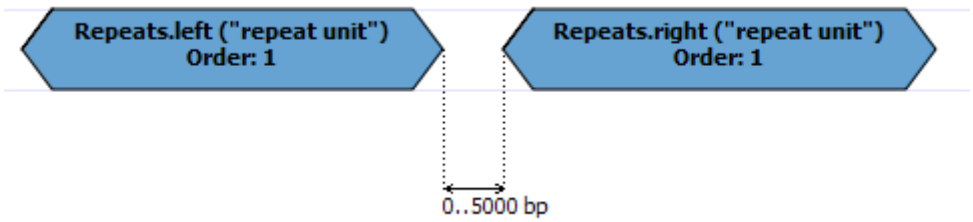


If the default elements appearance hasn't been *modified*, on the added element you can see:

1. The element name
2. The result annotations name, enclosed in parentheses
3. The order of the algorithm execution



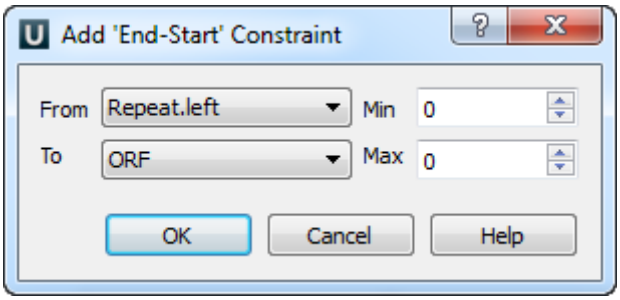
Some elements consist of two subunits. In this case the subunits are marked as **.left** and **.right**:



Adding Constraint Element

A constraint can be imposed on any two *algorithm elements*.

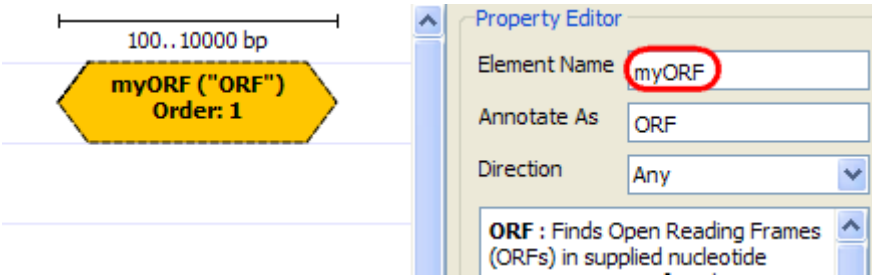
To add a *constraint element* drag it from the *Palette* to the *Scene*. The corresponding dialog appears:



Once the *algorithm elements* are selected and the *OK* button is pressed in the dialog, the *constraint element* is added to the *Scene*. From now the constraint is binded to the *algorithm elements* and it is not possible to modify this assignment.

Renaming Algorithm Element

To rename an *algorithm element*, select it and edit the *Element name* field in the *Property Editor*.



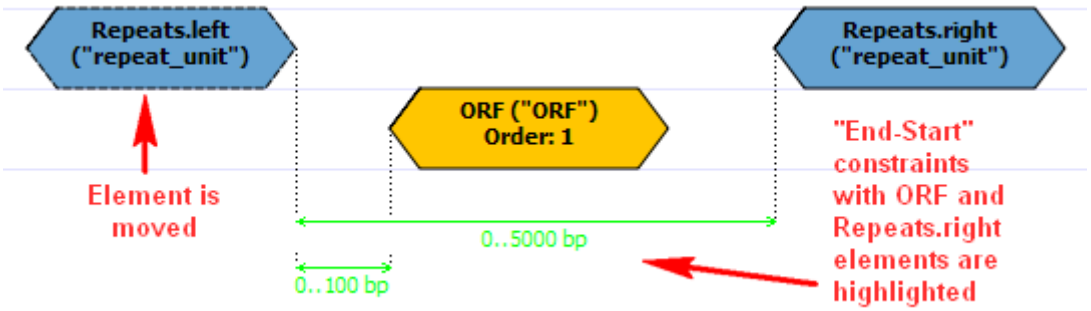
The name can contain only english letters, numbers, '_' (underscore) and '-' (hyphen) characters.

Resizing and Moving Algorithm Element

An *algorithm element* can be moved around on the *Scene* by dragging it and can be resized by dragging its borders.

But if some constraints are imposed on the element, then the element can only be moved / resized within the bounds of the constraints.

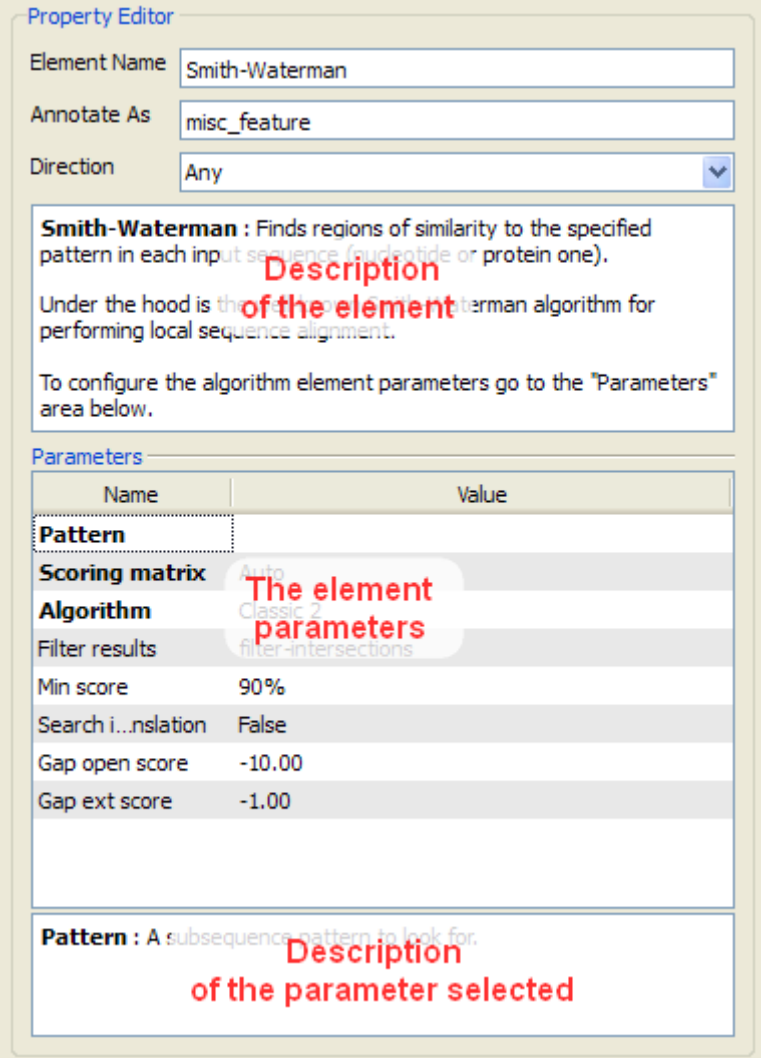
On the image below you can see constraints highlighted in green when an element is moved:



Managing of Elements Parameters

The parameters values can be edited for the both *algorithm* and *constraint* elements.

When you select an *element* on the *Scene*, it's description and parameters appear in the *Property Editor*.



