Adigraph, v1.7.2

Luca Cappelletti

March 2024

Contents

1 Introduction **3**
- 1.1 What is Adigraph? . . . 3
- 1.2 License . . . 3

2 Setup **4**
- 2.1 Installing the dependencies . . . 4
- 2.2 Installing Adigraph . . . 4

3 Usage **5**
- 3.1 Creating a new graph . . . 5
- 3.2 Changing an existing graph . . . 5
- 3.3 Adding nodes . . . 6
  - 3.3.1 Custom node colors . . . 6
  - 3.3.2 Custom node width . . . 6
  - 3.3.3 Custom node labels . . . 6
- 3.4 Automatically position nodes . . . 7
  - 3.4.1 Colored automatically positioned nodes . . . 7
- 3.5 Adding edges . . . 8
  - 3.5.1 A simple edge . . . 8
  - 3.5.2 A looped edge . . . 8
  - 3.5.3 A colored simple edge . . . 8
  - 3.5.4 A wider simple edge . . . 9
  - 3.5.5 A weighted edge . . . 9
  - 3.5.6 A weighted edge with label . . . 9
  - 3.5.7 Edge in both directions . . . 10
  - 3.5.8 Edge with weights in both directions . . . 10
  - 3.5.9 Positioning labels . . . 10
  - 3.5.10 Positioning weights . . . 11
  - 3.5.11 Multiple edges with weights . . . 11
- 3.6 Kleene star operators . . . 11
  - 3.6.1 Kleene star on an element . . . 11
  - 3.6.2 Kleene star minus the element . . . 12
  - 3.6.3 Combining Kleene operations . . . 12
- 3.7 Paths . . . 13

3.7.1 Augmenting paths . . . 13
3.7.2 Custom colored augmenting Paths . . . 16
3.7.3 Custom colored Paths . . . 17
3.8 Cuts . . . 17
3.8.1 Colored cuts . . . 18
3.9 Non oriented (undirected) edges and custom edge stiles . . . 18
3.9.1 Non oriented (undirected) . . . 19
3.9.2 Dashed . . . 20

4 PyAdigraph **21**
4.1 Installation . . . 21
4.2 Example . . . 22
4.2.1 Python code . . . 22
4.2.2 Latex result . . . 23

5 Utilities **25**

6 Warnings **26**
6.1 Reserved words . . . 26

Chapter 1

Introduction

1.1 What is Adigraph?

Adigraph is a latex library for drawing directed graphs and augmenting directed graphs, and to draw cuts over them.

It handles automatically the positioning of labels, with the exception of the horizontal position, and the inclinations of cuts.

The latest version is available on Github.

1.2 License

Copyright 2018 Luca Cappelletti

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sub-license, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Chapter 2

Setup

2.1 Installing the dependencies

Clearly you need to have texlive installed. Then, make sure you have the following packages:

fp Used for floating point calculations.

xparse Used for elaborating parameters.

xstring Used for elaborating strings.

etoolbox Used for operations on lists.

tikz Used for drawing the actual graphs.

tikz calc library Used for some internal calculations in tikz.

To be sure you can run the following, that will install the packages only if they are not already present:

```
sudo tlmgr install etoolbox fp xstring
```

2.2 Installing Adigraph

You can install Adigraph, if it isn't already present in your setup, by running the following on Unix systems:

```
sudo tlmgr install adigraph
```

On windows you should check on your package manager of choice (some latex distribution have a tlmgr implementation on windows too.)

Chapter 3

Usage

3.1 Creating a new graph

Here we create a new Adigraph object, called *myAdigraph*.

```
\NewAdigraph{myAdigraph}{
    <nodes here, separated by semicolon>
}{
    <edges here, separated by semicolon>
}{
    <cuts here, separated by semicolon>
}[
    <edge style here>
]
```

3.2 Changing an existing graph

You can renovate an older graph by calling \RenewAdigraph

```
\RenewAdigraph{myAdigraph}{
    <nodes here, separated by semicolon>
}{
    <edges here, separated by semicolon>
}{
    <cuts here, separated by semicolon>
}[
    <edge style here>
]
```

