streamfig Documentation

Release 1.0

Tiphaine Viard

Sep 11, 2017

Contents:

CHAPTER 1

Drawing module

class Drawing.**Drawing**(*alpha=0.0*, *omega=10.0*, *time_width=500*, *discrete=0*)

Bases: `object`

Initializes a stream graph drawing.

Parameters

- **alpha** (*float*) – Start time of stream graph.
- **omega** (*float*) – End time of stream graph.
- **time_width** (*positive int*) – Width (in the final fig) of one time unit.
- **discrete** (*positive int*) – Duration of the time step if time is discrete. 0 if time is continuous.

Example

```
>>> d = Drawing(alpha=0, omega=5.5)
>>> d = Drawing(alpha=0, omega=6, discrete=2)
```

addColor(*name*, *hex*)

Adds a new RGB color for use.

Parameters

- **name** (*str*) – Color identifier (must be unique, case sensitive)
- **hex** (*str*) – Color in hexadecimal format

Example

```
>>> d.addColor("red", "#FF0000")
```

addLink(*u*, *v*, *b*, *e*, *curving=0.0*, *color=0*, *height=0.5*, *width=3*)

Adds a link from time b to time e between nodes u and v.

Parameters

- **u** (*str*) – Node to be linked
- **v** (*str*) – Node to be linked
- **b** (*float*) – Start time of the link
- **e** (*float*) – End time of the link
- **curving** (*float*) – Curving of the link. 0 corresponds to a straight link, negative values will draw the link bent on the left, positive values will draw the link bent on the right
- **color** (*str/int*) – the link's color (see addColor())
- **height** (*float*) – Fixes the position of the duration bar; values are between 0 and 1. 0 would draw the duration bar at node u's level, 1 at node's v, 0.5 in between, etc.
- **width** (*int*) – The link's width

Example

```
>>> # Add link from time 1 to time 3 between nodes u and v
>>> d.addLink("u", "v", 1, 3)
>>> # Add a right curved link from time 1 to time 3 between nodes u and v
>>> d.addLink("u", "v", 1, 3, curving=0.3)
```

addNode (*u*, *times=[]*, *color=0*, *linetype=None*)

Adds a new node to the stream graph.

Parameters

- **u** (*str*) – Name of the node (should be unique).
- **times** (*list of 2-tuples*) – List of tuples indicating when the node is present.
- **color** (*int or str*) – Color of the node, either a XFIG int or a user-defined color.
- **linetype** – ?

Example

```
>>> # Adds a node "v" from alpha to omega
>>> d.addNode("v")
>>> # Adds a node "v" from time 1 to time 2.5 and from time 4 to time 8.
>>> d.addNode("v", times=[(1,2.5),(4,8)])
```

addNodeCluster (*u*, *times=[]*, *color=0*, *width=200*)

Adds a node cluster (drawn as a rectangle) for one node over time.

Parameters

- **u** (*str*) – The node in the cluster
- **times** (*list of tuples*) – The times at which u is in the cluster
- **color** (*str/int*) – The color of the rectangle
- **width** (*int*) – The width of the rectangle

Example

```
>>> # Create the blue node cluster {u}x[3,4] U {v}x[5,7.5] U {x}x[2,4]
>>> d.addNodeCluster("u", [(3,4)], color=11)
>>> d.addNodeCluster("v", [(5,7.5)], color=11)
>>> d.addNodeCluster("x", [(2,4)], color=11)
```

addNodeIntervalMark (*u*, *v*, *color=0*, *width=1*)

addParameter (*letter*, *value*, *color=0*, *width=1*)
Adds a parameter (like Delta=2). Multiple parameters will be placed at the top of the drawing, on each other's side

Parameters

- **letter** (*str*) – The letter for the parameter, in ascii (will be translated in greek, i.e. d gives delta, m gives mu, etc.)
- **value** (*float*) – The value for the parameter
- **color** (*int/str*) – The color (see addColor())
- **width** (*int*) – The interval's width

Example

```
>>> # Adds a parameter delta with value 3
>>> d.addParameter("d", 3)
```

addPath (*path*, *start*, *end*, *gamma=0*, *color=0*, *width=1*, *depth=51*)
Adds a temporal path from a sequence of (t,u,v) meaning that there was a hop from u to v at time t.