There are some common parameters:

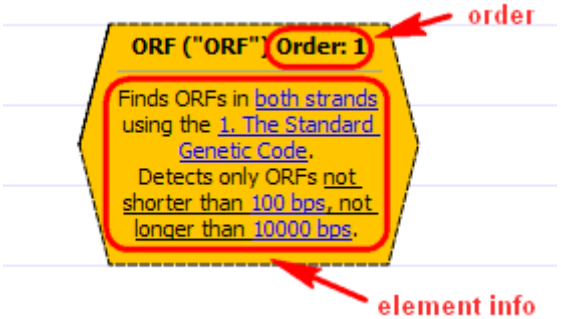
- *Element Name* sets the name of the current element.
- *Annotate As* sets the name of the result annotations.
- *Direction* specifies the direction of the current element relatively to other elements in the schema (i.e. either the result must be searched in the current strand of the input sequence or in the reverse complementary strand). To learn more read *Managing Strands* .

Other parameters can be found in the *Parameters* area. Depending on the parameter in question, the value is selected either from a drop-down list or a spinbox, etc.

Changing Algorithm Element Appearance

You can select to either show or hide an *algorithm element* **info** and **order** on the *Scene* by checking / unchecking the *Show element info* an d *Show order* items in the *View Mode* menu (to open the menu select either *View Mode* toolbar button or select *Actions View Mode*).

On the image below you can see an ORF element when the both items are checked:



See also the *Setting Order of Algorithms Execution* section.

Deleting Element

To delete one or more *algorithm* or *constraint* elements from a *schema*, select them on the *Scene* and press the Delete key.

Alternatively, you can use the *Actions Delete* item in the main menu or the *Delete* toolbar button.

Manipulating Schema

This chapter describes how to manipulate a *schema* using the GUI: how to create, to load, to save and, finally, to run one and so on.

Creating New Schema

A new *schema* is created each time you *launch the Query Designer*.

To create a new *schema* from the *Query Designer* select either the *Actions New Schema* item in the main menu or the *New Schema* toolbar button.

Loading Schema

To load an existing *schema* from the *Query Designer*, select the *Actions Load Schema* item in the main menu or use the *Load schema* toolb ar button.

The *Load Schema* dialog appears. Browse for the required schema file with the *.uql extension.

Saving Schema

To save a *schema* from the *Query Designer* select the *Actions Save Schema* item in the main menu or the *Save Schema* toolbar button.

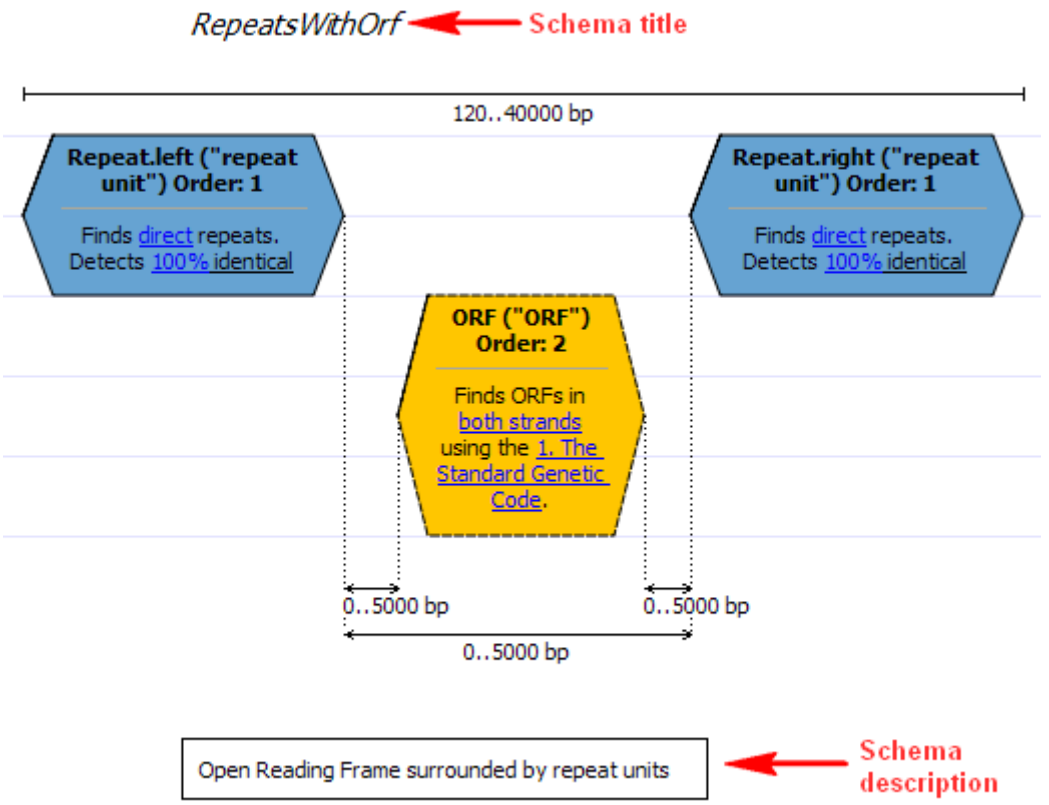
When you save a schema for the first time the *Save Schema* dialog appears where you should input a name and browse for a location of the schema file. The schema is saved to a file with the *.uql extension.

To save a schema to a different file select the *Save Schema As* item on the toolbar or in the *Actions* main menu.

Changing Schema Appearance

You can select to either show or hide a *schema* **title** and **description** on the *Scene* by checking / unchecking the *Show title* and *Show description* items in the *View Mode* toolbar menu.

On the image below the both **title** and **description** are displayed:



Selecting an element direction changes its appearance. See *Managing Strands* for details.

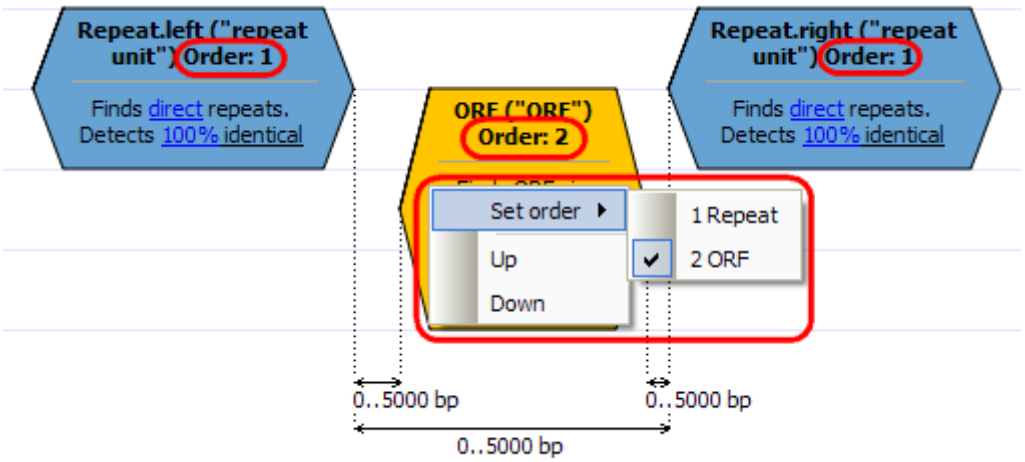
Setting Order of Algorithms Execution

By default, the algorithms are executed in the same order as corresponding *algorithm elements* have been added to the *schema*:

1. The element with order 1 is executed.
2. The element with order 2 is executed. It uses the results obtained from the step 1.
3. And so on.

