Partmac – typesetting set partitions

The file `partmac.tex` contains macros for typesetting set partitions, which are used for example in free probability or the theory of easy quantum groups. The macros use the PGF package to draw the partition. Thus, to use `partmac` one needs to install PGF and then in plainTEX, one can write

```
\input pgfcore
\input partmac
```

For LATEXusers, I've prepared the file `parmac.sty`, which includes PGF automatically, so it is sufficient to write

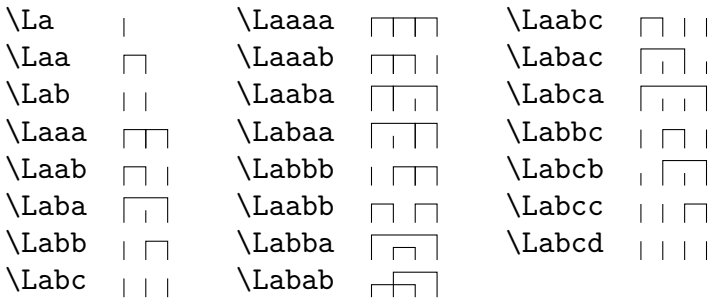
```
\usepackage{partmac}
```

The macros are written in the spirit of plainTEX. Instead of trying to make them universal (which is usually impossible), they are written simply enough that everybody can edit them to achieve their needs.

The main aim of the package is to provide an easy to use tool for drawing set partitions. It provides an universal description of partitions in such a way that the result adjusts its proportions according to the context. So, one can use the same commands for drawing a partition in a title of a chapter, inside paragraph, or as an index of some mathematical symbol.

1 Predefined partitions

There are macros of the form `\Lxxxx` for partitions on lower line, `\Uxxxx` for partitions on upper line and `\Pxxxx` for partitions with the same ammount of upper and lower points. Here `xxxx` is the lexicographically smallest word representation of the partiiton. That is, we have the following macros for partitions with points on the lower line.



Then we have the following macros for partitions with points on the upper line.

\Ua		\Uaaaa		\Uaabc	
\Uaa		\Uaaab		\Uabac	
\Uab		\Uaaba		\Uabca	
\Uaaa		\Uabaa		\Uabbc	
\Uaab		\Uabbb		\Uabcb	
\Uaba		\Uaabb		\Uabcc	
\Uabb		\Uabba		\Uabcd	
\Uabc		\Uabab			

Finally partitions with equal number of points on lower and upper line.

\Paa		\Pabab		\Pabcabc	
\Pab		\Paabc		\Pabcabd	
\Paaaa		\Pabac		\Pabcadc	
\Paaab		\Pabca		\Pabcdbc	
\Paaba		\Pabbc		\Pabcade	
\Pabaa		\Pabcb		\Pabcdbe	
\Pabbb		\Pabcc		\Pabcdec	
\Paabb		\Pabcd		\Pabcdef	
\Pabba				\Paabaab	

In addition, we define the following synonyms.

\singleton		\idpart		\fourpart	
\upsingleton		\disconnecterpart		\crosspart	
\pairpart		\positionerpart		\halflibpart	
\uppairpart		\connecterpart			

2 Partitions on one line

To define a general partition with points only on the lower or upper line, one can use macro `\Lpartition`, resp. `\Upartition`. The syntax is the following.

`\LPartition{`⟨*singletons*⟩`}{`⟨*remaining blocks*⟩`}`

The datum ⟨*singletons*⟩ should be of the form $h : i_1, i_2, \ldots, i_k$, where h is the height of the singleton blocks and $i_1, \ldots, i_k$ are the positions of the singletons. The datum ⟨*remaining blocks*⟩ consists of descriptions of other blocks. Each block is described similarly as the set of singletons, so in the format $h : i_1, \ldots, i_k$, where h is the height of the block and $i_1, \ldots, i_k$ are the positions of elements of the block. The data for the blocks are separated by semicolon.

Let us show this on example.