3.3 Adding nodes

We set its nodes with the following syntax: *¡node name[, textual color, border width]: x coordinate[, y coordinate][: label]¿*.

```
\NewAdigraph{myAdigraph}{
      s:0,0;
      t:4,0;
}
\myAdigraph{}
```

3.3.1 Custom node colors

To color a node you can use the following syntax: *¡node name[, textual color]: x coordinate[, y coordinate]¿*. For example, to draw s in red and t in blue we would write:

```
\NewAdigraph{myAdigraph}{
      s,red:0,0;
      t,blue:4,0;
}
\myAdigraph{}
```

Tested available colors are: red, blue, black, green. You may extend the possible colors with LaTex libraries such as xcolor.

3.3.2 Custom node width

To color a node you can use the following syntax: *¡node name[, textual color[, border width]]: x coordinate[, y coordinate]¿*. For example:

```
\NewAdigraph{myAdigraph}{
      s,red,5:0,0;
      t,blue,3:4,0;
}
\myAdigraph{}
```

3.3.3 Custom node labels

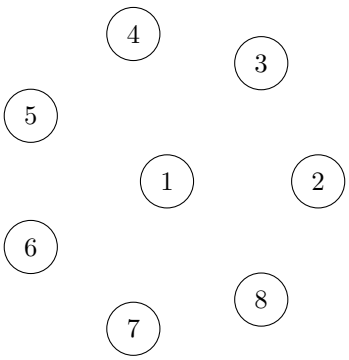
To add a custom label you can use the following syntax: either *¡node name: x coordinate[, y coordinate][: node label]¿* or *¡node name[, textual color]: x coordinate[, y coordinate][: node label]¿* will work:

```
\NewAdigraph{myAdigraph}{
     s,red:0,0:start;
     t:4,0:end;
}
\myAdigraph{}
```

3.4 Automatically position nodes

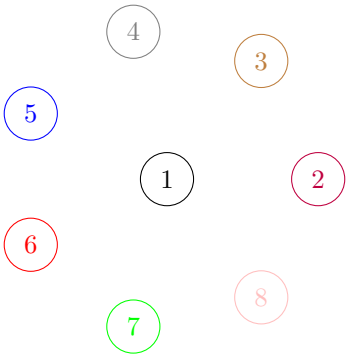
When no coordinates are given or you just don't have time to think about where to put those nodes, just choose a radius and Adigraph will position them on the circle of that radius.

```
\NewAdigraph{myAdigraph}{
    1:0,0;
    2:2;
    3:2;
    4:2;
    5:2;
    6:2;
    7:2;
    8:2;
}
\myAdigraph{}
```



3.4.1 Colored automatically positioned nodes

```
\NewAdigraph{myAdigraph}{
    1:0,0;
    2,purple:2;
    3,brown:2;
    4,gray:2;
    5,blue:2;
    6,red:2;
    7,green:2;
    8,pink:2;
}
\myAdigraph{}
```

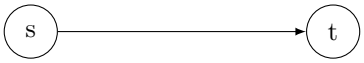


3.5 Adding edges

We set its edges with the following syntax: *¡first node, second node,[color,[edge width]][:weight[:label:[label position]]]¿.*

3.5.1 A simple edge

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t;
}
\myAdigraph{}
```



3.5.2 A looped edge

Looped edges position automatically by themselves to minimize overlapping.

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,s;
    t,t;
    s,t;
}
\myAdigraph{}
```

3.5.3 A colored simple edge

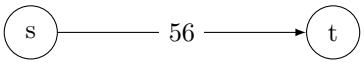
```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t,red;
}
\myAdigraph{}
```

3.5.4 A wider simple edge

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t,red,5;
}
\myAdigraph{}
```

3.5.5 A weighted edge

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t:56;
}
\myAdigraph{}
```

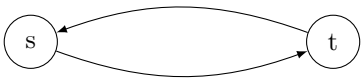


3.5.6 A weighted edge with label

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t:56:myLabel;
}
\myAdigraph{}
```

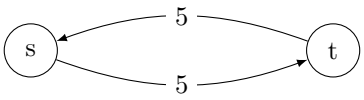
3.5.7 Edge in both directions

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t;
    t,s;
}
\myAdigraph{}
```