As each following algorithm element uses the results obtained from the previous element, setting the order can affect efficiency. For example, the less results are obtained on the first step, the faster the second algorithm is executed.

To change the order use the *Set order* submenu or *Up* / *Down* items in an *algorithm element* context menu.



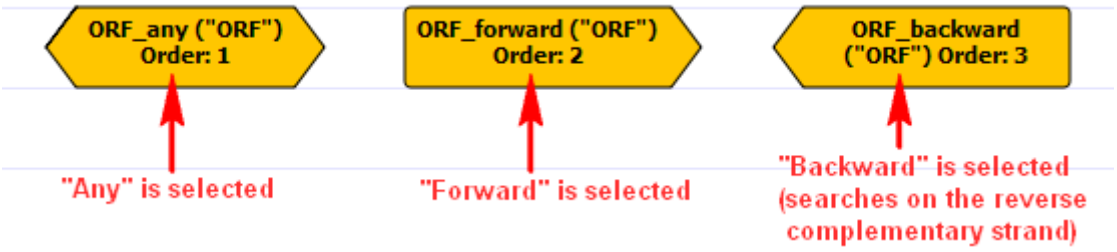
Managing Strands

Element Direction in Schema

For some *algorithm elements* (e.g. *ORF*) the *Direction* parameter is available in the *Property Editor*. It specifies the direction of the current element relatively to other elements in the schema and can take the following values:

- *Direct* — specifies to search the results for the element in the current strand.
- *Forward* — specifiesd to search the results for the element in the reverse complementary strand.
- *Any* — the results for the element are seached in the both strands.

Notice that an element changes it's appearance on the *Scene* when different values are selected:

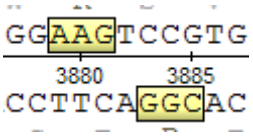


Example1:

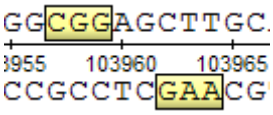
Create the following schema:

1. The Smith-Waterman *algorithm element* with *AAG* patern and the *Forward* direction.
2. The Smith-Waterman element with *CGG* pattern and the *Backward* direction.
3. Add a *constraint* to these elements.
4. Run the schema for a sequence.

By default, if the *Query Sequence Mode* hasn't been *modified*, the followings results will be found:



and



Querying Sequence Strands

As the **Example1** above shows, both sequence strands are queried by default.

To modify this behavior select the *Actions* → *Query Sequence Mode* item in the main menu or *Query Sequence Mode* toolbar button. You can choose between the following values:

- *Direct strand* — the search is performed for the direct strand only.

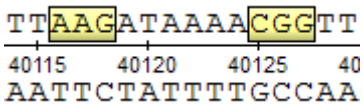
  Note, that the results can still be found in the complement strand if you have set the *Any* or *Backward* direction for an element.
- *Reverse complementary strand* — the search is performed for the reverse complementary strand.
- *Both strands* — the search is performed for the both strands.

Example2:

Create the following schema:

1. The Smith-Waterman *algorithm element* with *AAG* patern and the *Forward* direction.
2. The Smith-Waterman element with *CGG* pattern and the *Forward* direction.
3. Add a *constraint* to these elements.
4. Set the *Query Sequence Mode* to *Direct strand*.
5. Run the schema for a sequence.

Only the following results will be found:



Running Schema from the Query Designer

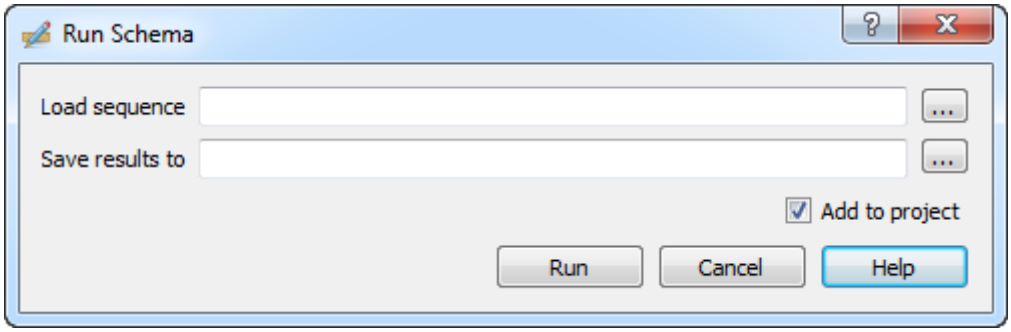
As soon as a *schema* construction is finished:

- all required *algorithm elements are added*
- all required *constraints are set*
- all required *parameters are set*

the schema can be run.

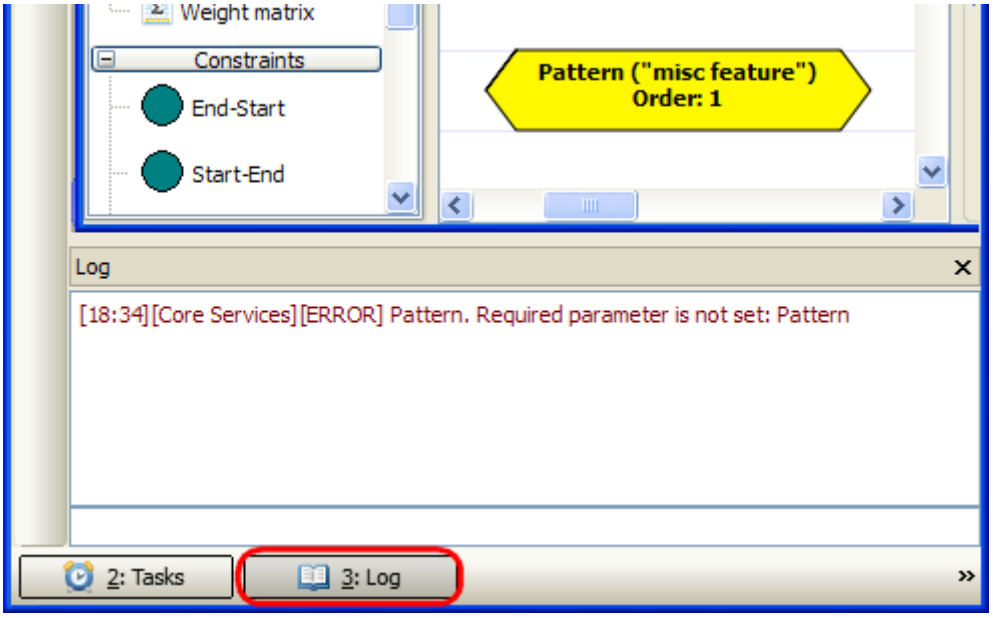
To run the schema select the *Actions Run Schema* item in the main menu or select the *Run Schema* toolbar button.

The *Run Schema* dialog will appear:



Select the *Sequence* to analyze with the schema and browse for the file in GenBank format to *Save results to*. Optionally, check the *Add to project* button to add the created result file to the active project. Press the *Run* button.

In case of an error check the Log View:



Note also that the schema execution task can be canceled from the Task View.