`\LPartition{0.4:1,4,8}{0.4:2,3;0.4:5,7;0.8:6,9,10}`

Here, the singletons are on position 1, 4, 8 and each of them is represented by a line of height 0.4. Then there are three additional blocks. First connecting points 2 and 3 is represented by a node of height 0.4 (that is, the same as the singletons). Second block

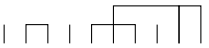
connects points 5 and 7 and has the same height. Finally a block connecting points 6, 9, 10 has double height, that is, 0.8.

The macro `\UPartition` works the same. Except that instead of the height, we should put 1 − height, that is, we put there actually the y-coordinate of the point. That means, to obtain the same result horizontally flipped, we have to write down

```
\UPartition{0.6:1,4,8}{0.6:2,3;0.6:5,7;0.2:6,9,10}
```

The units are chosen in such a way that one should keep the height between 0 and 1 to stick within the line in a paragraph. However, the macro works also if you put there higher numbers, which can be used especially in display mode. For example

```
\LPartition{0.6:1,4,8}{0.6:2,3;0.6:5,7;1.2:6,9,10}
```



3 General partitions

To draw general partitions with upper and lower points, one can use `\Partition{`⟨*data*⟩`}`. The data can consist of the following commands.

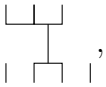
```
\Psingletons y1 to y2:i1,i2,...,ik    %draws singletons
\Pblock      y1 to y2:i1,i2,...,ik    %draws one block
\Pline       (x1,y1) (x2,y2)          %draws a line
```

Here, x_1 and x_2 represent the x coordinates (i.e. position of a point) and y_1 and y_2 the y-coordinates. Again, one is advised to keep the y coordinates between 0 and 1. As an example, we mention the definition of the connecter partition and the positioner partition  .

```
\Partition{                  % connecter partition
\Pblock 0 to 0.3:1,2         % connecting two lower points
\Pblock 1 to 0.7:1,2         % connecting two upper points
\Pline (1.5,0.3) (1.5,0.7)   % connecting the two blocks together
}

\Partition{                 % positioner partition
\Psingletons 0to0.3:2       % singleton on lower line, pos. 2
\Psingletons 1to0.7:1       % singleton on upper line, pos. 1
\Pline (1,0) (2,1)          % line connecting lower pt 1 and upper pt 2
}
```

For drawing more complicated partitions, one can use the `\BigPartition{`⟨*data*⟩`}`, which works exactly the same, but produces a larger result. Another difference is that `\BigPartition` aligns the middle of the partition, i.e. the point $y = 0.5$ with the equals sign. So, for example the result

$p =$  , $q =$ 

can be obtained writing

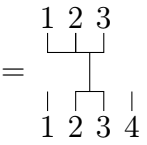
```
$$p=
\BigPartition{
\Pblock 0 to 0.25:2,3
```

```
\Pblock 1 to 0.75:1,2,3
\Psingletons 0 to 0.25:1,4
\Pline (2.5,0.25) (2.5,0.75)
},
\qquad
q=
\BigPartition{
\Psingletons 0 to 0.25:1,4
\Psingletons 1 to 0.75:1,4
\Pline (2,0) (3,1)
\Pline (3,0) (2,1)
\Pline (2.75,0.25) (4,0.25)
}$$
```

4 Adding text

To add text, one can use `\Ptext(`$\langle x\rangle$`,`$\langle y\rangle$`){`$\langle text\rangle$`}`, where $\langle x\rangle$ and $\langle y\rangle$ are coordinates and $\langle text\rangle$ is any TEX code. The $\langle text\rangle$ is wrapped in a box, whose center is described by the coordinates. An example:

```
$$p=
\BigPartition{
\Pblock 0 to 0.25:2,3
\Pblock 1 to 0.75:1,2,3
\Psingletons 0 to 0.25:1,4
\Pline (2.5,0.25) (2.5,0.75)
\Ptext(1,1.2){1}
\Ptext(2,1.2){2}
\Ptext(3,1.2){3}
\Ptext(1,-0.2){1}
\Ptext(2,-0.2){2}
\Ptext(3,-0.2){3}
\Ptext(4,-0.2){4}
}$$
```

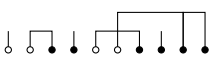
$p =$ 

5 Coloring points

In this section, we describe how to assign different shapes to the set of partitioned points to obtained so-called colored partitions. In the package, we prepared two colors. Command `\Pw` is used to draw white circle ◦ and `\Pb` is used to draw black circle •.