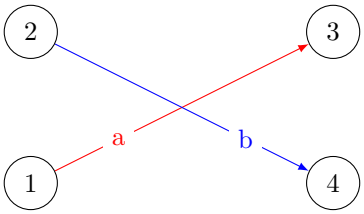
3.5.8 Edge with weights in both directions

```
\NewAdigraph{myAdigraph}{
     s:0,0;
     t:4,0;
}{
    s,t:5;
    t,s:5;
}
\myAdigraph{}
```



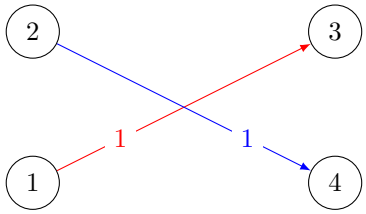
3.5.9 Positioning labels

```
\NewAdigraph{myAdigraph}{
    1:0,0;
    2:0,2;
    3:4,2;
    4:4,0;
}{
    1,3,red:1:a:near start;
    2,4,blue:1:b:near end;
}
\myAdigraph{}
```



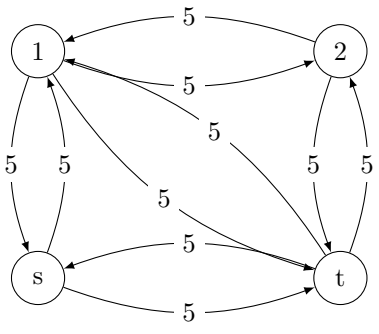
3.5.10 Positioning weights

```
\NewAdigraph{myAdigraph}{
    1:0,0;
    2:0,2;
    3:4,2;
    4:4,0;
}{
    1,3,red:1::near start;
    2,4,blue:1::near end;
}
\myAdigraph{}
```



3.5.11 Multiple edges with weights

```
\NewAdigraph{myAdigraph}{
    s:0,0;
    t:4,0;
    1:0,3;
    2:4,3;
}{
    s,t:5;
    t,s:5;
    s,1:5;
    1,s:5;
    1,2:5;
    2,1:5;
    2,t:5;
    t,2:5;
    t,1:5;
    1,t:5;
}
\myAdigraph{}
```

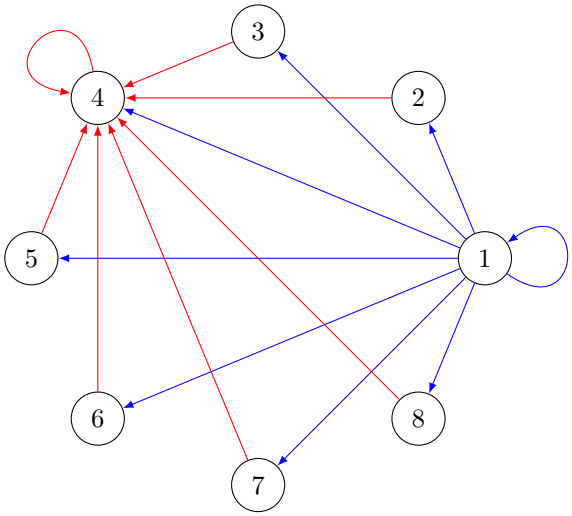


3.6 Kleene star operators

3.6.1 Kleene star on an element

This works only when you don't have a node called ¡*¿. When this happens, the behaviour of a tuple ¡a,*¿ becomes the normal one.

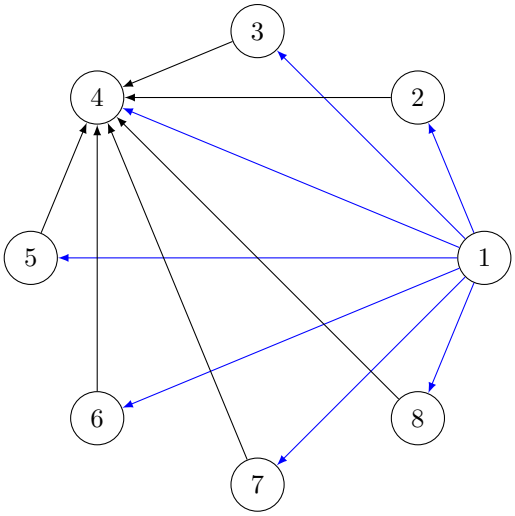
```
\NewAdigraph{myAdigraph}{
    1:3;
    2:3;
    3:3;
    4:3;
    5:3;
    6:3;
    7:3;
    8:3;
}{
    1,*,blue;
    *,4,red;
}
\myAdigraph{}
```