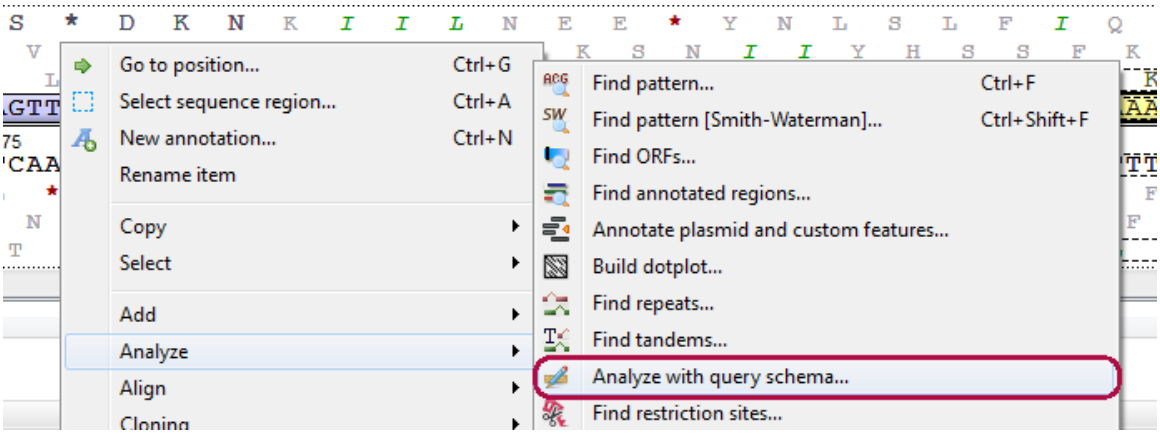
Running Schema from the Sequence View

Prepare a query *schema* and *save* it to a file.

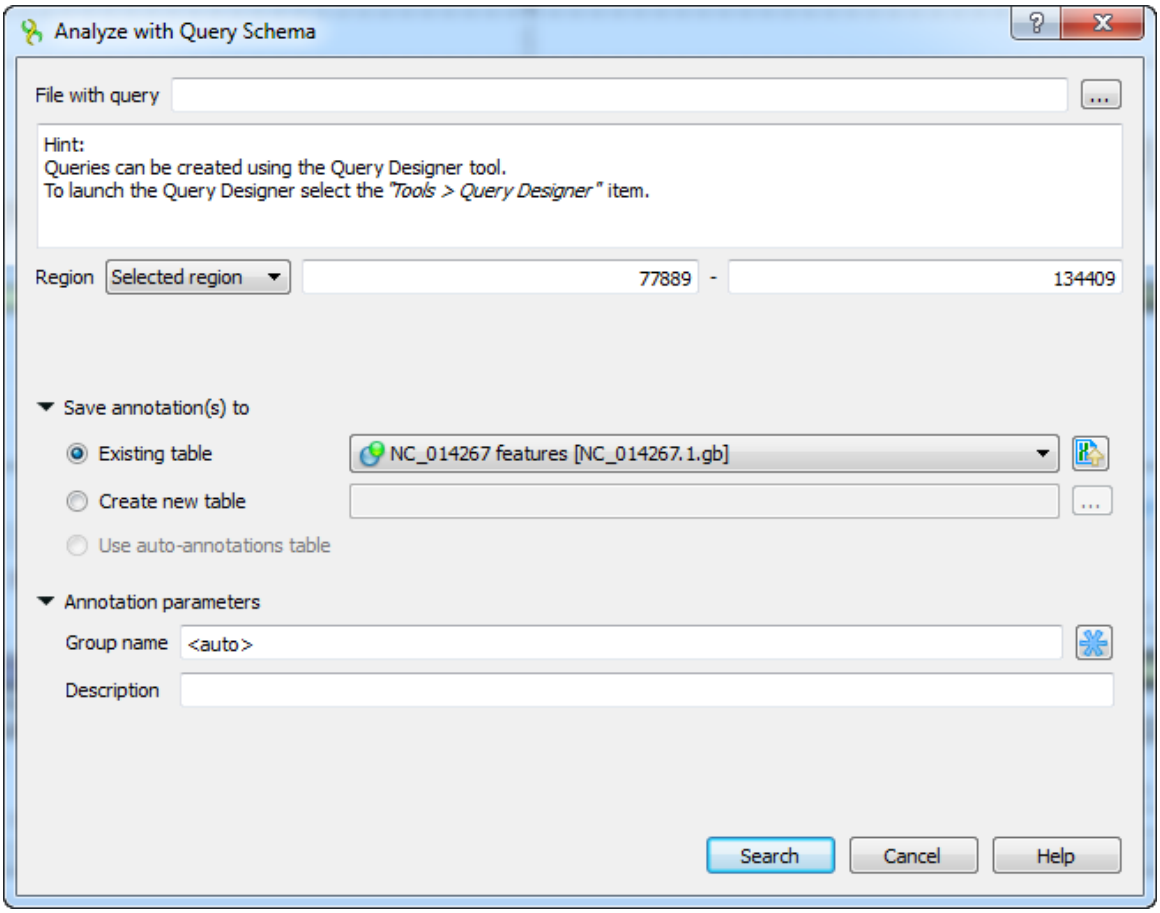
Open a nucleotide sequence that you want to analyze with this query schema. You can see the sequence displayed in the **Sequence view**.

> To learn more about the **Sequence view** read the main UGENE User Manual.

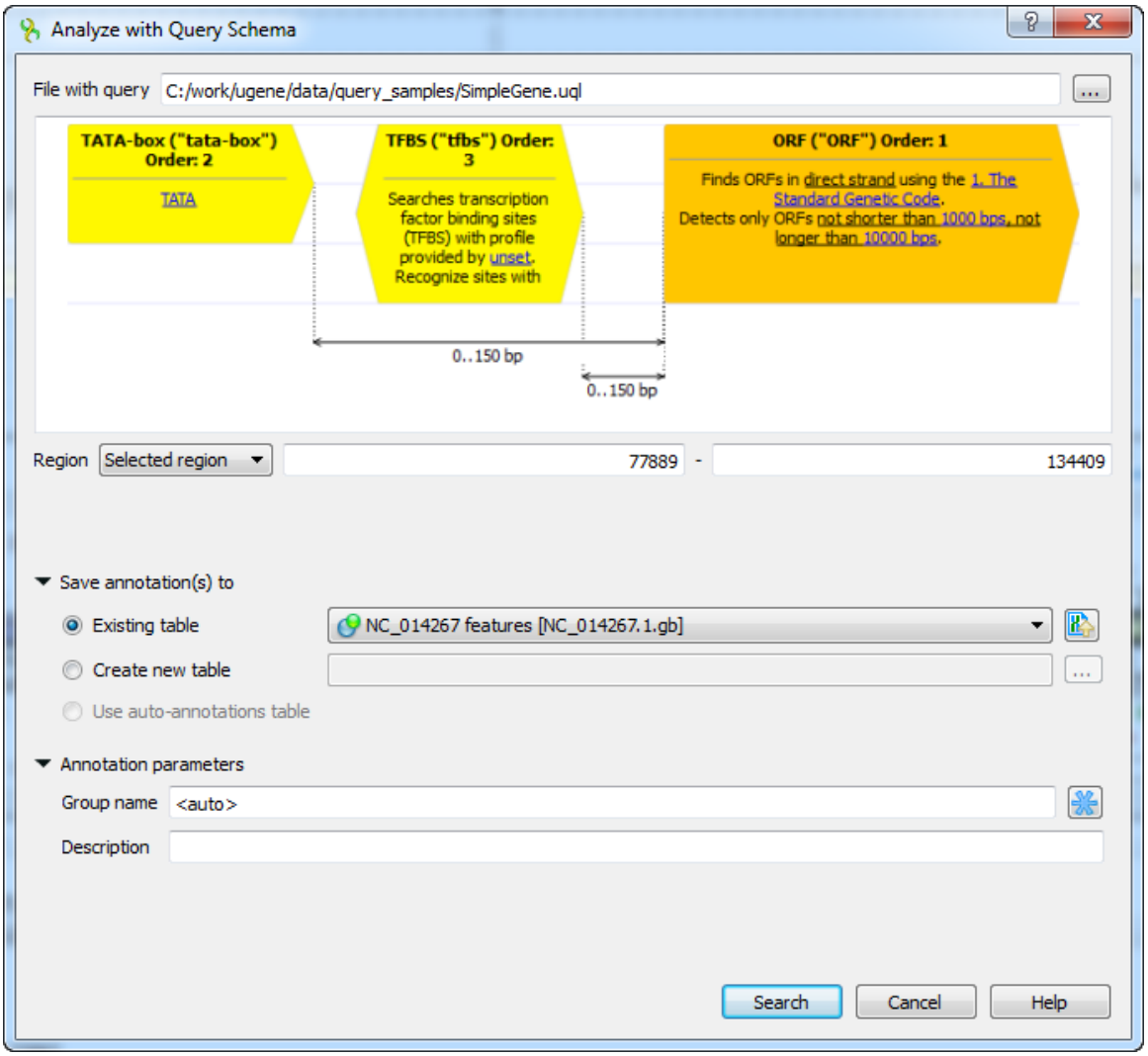
Select the *Analyze Find query designer pattern* item in the *Actions* main menu or in the context menu:



