mutalyzer-hgvs-parser

Release 0.3.5

Aug 17, 2023

Contents:

1 Quick start **3**
- 1.1 Installation . . . 3
- 1.2 Usage . . . 4
- 1.3 Command Line Interface . . . 6
- 1.4 Grammar . . . 6
- 1.5 Library . . . 7
- 1.6 API documentation . . . 7
- 1.7 Contributors . . . 11

Python Module Index **13**

Index **15**

Package to syntax check and convert Mutalyzer HGVS variant descriptions into a dictionary model to easily access descriptions information in a programmatically manner.

Features:

- Accepts HGVS descriptions with multiple variants (one HGVS allele).
- Any description sub-part can be parsed and converted as well.
- Supports common deviations to the HGVS guidelines.
- Command line and library interfaces available.

CHAPTER 1

Quick start

Parse and convert a description from the command line:

```
$ mutalyzer_hgvs_parser -c "NG_012337.1:c.20del"
{
  "reference": {
    "id": "NG_012337.1"
  },
  "coordinate_system": "c",
  "variants": [
    {
      "location": {
        "type": "point",
        "position": 20
      },
      "type": "deletion",
      "source": "reference"
    }
  ]
}
```

The `to_model()` function can be used for the same purpose:

```
>>> from mutalyzer_hgvs_parser import to_model
>>> model = to_model("NG_012337.1:c.20del")
>>> model['reference']
{'id': 'NG_012337.1'}
```

Please see ReadTheDocs for the latest documentation.

1.1 Installation

The software is distributed via PyPI and can be installed with `pip`:

```
pip install mutalyzer-hgvs-parser
```

1.1.1 From source

The source is hosted on GitHub, to install the latest development version, use the following commands.

```
git clone https://github.com/mutalyzer/hgvs-parser.git
cd hgvs-parser
pip install .
```

1.2 Usage

This package provides a *command line interface*.

1.2.1 Syntax check

To only check if a description can be successfully parsed.

```
$ mutalyzer_hgvs_parser 'NG_012337.1(SDHD_v001):c.274G>T'
Successfully parsed:
 NG_012337.1(SDHD_v001):c.274G>T
```

1.2.2 Description model

To obtain the model of a description add the `-c` flag.

```
$ mutalyzer_hgvs_parser -c 'NG_012337.1(SDHD_v001):c.274G>T'
{
  "reference": {
    "id": "NG_012337.1",
    "selector": {
      "id": "SDHD_v001"
    }
  },
  "coordinate_system": "c",
  "variants": [
    {
      "type": "substitution",
      "source": "reference",
      "location": {
        "type": "point",
        "position": 274
      },
      "deleted": [
        {
          "source": "description",
          "sequence": "G"
        }
      ],
      "inserted": [
```

(continues on next page)

(continued from previous page)

```
        {
          "source": "description",
          "sequence": "T"
        }
      ]
    }
  ]
}
```

1.2.3 Grammar start rule

By default, the Mutalyzer `grammar` is used, with `description` as the start (top) rule. It is however possible to choose a different start rule with the `-r` option.

```
$ mutalyzer_hgvs_parser -r variant '274G>T'
Successfully parsed:
 274G>T
```

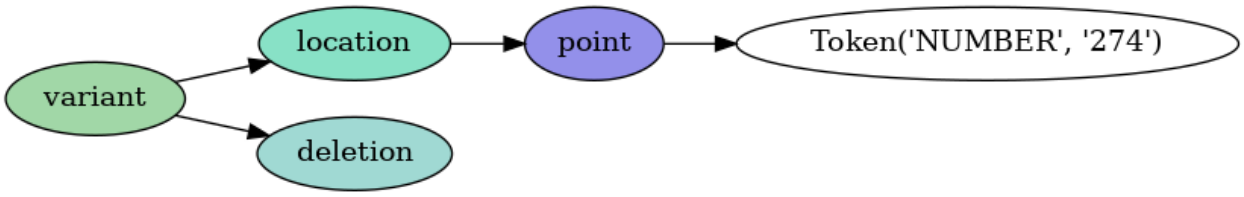
The `-c` flag can be employed together with a different start rule.

```
$ mutalyzer_hgvs_parser -c -r variant '274G>T'
{
  "location": {
    "type": "point",
    "position": 274
  },
  "type": "substitution",
  "source": "reference",
  "deleted": [
    {
      "sequence": "G",
      "source": "description"
    }
  ],
  "inserted": [
    {
      "sequence": "T",
      "source": "description"
    }
  ]
}
```