3.6.2 Kleene star minus the element

This works only when you don't have a node called ¡+¿. When this happens, the behaviour of a tuple ¡a,+¿ becomes the normal one.

```
\NewAdigraph{myAdigraph}{
    1:3;
    2:3;
    3:3;
    4:3;
    5:3;
    6:3;
    7:3;
    8:3;
}{
    1,+,blue;
    +,4;
}
\myAdigraph{}
```



3.6.3 Combining Kleene operations

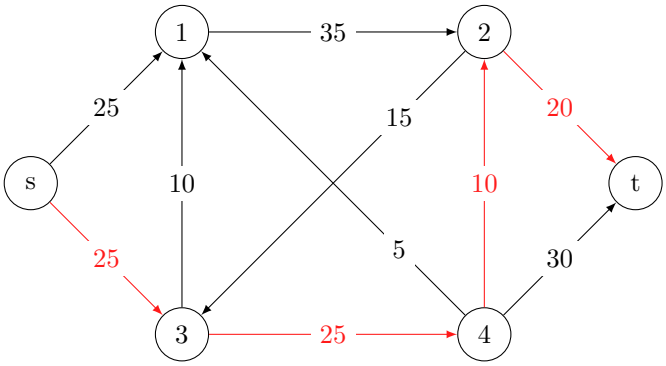
Sadly, operations such as ¡*,+¿ or ¡+,+¿ are not currently supported and not for lack of trying. I'll try implementing them again in the future when I'll have

more time.

3.7 Paths

A path is specified by the following syntax: *¡comma separated list of nodes¿*.

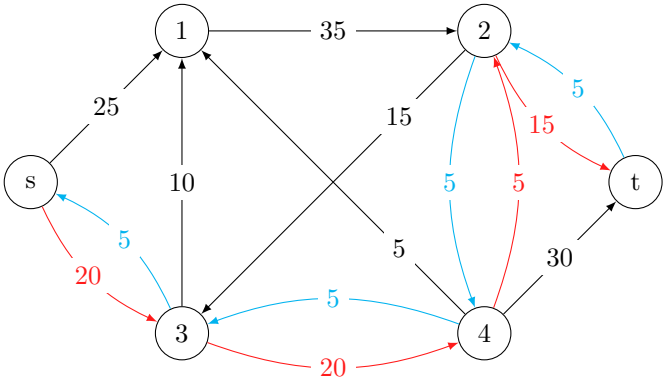
```
\NewAdigraph{myAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}
\myAdigraph{
    s,3,4,2,t;
}
```



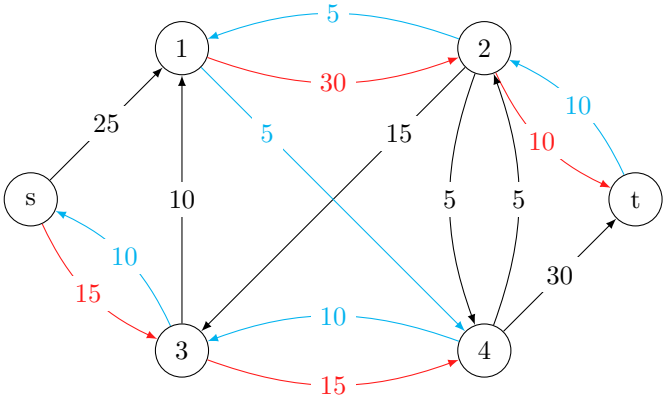
3.7.1 Augmenting paths

An augmenting path is specified by the following syntax: *¡comma separated list of nodes:units¿*. It is **very important** to note that incremental paths called upon the same object are memorized by default.