The *Analyze with query schema* dialog appears:



Browse for the file with a query schema. The selected schema preview appears in the dialog, for example:



You can also adjust other parameters:

Region — the sequence range to analyze with the query schema, you can select:

- *Whole sequence* — to analyze the whole sequence.
- *Selected range* — to analyze the currently selected sequence region. This item is disabled if there is no region selected.
- *Custom range* — to specify manually a range to analyze.

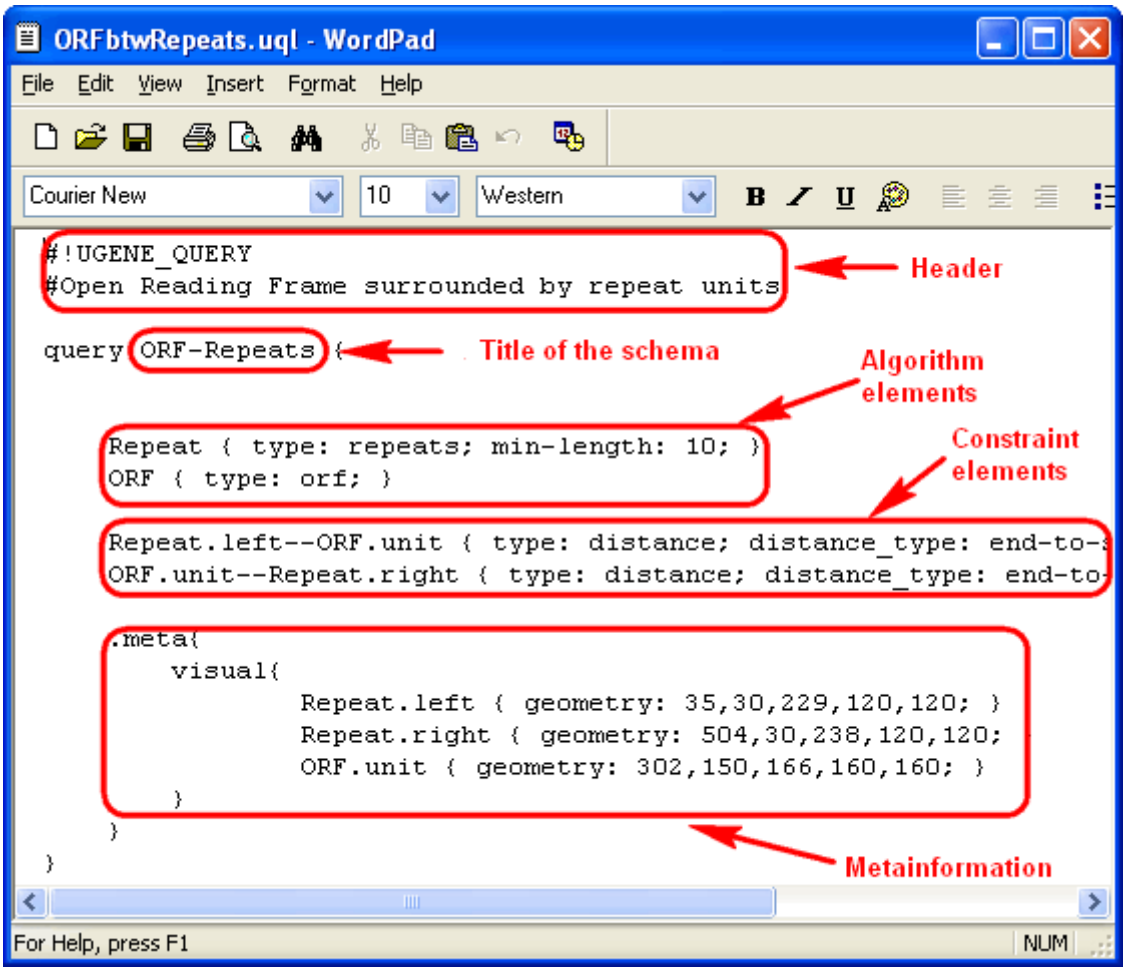
In the *Save annotation(s) to* group you can set up a file to store annotations. It could be either an existing annotation table object, a new annotation table or auto-annotations table (if it is available).

In the *Annotation parameters* group you can specify the name of the group. If the group name is set to <auto> UGENE will use the group name as the name for the group. Also you can add a description in the corresponding text field.

Query Designer Schema File Format

Using the GUI is not the only way to create / edit a *schema*. As specified *earlier* a schema is saved to a file with *.uql extension. This chapter describes the format of the file and explains how you can create / edit a schema file using a text editor.

The best way to learn schema file format is to study an existent *.uql file. For example, let's take the sample schema file described in the *example*. Open the file in a text editor. On the image below you can see the file opened in the MS WordPad.



The file consists of the header and the body. Check the description of each part below.

Header

The header consists of the following key string:

```
#!UGENE_QUERY
```

And the description of the schema:

```
# Write here the description of the schema.
```

Body

The body begins with the **query** keyword followed by the title of the schema and curly braces:

```
query schema_title {

  # Description of algorithm and constraint elements

  # Metainformation

}
```

Within the body you describe the required *algorithm* and *constraint* elements. The metainformation block can also be presented automatically added by the *Query Designer*.

Element Description

An *element* description consists of the element name and a set of parameters enclosed in curly braces. A parameter and the value are separated by ':', different parameters are separated by ';':

```
element_name {

    parameter1: value1;
    parameter2: value2;
    ...
}
```

A parameter's type can be one of the following:

string

A string.

numeric

A number.

boolean

A boolean data type.

Available values are *true* / *false*, *0* / *1* and *yes* / *no*.

Algorithm Element Description

When you describe an algorithm element you must specify the **element_name** and the element type.

The element name corresponds to the element's *Element Name* in the GUI. The name can contain only english letters, numbers, '_' (underscore) and '-' (hyphen) characters.

See, for example, the description of an ORF element:

```
myORF {

   type: orf;

   # Other parameters
}
```

To find the type and the parameters available for the required algorithm element refer to *Algorithm Elements* chapter.