1.2.4 Parse tree representation

If pydot is installed, an image of the lark parse tree can be obtained with the `-i` option.

```
$ mutalyzer_hgvs_parser "274del" -r variant -i tree.png
Successfully parsed:
 274del
Parse tree image saved to:
 tree.png
```



1.3 Command Line Interface

Mutalyzer HGVS variant description parser.

```
usage: mutalyzer_hgvs_parser [-h] [-c] [-r R] [-g G] [-p] [-i I] [-v]
                             description
```

1.3.1 Positional Arguments

description	the HGVS variant description to be parsed

1.3.2 Named Arguments

-c	convert the description to the model
	Default: False
-r	alternative start (top) rule for the grammar
-g	alternative input grammar file path (do not use with -c)
-p	raw parse tree (no ambiguity solving)
	Default: False
-i	save the parse tree as a PNG image (pydot required!)
-v	show program's version number and exit

Copyright (c) Mihai Lefter <M.Lefter@lumc.nl>

1.4 Grammar

The derived `EBNF grammar` does not consider all the HGVS nomenclature recommendations. Currently, the focus is mostly on descriptions at the DNA level. Examples of descriptions not supported:

- `LRG_199t1:c.[2376G>C];[3103del]`
- `LRG_199t1:c.2376G>C(;)3103del`
- `NC_000002.12:g.pter_8247756delins[NC_000011.10:g.pter_15825272]`
- `NC_000009.12:g.pter_26393001delins102425452_qterinv`
- `NC_000011.10::g.1999904_1999946|gom`

At the same time, the grammar allows for descriptions which are not HGVS compliant, but interpretable, in order to help users reach a normalized description. Examples:

- `LRG_1:g.20>T`
- `LRG_1:g.20_40>70_80`
- `LRG_1:g.20_23delAATG`
- `NG_012337.1(NM_003002.2):274G>T`

1.5 Library

The library provides a number of functions/classes to parse and convert descriptions.

1.5.1 The `to_model()` function

The `to_model()` function can be used to convert an HGVS description to a dictionary model.

```
>>> from mutalyzer_hgvs_parser import to_model
>>> model = to_model('NG_012337.1(SDHD_v001):c.274del')
>>> model['reference']
{'id': 'NG_012337.1', 'selector': {'id': 'SDHD_v001'}}
```

An alternative start rule for the grammar can be used.

```
>>> from mutalyzer_hgvs_parser import to_model
>>> model = to_model('274del', 'variant')
>>> model
{'location': {'type': 'point', 'position': 274}, 'type': 'deletion', 'source':
↪'reference'}
```

1.5.2 The `parse()` function

The `parse()` function can be used to parse for syntax correctness purposes an HGVS description. Its output is a lark parse tree.

```
>>> from mutalyzer_hgvs_parser import parse
>>> parse("LRG_1:100del")
Tree('description', [Tree('reference', [Token('ID', 'LRG_1')]), Tree('variants',
[Tree('variant', [Tree('location', [Tree('point', [Token('NUMBER', '100')])]), Tree(
↪'deletion', [])])])])
```

1.6 API documentation

1.6.1 Hgvs parser

Module for parsing HGVS variant descriptions.

class mutalyzer_hgvs_parser.hgvs_parser.**AmbigTransformer**(*visit_tokens: bool = True*)

Bases: `lark.visitors.Transformer`

class mutalyzer_hgvs_parser.hgvs_parser.**FinalTransformer**(*visit_tokens: bool = True*)

Bases: `lark.visitors.Transformer`

variant(*children*)

variant_predicted(*children*)

variants(*children*)

class mutalyzer_hgvs_parser.hgvs_parser.**HgvsParser**(*grammar_path=None*, *start_rule=None*, *ignore_white_spaces=True*)

Bases: object

HGVS parser object.