Parameters

- **path** (*list*) – A list of (t,u,v) that are the hops in the path
- **start** (*float*) – The start time of the path
- **end** (*float*) – The end time of the path
- **gamma** (*float*) – Useful for gamma-path (if gamma > 0, the hops from u to v will take gamma time units)
- **color** (*int/str*) – The path's color (see addColor())
- **width** (*int*) – The path's width
- **depth** (*int*) – Layer for XFIG. Higher values will put the mark in the background, lower in the foreground.

Example

```
>>> # Path from u to x from time 1 to time 9
>>> d.addPath([(2,u,v), (5, v, x)], 1, 7)
>>> # gamma=2-path from u to x from time 1 to time 9
>>> d.addPath([(2,u,v), (5, v, x)], 1, 9, gamma=2)
```

addRectangle (*u*, *v*, *b*, *e*, *width=100*, *depth=51*, *color=0*, *border="*, *bordercolor=0*, *borderwidth=2*)
Adds a rectangle from node u to node v and from time b to time e. The corners of the rectangle will be (u,b), (u,e), (v,b), (v,e)

Parameters

- **u** (*str*) – Start node
- **v** (*str*) – End node
- **b** (*float*) – Start time
- **e** (*float*) – End time
- **width** (*int*) – The rectangle's width (to add an offset)
- **depth** (*int*) – Layer for XFIG. Higher values will put the mark in the background, lower in the foreground

- **color** (*int/str*) – Background color (see addColor())
- **border** (*str*) – If borders should be drawn, takes "lrtb" (for left, right, top, bottom) as arguments
- **bordercolor** (*int/str*) – The border's color (see addColor())
- **borderwidth** (*int*) – The border's width

Example

```
>>> # Rectangle without border
>>> d.addRectangle("u", "v", 2, 6, color=11)
>>> # Rectangle with border all around
>>> d.addRectangle("u", "v", 2, 6, color=11, border="lrtb")
>>> # Rectangle with borders except on top
>>> d.addRectangle("u", "v", 2, 6, color=11, border="lrb")
```

addTime (*t*, *label=''*, *width=1*, *font=12*, *color=0*)

Adds a vertical dotted line at a given time.

Parameters

- **t** – the time at which the line will be drawn
- **label** – the label that will be displayed on top of the vertical line
- **width** – the line's width
- **font** – the label's font (in pt)
- **color** – the line's color (XFIG defined or user-defined, see addColor())

Example

```
>>> # Adds a vertical line labelled "t" at time 2
>>> d.addTime(2, "t")
```

addTimeIntervalMark (*b*, *e*, *color=0*, *width=1*)

addTimeLine (*ticks=1*, *marks=None*)

Adds a time line at the bottom of the stream graph.

/!Should be called last /! :param ticks: Granularity a which ticks should be outputted (every 2, every 1, etc.) :param marks: Custom ticks in the form (t, l)

Example

```
>>> # Most common usage
>>> d.addTimeLine(ticks=2)
>>> # With one custom tick labeled "a" at time 2.5
>>> d.addTimeLine(ticks=2, marks=[(2.5, "a")])
```

addTimeNodeMark (*t*, *v*, *color=0*, *width=2*, *depth=49*)

Adds a mark (a cross) at a given node and time.

Parameters

- **t** (*float*) – The time at which to add the mark
- **v** (*str*) – The node at which to add the mark
- **color** (*int/str*) – The mark's color (see addColor())
- **width** (*int*) – The mark's width

- **depth** (*int*) – Layer for XFIG. Higher values will put the mark in the background, lower in the foreground.

Example

```
>>> d.addTimeNodeMark(2, "u", color=11, width=3)
```

setLineType (*def_linetype*)
Changes the linetype for nodes (i.e. from dashed to dotted). Default is dotted (linetype=2). See FIG documentation for all values.

Parameters **def_linetype** (*int*) – the new linetype

Drawing.**drange** (*start*, *stop*, *step*)
Helper function generating a range of numbers.

Parameters

- **start** (*float*) – Range start
- **end** (*float*) – Range end
- **step** (*float*) – Range step (the difference between two subsequent elements in the range equals step)

Returns an iterator over the range

Return type generator

Example

```
>>> [i for i in drange(0.0,1.0,0.1)]
[0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index