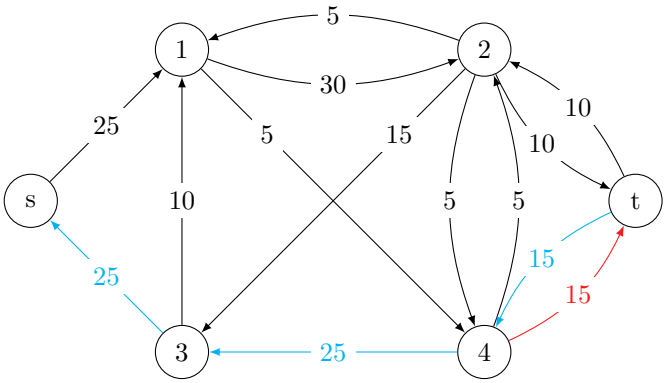
```
\NewAdigraph{myAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}
\myAdigraph{
    s,3,4,2,t:5;
}
```



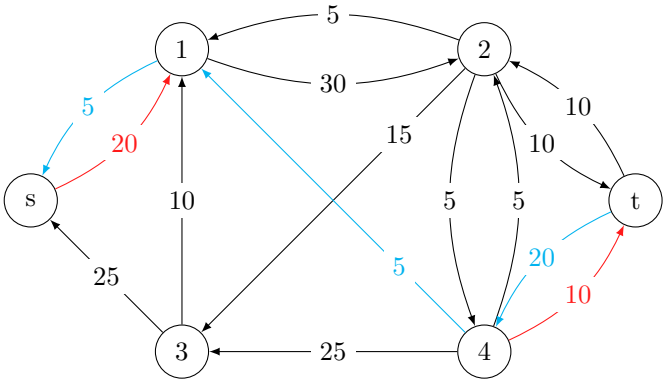
For example, suppose now we'd like to send another 5 units on the graph edited by the previous incremental path, we'll have just to write the following:



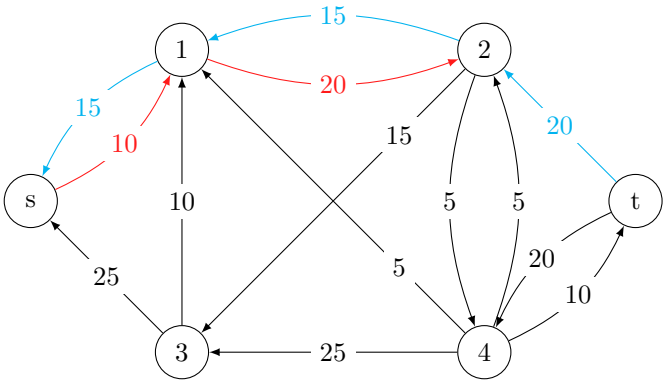

```
\myAdigraph{
    s,3,4,1,2,t:5;
}
```




```
\myAdigraph{
    s,3,4,t:15;
}
```

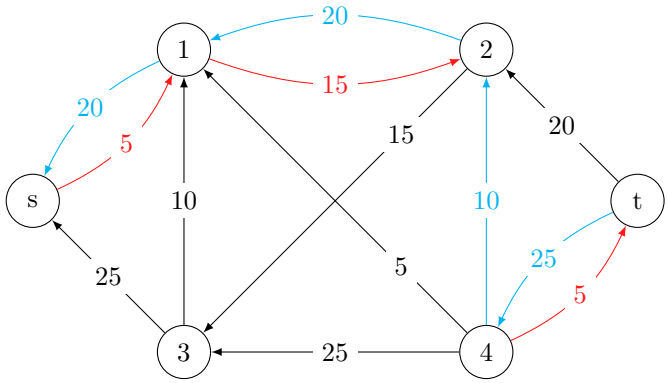



```
\myAdigraph{
    s,1,4,t:5;
}
```




```
\myAdigraph{
    s,1,2,t:10;
}
```