Parameters

- **grammar_path** (*str*) – Path to a different EBNF grammar file.
- **start_rule** (*str*) – Alternative start rule for the grammar.
- **ignore_white_spaces** (*bool*) – Ignore or not white spaces in the description.

parse(*description*)

Parse the provided description.

Parameters **description** (*str*) – An HGVS description.

Returns A parse tree.

Return type lark.Tree

status()

Print parser's status information.

class mutalyzer_hgvs_parser.hgvs_parser.**ProteinTransformer**(*visit_tokens: bool = True*)

Bases: lark.visitors.Transformer

P_COORDINATE_SYSTEM(*name*)

extension(*children*)

extension_c(*children*)

extension_n(*children*)

frame_shift(*children*)

p_deletion(*children*)

p_deletion_insertion(*children*)

p_duplication(*children*)

p_equal(*children*)

p_insert(*children*)

p_inserted(*children*)

p_insertion(*children*)

p_length(*children*)

p_location(*children*)

p_point(*children*)

p_range(*children*)

p_repeat(*children*)

p_repeat_mixed(*children*)

p_repeat_number(*children*)

p_substitution(*children*)

p_variant(*children*)

p_variant_certain(*children*)

p_variant_predicted(*children*)

p_variants(*children*)

p_variants_certain(*children*)

p_variants_predicted(*children*)

mutalyzer_hgvs_parser.hgvs_parser.**parse**(*description*, *grammar_path=None*, *start_rule=None*)

Parse the provided HGVS *description*, or the description part, e.g., a location, a variants list, etc., if an appropriate alternative *start_rule* is provided.

Parameters

- **description** (*str*) – Description (or description part) to be parsed.
- **grammar_path** (*str*) – Path towards a different grammar file.
- **start_rule** (*str*) – Alternative start rule for the grammar.

Returns Parse tree.

Return type lark.Tree

1.6.2 Convert

Module for converting HGVS descriptions and lark parse trees to their equivalent dictionary models.

class mutalyzer_hgvs_parser.convert.**Converter**(*visit_tokens: bool = True*)

Bases: lark.visitors.Transformer

AA(*name*)

COORDINATE_SYSTEM(*name*)

ID(*name*)

INVERTED(*name*)

OFFSET(*name*)

OUTSIDE_CDS(*name*)

P_SEQUENCE(*name*)

SEQUENCE(*name*)

UNKNOWN(*name*)

conversion(*children*)

deletion(*children*)

deletion_insertion(*children*)

description(*children*)

description_dna (*children*)

description_protein (*children*)

duplication (*children*)

equal (*children*)

exact_range (*children*)

extension (*children*)

frame_shift (*children*)

insert (*children*)

inserted (*children*)

insertion (*children*)

inversion (*children*)

length (*children*)

location (*children*)

point (*children*)

range (*children*)

reference (*children*)

repeat (*children*)

repeat_mixed (*children*)

repeat_number (*children*)

substitution (*children*)

uncertain_point (*children*)

variant (*children*)

variant_certain (*children*)

variant_predicted (*children*)

variants (*children*)

variants_predicted (*children*)

mutalyzer_hgvs_parser.convert.**parse_tree_to_model** (*parse_tree*)

Convert a parse tree to a nested dictionary model.

Parameters **parse_tree** (*lark.Tree*) – HGVS description.

Returns Description dictionary model.

Return type dict

mutalyzer_hgvs_parser.convert.**to_model** (*description*, *start_rule=None*)

Convert an HGVS description, or parts of it, e.g., a location, a variants list, etc., if an appropriate alternative *start_rule* is provided, to a nested dictionary model.

Parameters

- **description** (*str*) – HGVS description.
- **start_rule** (*str*) – Alternative start rule.

Returns Description dictionary model.