Constraint Element Description

When you describe a constraint element the **element_name** consists of two parts separated by two hyphens.

```
part1--part2
```

Each part represents one of the algorithms the constraint is imposed on.

If the algorithm is presented as one element on a schema (like *ORF*, *Pattern*) the algorithm's part has format:

```
algorithm_element_name.unit
```

If the algorithm is presented as two subelements on a schema (like *Repeats*, *Primer*) the algorithm's part has format:

```
algorithm_element_name.left
```

or:

```
algorithm_element_name.right
```

depending on the subelement the constraint is imposed on.

Also you should specify the constraint type parameter (currently the only available type is *distance*):

```
type: distance;
```

And specify one of the distance types, for example:

```
distance-type: end-to-start;
```

Example1: The constraint is imposed on **myORF** and **myPattern** algorithm elements:

```
myORF.unit--myPattern.unit {

   type: distance;
   distance-type: start-to-start;

   # Other parameters
}
```

Example2: The constraint is imposed on **myORF** algorithm element and the left **myRepeats** algorithm subelement:

```
myORF.unit--myRepeats.left {

   type: distance;
   distance-type: start-to-end;

   # Other parameters
}
```

The available constraint elements are described in the *Constraint Elements* chapter.

Metainformation

The metainformation is added when you create / edit the schema with the *Query Designer*. It is not required for running the schema and is skipped when the schema is created manually, for example:

```
#Open Reading Frame surrounded by repeat units

query ORF-Repeats {

    Repeat {
        type: repeats;
        min-length: 10;
    }

    ORF { type: orf; }

    Repeat.left--ORF.unit {
        type: distance;
        distance_type:
        end-to-start;
        min: 0;
        max: 5000;
    }
    ORF.unit--Repeat.right {
        type: distance;
        distance_type: end-to-start;
        min: 0;
        max: 5000;
    }
}
```

Query Elements

Algorithm Elements

CDD Algorithm Element

When the *element* is used the input nucleotide sequence is translated into 6 amino sequences. The translated sequences are used to query the NCBI Conserved Domain Database (CDD).

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	CDD result
Expected value	It is also called E-value. Describes the number of hits one can “expect” to see by chance when searching a database of a particular size.	10
Min length	Minimum result length.	50bp
Max length	Maximum result length.	5000bp
Pattern	Filters the result annotations: the qualifier value must contain the specified pattern value.	No default value.

Parameters in Schema File

Type: CDD

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*

evalue	**Expected value**	*numeric*
min-length	**Min length**	*numeric*
max-length	**Max length**	*numeric*
pattern	**Pattern**	*string*

Base Content Algorithm Element

Searches regions in a sequence that contain a specified percentage of a certain base.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	misc_feature
Base	Specifies the base, i.e. A, C, G or T.	You must specify a value!
Percentage	Percentage of the base in a region. The value must be >= 50%.	90%
Min Length	Minimum length of a region. The value must be >= 5.	5 bp
Max Length	Maximum length of a region.	5 bp
Direction	See the description *here*.	Any

Parameters in Schema File

Type: base-content

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
base	**Base**	*string*
percent	**Percentage**	*numeric*
min-len	**Min Length**	*numeric*
max-len	**Max Length**	*numeric*

Gc Content Algorithm Element

Searches regions in a sequence with GC content in the specified range.

Parameters in GUI

Parameter	Description	Default value
Min GC content	Min GC content value in percents.	20%
Max GC content	Max GC content value in percents.	40%
Min length	Minimum length of a region.	50bp
Max length	Maximum length of a region.	1000bp

Parameters in Schema File

Type: find-gc

Parameter	Parameter in the GUI	Type
region-start	**Min GC content**	*numeric*

region-end	**Max GC content**	*numeric*
min-len	**Min length**	*numeric*
max-len	**Max length**	*numeric*

HMM3 Algorithm Element

Searches a sequence for significantly similar sequence matches with one or more profile HMM and saves the results as annotations.

The search is performed using HMMER3 **hmmsearch** tool integrated into UGENE.

Parameters in GUI

General parameters:

Parameter	Description	Default value
Annotate As	Name of the result annotations.	hmm_signal
Profile HMM	Semicolon-separated list of input HMM files.	You must specify a value!
Min Length	Minimum length of a result region.	30
Max Length	Maximum length of a result region.	5000

Parameters controlling reporting threshold:

Reporting thresholds controls which hits are reported.

Parameter	Description	Default value
Use E-value	Filter by E-value if true. Otherwise, filters by score.	True
Filter by High E-value	Reports domains <= this E-value threshold in output (**hmmsearch–domE** option).	1e+1
Filter by Low Score	Reports domains >= this score cutoff in output (**hmmsearch–domT** option).	0.01

Parameters controlling the acceleration pipeline:

HMMER3 searches are accelerated in a three-step filter pipeline: the MSV filter, the Viterbi filter, and the Forward filter. The first filter is the fastest and most approximate; the last is the full Forward scoring algorithm. There is also a bias filter step between MSV and Viterbi.

Parameter	Description	Default value
Max	Turns off all acceleration heuristic filters. This increases sensitivity somewhat, at a large cost in speed.	False
MSV Filter Threshold	P-value threshold for the MSV filter step.	0.02
Viterbi Filter Threshold	P-value threshold for the Viterbi filter step.	0.001
Forward Filter Threshold	P-value threshold for the Forward filter step.	1e-5
No Bias Filter	Turns off composition bias filter. This increases sensitivity somewhat, but can come at a high cost in speed.	False

Other parameters:

Parameter	Description	Default value
No Null2	Turns off the null2 score corrections for biased composition.	False
Number of Sequences	Specifies number of significant sequences. It is used for domain E-value calculations (**hmmsearch–domZ** option).	1 (i.e. one input sequence)
Seed	Random number seed. The default is to use a fixed seed(42), so that results are exactly reproducible. Any other positive integer will give different (but also reproducible) results. A choice of 0 uses a randomly chosen seed.	42

Parameters in Schema File

Type: hmm3

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
min-len	**Min Length**	*string*
max-len	**Max Length**	*string*
hmm-profile	**Profile HMM**	*string*
use-e-val	**Use E-value**	*boolean*
e-val	**Filter by High E-value**	*numeric*
score	**Filter by Low Score**	*numeric*
do-max	**Max**	*boolean*
msv-filter-threshold	**MSV Filter Threshold**	*numeric*
viterbi-filter-threshold	**Viterbi Filter Threshold**	*numeric*
forward-filter-threshold	**Forward Filter Threshold**	*numeric*
no-bias-filter	**No Bias Filter**	*boolean*
no-score-corrections	**No Null2**	*boolean*
seqs-num	**Number of Sequences**	*numeric*
random-generator-seed	**Seed**	*numeric*

ORF Algorithm Element

The *element* searches for open reading frames (ORFs) in the supplied sequence.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	ORF
Direction	See the description *here*.	Any
Allow alternative codons	Allows / disallows ORFs starting with alternative initiation codons, accordingly to the current translation table.	False
Require init codons	Allows / disallows ORFs starting with any codon other than terminator.	True

Require stop codons	Ignores or takes into account boundary ORFs which last beyond the search region.	False
Min length	Ignores ORFs shorter than the specified length.	100bp
Max length	Maximum length of annotation allowed.	10000bp
Genetic code	Genetic code that should be used to translate the input nucleotide sequence.	The standard genetic code

Parameters in Schema File

Type: orf

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
strand	**Direction**	*string* Available values are: • complement • direct • both
alt-start	**Allow alternative codons**	*boolean*
starts-with-init	**Require init codons**	*boolean*
ends-with-stop	**Require stop codons**	*boolean*
min-length	**Min length**	*numeric*
max-length	**Max length**	*numeric*
trans-id	**Genetic code**	*string* Available values are: • “NCBI-GenBank #1” • “NCBI-GenBank #2” • etc.