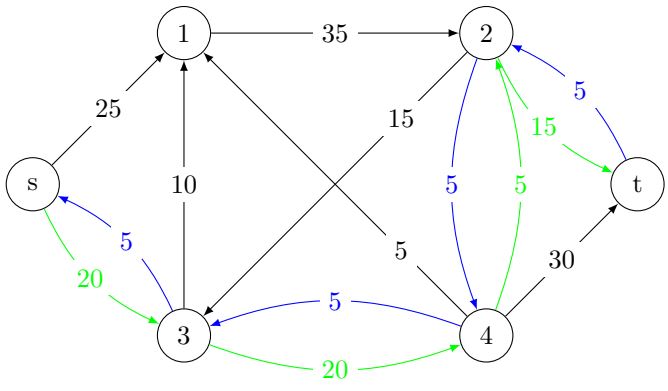
```
\myAdigraph{
    s,1,2,4,t:5;
}
```



3.7.2 Custom colored augmenting Paths

A path is specified by the following syntax: *¡comma separated list of nodes¿:¡units¿:¡forward path color, backward path color¿.*

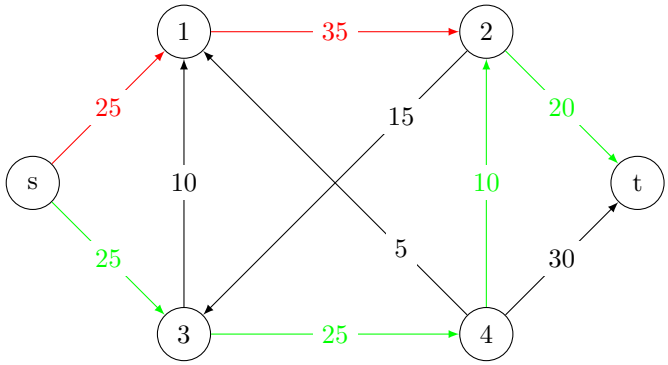
```
\NewAdigraph{myAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}
\myAdigraph{
    s,3,4,2,t:5:green,blue;
}
```



3.7.3 Custom colored Paths

A path is specified by the following syntax: *¡comma separated list of nodes¿::¡forward path color, backward path color¿*. **Note the double colons!**.

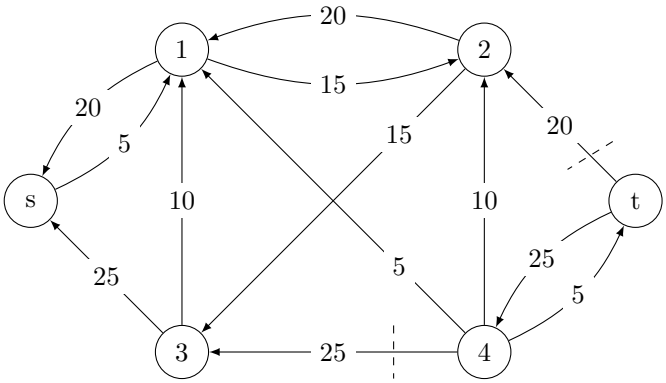
```
\NewAdigraph{myAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}
\myAdigraph{
    s,3,4,2,t::green;
    s,1,2::red;
}
```



3.8 Cuts

The following is to add cuts to show minimum cuts for example, the syntax is: *¡first node, second node;¿*

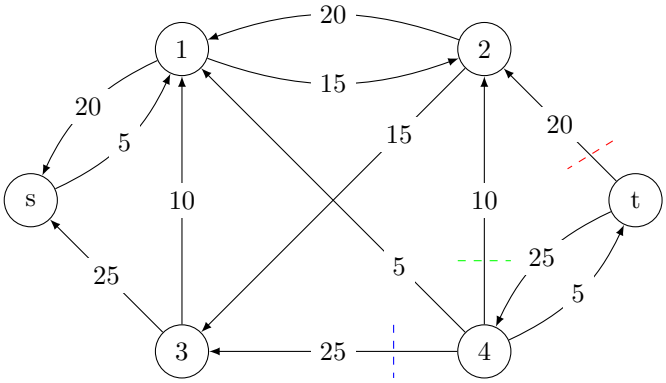
```
\myAdigraph{}{
    3,4;
    2,t;
}
```