Return type dict

1.7 Contributors

Main developers:

- Jeroen F.J. Laros <J.F.J.Laros@lumc.nl> (Author initial version)
- Martijn Vermaat <martijn@vermaat.name> (Author initial version, maintainer 2011 - 2016)
- Mihai Lefter <m.lefter@lumc.nl> (Author current version, maintainer)

Other contributions by:

- Jonathan K. Vis <J.K.Vis@lumc.nl> (Architecture current version)
- Mark Santcroos <m.a.santcroos@lumc.nl> (Bug fix current version)
- Mark Kroon <m.kroon@lumc.nl> (Add feature initial version)
- Gerben Stouten <gstouten@gmail.com>

Find out who contributed:

```
git shortlog -s -e
```

Python Module Index

m

mutalyzer_hgvs_parser.convert, 9
mutalyzer_hgvs_parser.hgvs_parser, 7

Index

A

AA() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
AmbigTransformer (*class in mutalyzer_hgvs_parser.hgvs_parser*), 7

C

conversion() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
Converter (*class in mutalyzer_hgvs_parser.convert*), 9
COORDINATE_SYSTEM() (*mutalyzer_hgvs_parser.convert.Converter method*), 9

D

deletion() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
deletion_insertion() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
description() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
description_dna() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
description_protein() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
duplication() (*mutalyzer_hgvs_parser.convert.Converter method*), 10

E

equal() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
exact_range() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
extension() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
extension() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
extension_c() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
extension_n() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8

F

FinalTransformer (*class in mutalyzer_hgvs_parser.hgvs_parser*), 7
frame_shift() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
frame_shift() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8

H

HgvsParser (*class in mutalyzer_hgvs_parser.hgvs_parser*), 8

I

ID() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
insert() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
inserted() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
insertion() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
inversion() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
INVERTED() (*mutalyzer_hgvs_parser.convert.Converter method*), 9

L

length() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
location() (*mutalyzer_hgvs_parser.convert.Converter method*), 10

M

mutalyzer_hgvs_parser.convert (*module*), 9
mutalyzer_hgvs_parser.hgvs_parser (*module*), 7

O

OFFSET() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
OUTSIDE_CDS() (*mutalyzer_hgvs_parser.convert.Converter method*), 9

P

P_COORDINATE_SYSTEM() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_deletion() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_deletion_insertion() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_duplication() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_equal() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_insert() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_inserted() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_insertion() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_length() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_location() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_point() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_range() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_repeat() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_repeat_mixed() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 8
p_repeat_number() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
P_SEQUENCE() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
p_substitution() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variant() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variant_certain() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variant_predicted() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variants() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variants_certain() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
p_variants_predicted() (*mutalyzer_hgvs_parser.hgvs_parser.ProteinTransformer method*), 9
parse() (*in module mutalyzer_hgvs_parser.hgvs_parser*), 9
parse() (*mutalyzer_hgvs_parser.hgvs_parser.HgvsParser method*), 8
parse_tree_to_model() (*in module mutalyzer_hgvs_parser.convert*), 10
point() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
ProteinTransformer (*class in mutalyzer_hgvs_parser.hgvs_parser*), 8

R

range() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
reference() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
repeat() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
repeat_mixed() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
repeat_number() (*mutalyzer_hgvs_parser.convert.Converter method*), 10

S

SEQUENCE() (*mutalyzer_hgvs_parser.convert.Converter method*), 9
status() (*mutalyzer_hgvs_parser.hgvs_parser.HgvsParser method*), 8
substitution() (*mutalyzer_hgvs_parser.convert.Converter method*), 10

T

to_model() (*in module mutalyzer_hgvs_parser.convert*), 10

U

uncertain_point() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
UNKNOWN() (*mutalyzer_hgvs_parser.convert.Converter method*), 9

V

variant() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
variant() (*mutalyzer_hgvs_parser.hgvs_parser.FinalTransformer method*), 7
variant_certain() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
variant_predicted() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
variant_predicted() (*mutalyzer_hgvs_parser.hgvs_parser.FinalTransformer method*), 8
variants() (*mutalyzer_hgvs_parser.convert.Converter method*), 10
variants() (*mutalyzer_hgvs_parser.hgvs_parser.FinalTransformer method*), 8
variants_predicted() (*mutalyzer_hgvs_parser.convert.Converter method*), 10