Primer Algorithm Element

The *element* searches primers against the input sequence.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	top primers
Direction	See the description *here*.	Any
Excluded regions	The regions should be avoided for primer selection.	No default value.
Max repeat mispriming	The maximum allowed weighted similarity with any sequence in Mispriming Library.	12
Max template mispriming	The maximum allowed similarity to ectopic sites in the sequence from which you are designing the primers.	12
Number to return	The maximum number of primer pairs to return.	5

Pair max repeat mispriming	The maximum allowed sum of similarities of a primer pair (one similarity for each primer) with any single sequence in Mispriming Library.	24
Pair max template mispriming	The maximum allowed summed similarity of both primers to ectopic sites in the sequence from which you are designing the primers.	24
Product size ranges	List of product size ranges. Primer first tries to pick primers in the first range. If that is not possible, it goes to the next range and tries again. It continues in this way until it has either picked all necessary primers or until there are no more ranges.	150-200 100-300 301-400 401-500 501-600 601-700 701-850 851-1000
Max 3' stability	The maximum stability for the last five 3' bases of a left or right primer. Bigger numbers mean more stable 3' ends.	9
Targets	If one or more Targets is specified then a legal primer pair must flank at least one of them.	No default value.

Parameters in Schema File

Type: primer

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
excluded_regions	**Excluded region**	*string*
max_mispriming	**Max repeat mispriming**	*numeric*
max_template_mispriming	**Max template mispriming**	*numeric*
num_return	**Number to return**	*numeric*
pair_max_mispriming	**Pair max repeat mispriming**	*numeric*
pair_max_template_mispriming	**Pair max template mispriming**	*numeric*
size_ranges	**Product size range**	*string*
stability	**Max 3' stability**	*numeric*
targets	**Targets**	*string*

Repeats Algorithm Element

The *element* searches for repeats in the input sequence.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	repeat_unit

Algorithm	Algorithm used to search for repeats. Available values are: • Suffix index • Diagonals • Auto	Suffix index
Filter nested	If *True*, filters nested repeats.	True
Identity	Repeats percentage identity.	100%
Inverted	If *True*, searches for inverted repeats.	False
Min length	Minimum length of repeats.	5bp
Max length	Maximum length of repeats.	10000bp
Parallel threads	Number of parallel threads used to execute the calculations.	Auto

Parameters in Schema File

Type: repeats

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
algorithm	**Algorithm**	*string* Available values are: • suffix • diagonals • auto
filter-nested	**Filter nested**	*boolean*
identity	**Identity**	*numeric*
invert	**Invert**	*boolean*
min-length	**Min length**	*numeric*
max-length	**Max length**	*numeric*
threads	**Parallel threads**	*numeric*

Restriction Sites Algorithm Element

The *element* searches restriction sites in the input sequence.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	Special value <rsite> is used. It specifies to use the enzymes names as names of the result annotations.
Circular	If *True* allows to search for restriction sites between the end and the beginning of the sequence.	False
Enzymes	Restriction enzymes used to recognize the restriction sites.	You must specify a value!

Parameters in Schema File

Type: rsite

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
circular	**Circular**	*boolean*
enzymes	**Enzymes**	*string*

Pattern Algorithm Element

The *element* searches for the specified pattern in the supplied sequence.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	misc_feature
Pattern	The pattern to search for.	You must specify a value!
Direction	See the description *here*.	Any

Parameters in Schema File

Type: search

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
pattern	**Pattern**	*string*

SITECON Algorithm Element

The *element* searches the input sequence for transcription factor binding sites (TFBSs) significantly similar to the specified SITECON profiles.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	misc_feature
Direction	See the description *here*.	Any
Min Err1	Filters the results by minumum value of Error type I.	0
Max Err2	Filters the results by maximum value of Error type II.	0.001
Model	Semicolon-separated list of SITECON profiles.	You must specify a value!
Min score	Recognition quality percentage threshold. Choosing too low threshold will lead you to recognition of too many TFBS. Choosing too high threshold will lead to no TFBS recognized.	85%

Parameters in Schema File

Type: sitecon

Parameter	Parameter in the GUI	Type

key	**Annotate As**	*string*
err1	**Min Err1**	*numeric*
err2	**Max Err2**	*numeric*
profile	**Model**	*string*
score	**Min score**	*numeric*
strand	**Direction**	*string* Available values are: • complement • direct • both

Smith-Waterman Algorithm Element

The *element* uses the Smith-Waterman algorithm to search in the input sequence for regions similar to the specified pattern.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	misc_feature
Direction	See the description *here*.	Any
Algorithm	Algorithm version. Depending on the computer configuration the following values may be available: • Classic 2 • SSE2 • CUDA	Classic 2
Filter results	Results filtering strategy. The values available are: • filter-intersections • none	filter-intersections
Gap ext score	Gap extension score.	-1.00
Gap open score	Gap open score.	-10.00
Scoring matrix	Specifies the scoring matrix to use.	Auto
Min score	Percentage of matching between the pattern and the searched sequence region.	90%
Pattern	The pattern to search for.	You must specify a value!
Search in translation	Translates the nucleotide sequence supplied to a protein sequence and searches in the translated sequence.	False

Parameters in Schema File

Type: ssearch

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*

algorithm	**Algorithm**	*string* Depending on the computer configuration the values available are: • “Classic 2” • SSE2 • CUDA
filter	**Filter results**	*string* The values available are: • filter-intersections • none
gap-ext-score	**Gap ext score**	*numeric*
gap-open-score	**Gap open score**	*numeric*
matrix	**Scoring matrix**	*string*
min-score	**Min score**	*numeric*
pattern	**Pattern**	*string*
strand	**Direction**	*string* Available values are: • complement • direct • both
translate	**Search in translation**	*boolean*

Tandem Repeats Algorithm Element

The *Tandem repeats element* finds tandem repeats in a supplied sequence, stores found regions as annotations.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	repeate_unit
Direction	See the description *here*.	Any
Algorithm	The algorithm parameter allows one to select the search algorithm. The default and a fast one is the optimized suffix array algorithm.	Suffix index(optimized)
Min period	Minimum acceptable repeat length measured in base symbols.	1 n
Max period	Maximum acceptable repeat length measured in base symbols.	1000000 n
Min tandem size	The minimum tandem size sets the limit on minimum acceptable length of the tandem, i.e. the minimum total repeats length of the searched tandem.	9
Min repeat count	The minimum number of repeats of a searched tandem.	x3
Search for overlapped tandems	If this parameter is set to *True* then overlapped tandems should be included into the result.	