3.8.1 Colored cuts

If you'd like to color the cuts you just have to add the color as follows: *¡first node, second node, color;¿*. **Note that if you want to only add a cut and not an augmenting path and a cut, you still need to add the empty curly braces {}.**

```
\myAdigraph{}{
    3,4,red;
    2,t,blue;
    2,4,green;
}
```

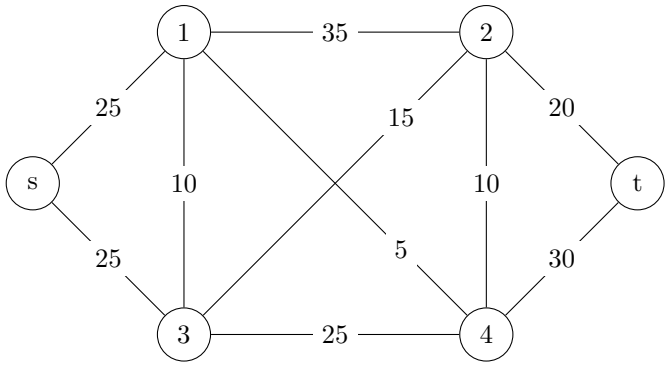


3.9 Non oriented (undirected) edges and custom edge stiles

If you need non oriented edges or in general to ad a custom style to your edges you can proceed as follows:

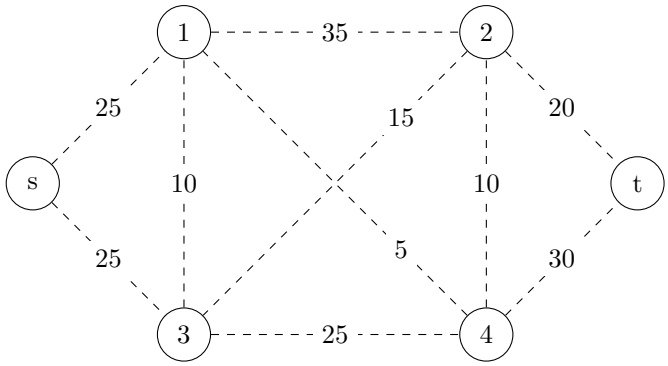
3.9.1 Non oriented (undirected)

```
\NewAdigraph{myCustomEdgesAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}[-]
\myCustomEdgesAdigraph{}
```



3.9.2 Dashed

```
\NewAdigraph{myCustomEdgesAdigraph}{
    s:0,0;
    1:2,2;
    3:2,-2;
    2:6,2;
    4:6,-2;
    t:8,0;
}{
    s,1:25;
    s,3:25;
    3,4:25;
    1,2:35;
    2,t:20;
    4,t:30;
    3,1:10;
    4,2:10;
    2,3:15::near start;
    4,1:5::near start;
}[dashed]
\myCustomEdgesAdigraph{}
```



Chapter 4

PyAdigraph

Pyadigraph turns your networkx into Adigraph latex package. It requires Adigraph (1.7.0+) to work.

4.1 Installation

The package can be installed by simply running:

```
pip installed pyadigraph
```

4.2 Example

4.2.1 Python code

For example by running the following python code:

```python
from pyadigraph import Adigraph
import networkx as nx

A = Adigraph(
    vertices_color_fallback="gray!90",
    edges_color_fallback="gray!90",
    sub_caption="My adigraph number {i} of {n}",
    sub_label="adigraph_{i}_{n}",
    row_size=1,
    caption="A graph generated with python and latex.",
    label="pyadigraph_example"
)

A.add_graph(
    nx.bipartite.random_graph(4, 4, 1),
    vertices_color={
        0: 'red!90',
        1: 'red!90',
        4: 'cyan!90',
        7: 'cyan!90'
    })

A.save("test/result.tex", document=True)
```