Parallel threads	Number of parallel threads used for the task.	Auto

Parameters in Schema File

Type: tandems

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
algorithm	**Algorithm**	*string* Available values are: • Suffix index • Suffix index (optimized)
min-period	**Min period**	*numeric*
max-period	**Max period**	*numeric*
min-tandem-size	**Min tandem size**	*numeric*
min-repeat-count	**Min repeat count**	*numeric*
show-overlapped-tandems	**Search for overlapped tandems**	*boolean*
strand	**Direction**	*string* Available values are: • complement • direct • both
n-threads	**Parallel threads**	*string*

Weight Matrix Algorithm Element

The *element* searches the input sequence for transcription factor binding sites (TFBSs) significantly similar to the specified weight matrix.

Parameters in GUI

Parameter	Description	Default value
Annotate As	Name of the result annotations.	misc_feature
Direction	See the description *here*.	Any
Matrix	Path to the profile.	You must specify a value!
Min score	Minimum score to detect TFBS.	85%

Parameters in Schema File

Type: wsearch

Parameter	Parameter in the GUI	Type
key	**Annotate As**	*string*
matrix	**Matrix**	*string*
min-score	**Min score**	*numeric*

strand	**Direction**	*string* Available values are: • complement • direct • both

HMM2 Algorithm Element

Searches HMM signals in a sequence with one or more profile HMM2 and saves the results as annotations.

Parameters in GUI

Parameter	Description	Default value
Profile HMM	Semicolon-separated list of input HMM files.	You must specify a value!
Min Length	Minimum length of a result region.	10
Max Length	Maximum length of a result region.	1000
Filter by High E-value	Reports domains <= this E-value threshold in output (**hmmsearch–domE** option).	1e+1
Filter by Low Score	Reports domains >= this score cutoff in output (**hmmsearch–domT** option).	0.01
Number of Sequences	Specifies number of significant sequences. It is used for domain E-value calculations (**hmmsearch–domZ** option).	1 (i.e. one input sequence)

Parameters in Schema File

Type: hmm2

Parameter	Parameter in the GUI	Type
min-len	**Min Length**	*string*
max-len	**Max Length**	*string*
hmm-profile	**Profile HMM**	*string*
e-val	**Filter by High E-value**	*numeric*
score	**Filter by Low Score**	*numeric*
seqs-num	**Number of Sequences**	*numeric*

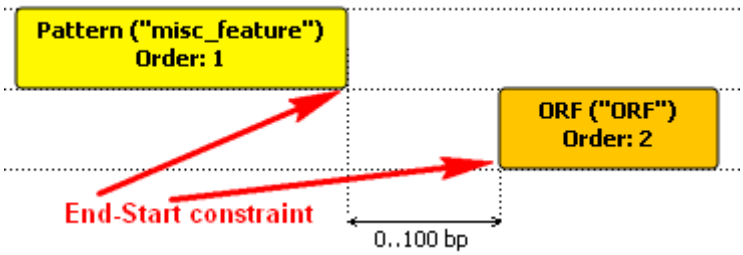
Constraint Elements

To add a constraint element add the required algorithm elements to the scene and drug&drop the required constraint element.

When a *constraint element is added*, two *algorithm elements* are selected.

There are four types of constraints that you can impose on the positional relationship of the results obtained from the algorithms calculations: *End-Start*, *Start-End*, *End-End* and *Start-Start*.

On the image below you can see a *schema* with *Pattern* and *ORF* algorithm elements and an *End-Start* constraint:



This means the following:

1. The algorithm elements specify to analyze the sequence with Pattern and ORF algorithms. The results of these analyses are the sets of annotations.

2. The condition says that the distance between a “Pattern annotation end” and an “ORF annotation start” should be within the specified bounds, i.e.:

Let:

pattern_annot_end := the last nucleotide of an annotation obtained from the Pattern algorithm calculations

orf_annot_start := the first nucleotide of an annotation obtained from the ORF algorithm calculations

The constraint is:

0bp <= Distance(**pattern_annot_end**, **orf_annot_start**) <= 100bp

Find the details on each constraint element below.

- End-Start Constraint Element
- Start-End Constraint Element
- End-End Constraint Element
- Start-Start Constraint Element

End-Start Constraint Element

Add the End-Start constraint to some two *algorithm elements*. Lets denote these elements as **alg1** and **alg2**.

Parameters in GUI

Parameter	Description	Default value
Min distance	Minimum distance between an **alg1** annotation end and an **alg2** annotation start.	0bp
Max distance	Maximum distance between an **alg1** annotation end and an **alg2** annotation start.	0bp

Constraint Explanation:

Let:

alg1_annot_end := the last nucleotide of an annotation obtained from the **alg1**.

alg2_annot_start := the first nucleotide of an annotation obtained from the **alg2**.

The result annotations should comply with the rule:

Min distance <= Distance(**alg1_annot_end**, **alg2_annot_start**) <= **Max distance**

Parameters in Schema File

Type: distance

Distance-type: end-to-start

Parameter	Parameter in the GUI	Type
min	**Min distance**	*numeric*
max	**Max distance**	*numeric*

Start-End Constraint Element

Add the Start-End constraint to some two *algorithm elements*. Lets denote these elements as **alg1** and **alg2**.

Parameters in GUI

Parameter	Description	Default value
Min distance	Minimum distance between an **alg1** annotation start and an **alg2** annotation end.	0bp
Max distance	Maximum distance between an **alg1** annotation start and an **alg2** annotation end.	0bp

Constraint Explanation:

Let:

alg1_annot_start := the first nucleotide of an annotation obtained from the **alg1**.

alg2_annot_end := the last nucleotide of an annotation obtained from the **alg2**.

The result annotations should comply with the rule:

Min distance <= Distance(**alg1_annot_start**, **alg2_annot_end**) <= **Max distance**

Parameters in Schema File

Type: distance

Distance-type: start-to-end

Parameter	Parameter in the GUI	Type
min	**Min distance**	*numeric*
max	**Max distance**	*numeric*

End-End Constraint Element

Add the End-End constraint to some two *algorithm elements*. Lets denote these elements as **alg1** and **alg2**.

Parameters in GUI

Parameter	Description	Default value
Min distance	Minimum distance between an **alg1** annotation end and an **alg2** annotation end.	0bp
Max distance	Maximum distance between an **alg1** annotation end and an **alg2** annotation end.	0bp

Constraint Explanation:

Let:

alg1_annot_end := the last nucleotide of an annotation obtained from the **alg1**.

alg2_annot_end := the last nucleotide of an annotation obtained from the **alg2**.

The result annotations should comply with the rule:

Min distance <= Distance(**alg1_annot_end**, **alg2_annot_end**) <= **Max distance**

Parameters in Schema File

Type: distance

Distance-type: end-to-end

Parameter	Parameter in the GUI	Type

min	**Min distance**	*numeric*
max	**Max distance**	*numeric*

Start-Start Constraint Element

Add the Start-Start constraint to some two *algorithm elements*. Lets denote these elements as **alg1** and **alg2**.

Parameters in GUI

Parameter	Description	Default value
Min distance	Minimum distance between an **alg1** annotation start and an **alg2** annotation start.	0bp
Max distance	Maximum distance between an **alg1** annotation start and an **alg2** annotation start.	0bp

Constraint Explanation:

Let:

alg1_annot_start := the first nucleotide of an annotation obtained from the **alg1**.

alg2_annot_start := the first nucleotide of an annotation obtained from the **alg2**.

The result annotations should comply with the rule:
Min distance <= Distance(**alg1_annot_start**, **alg2_annot_start**) <= **Max distance**

Parameters in Schema File

Type: distance

Distance-type: start-to-start

Parameter	Parameter in the GUI	Type
min	**Min distance**	*numeric*
max	**Max distance**	*numeric*