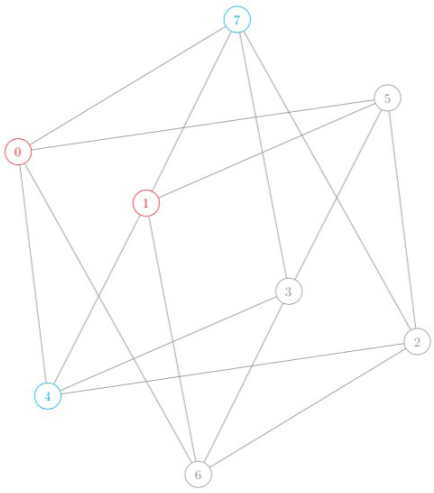
4.2.2 Latex result

You automatically obtain the following latex:

```
\documentclass{report}
\usepackage{adigraph}
\usepackage{subcaption}

\begin{document}
\begin{figure}
    \begin{subfigure}{1.0\textwidth}
        \NewAdigraph{myAdigraph}{
            0,red!90,:-0.4386601404141742\textwidth,0.2091077552922947\textwidth:;
            1,red!90,:-0.15708496776680972\textwidth,0.09630690244229406\textwidth:;
            2,gray!90,:0.43887677279554366\textwidth,-0.2079924280020609\textwidth:;
            3,gray!90,:0.15678823839504888\textwidth,-0.09746320565948384\textwidth:;
            4,cyan!90,:-0.3736460590634439\textwidth,-0.327631363498189\textwidth:;
            5,gray!90,:0.3735687548614322\textwidth,0.3275275669374224\textwidth:;
            6,gray!90,:-0.042735184609099336\textwidth,-0.4998552275122768\textwidth:;
            7,cyan!90,:0.0428925858015027\textwidth,0.5\textwidth:;
        }{
            0,4,gray!90,::;
            0,5,gray!90,::;
            0,6,gray!90,::;
            0,7,gray!90,::;
            1,4,gray!90,::;
            1,5,gray!90,::;
            1,6,gray!90,::;
            1,7,gray!90,::;
            2,4,gray!90,::;
            2,5,gray!90,::;
            2,6,gray!90,::;
            2,7,gray!90,::;
            3,4,gray!90,::;
            3,5,gray!90,::;
            3,6,gray!90,::;
            3,7,gray!90,::;
        }[]
        \myAdigraph{}
    \caption{My adigraph number 1 of 1}\label{adigraph_1_1}
    \end{subfigure}
    \caption{A graph generated with python and latex.}\label{pyadigraph_example}
\end{figure}
\end{document}
```

And once you compile that you receive a graph like the following:



(a) My adigraph number 1 of 1

Figure 1: A graph generated with python and latex.

Chapter 5

Utilities

If for some reason you need to disable all the adigraphs in your latex code, for example using an online editor such as Overleaf that allows only a given maximum compile time, you can use the following command:

```
\DisableAdigraphs
```

You can then re-enable adigraphs with:

```
\EnableAdigraphs
```

Chapter 6

Warnings

6.1 Reserved words

I reserve to use for the package the following tokens:

1. \Adigraph
2. \AdigraphBuildEdge
3. \AdigraphBuildEdgeWrapper
4. \AdigraphBuildNode
5. \AdigraphBuildNodeWrapper
6. \AdigraphBuildPath
7. \AdigraphCalculateOrientation
8. \AdigraphCountPaths
9. \AdigraphCutBuilder
10. \AdigraphDrawEdge
11. \AdigraphDrawNode
12. \AdigraphEdgeBuilder
13. \AdigraphEdgeDrawer
14. \AdigraphElaboratePath
15. \AdigraphExecuteCutBuilder
16. \AdigraphGenerateNodeName
17. \AdigraphMemorizeEdge
18. \AdigraphMemorizeNode
19. \AdigraphNodeBuilder
20. \AdigraphNodeCounter
21. \AdigraphNodeCounterSecondWrapper
22. \AdigraphNodeCounterWrapper
23. \AdigraphNodesCounter
24. \AdigraphPathBuilder
25. \AdigraphProcessAugmentingPaths
26. \AdigraphProcessAugmentingPathsList
27. \AdigraphProcessCuts
28. \AdigraphProcessEdges
29. \AdigraphProcessNodes
30. \AdigraphProcessPaths
31. \AdigraphSimpleSum
32. \NewAdigraph
33. \RenewAdigraph
34. \DisableAdigraphs