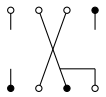
For partitions with lower or upper points only, one can use `\LPartition` resp. `\UPartition` and specify the colorings in the $\langle singletons\rangle$ parameter. An example:

```
\LPartition{0.6:1,4,8;\Pw:1,2,5,6;\Pb:3,4,7,8,9,10}
   {0.6:2,3;0.6:5,7;1.2:6,9,10}
```



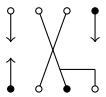
To add points inside `\Partition` or `\Bigpartition`, one can use command of the form `\Ppoint` $\langle y\rangle$ $\langle shape\rangle$`:`$\langle positions\rangle$ as in the following example

```
\BigPartition{
\Psingletons 0 to 0.25:1,4
\Psingletons 1 to 0.75:1,4
\Pline (2,0) (3,1)
\Pline (3,0) (2,1)
\Pline (2.75,0.25) (4,0.25)
\Ppoint0 \Pw:2,4
\Ppoint0 \Pb:1,3
\Ppoint1 \Pw:1,2,3
\Ppoint1 \Pb:4
}
```



Actually, we have an additional two pre-defined points, which are actually arrows. That is, `\Ls` for arrow up (letters stand for lower singleton) and `\Us` for arrow down. Using them, we can draw the colored singleton $\updownarrow$ by `\LPartition{\Ls:1;\Pw:1}{}`. If we wanted to emphasize the singletons in the example above, we can also replace them by arrows.

```
\BigPartition{
\Psingletons 0 to 0.25:4
\Ppoint0 \Ls:1
\Ppoint1 \Us:1,4
\Pline (2,0) (3,1)
\Pline (3,0) (2,1)
\Pline (2.75,0.25) (4,0.25)
\Ppoint0 \Pw:2,4
\Ppoint0 \Pb:1,3
\Ppoint1 \Pw:1,2,3
\Ppoint1 \Pb:4
}
```



One can define his own points using PGF commands. The points should be a macro taking two parameters for the coordinates, where the point should be drawn. For example, we can define diamond-shaped black and white points by

```
\def\Pwd#1#2{
\pgftransformshift{\pgfpointxy{#1}{#2}}
\pgfpathmoveto{\pgfpoint{-0.1em}{0em}}
\pgfpathlineto{\pgfpoint{0em}{0.15em}}
\pgfpathlineto{\pgfpoint{0.1em}{0em}}
\pgfpathlineto{\pgfpoint{0em}{-0.15em}}
\pgfpathclose
\pgfsetfillcolor{white}
\pgfusepath{stroke,fill}
\pgftransformreset
}
```

```
\def\Pbd#1#2{
\pgftransformshift{\pgfpointxy{#1}{#2}}
\pgfpathmoveto{\pgfpoint{-0.1em}{0em}}
\pgfpathlineto{\pgfpoint{0em}{0.15em}}
\pgfpathlineto{\pgfpoint{0.1em}{0em}}
\pgfpathlineto{\pgfpoint{0em}{-0.15em}}
\pgfpathclose
\pgfsetfillcolor{black}
\pgfusepath{stroke,fill}
\pgftransformreset
}
```

Now, we can modify our example

```
\BigPartition{
\Psingletons 0 to 0.25:1,4
\Psingletons 1 to 0.75:1,4
\Pline (2,0) (3,1)
\Pline (3,0) (2,1)
\Pline (2.75,0.25) (4,0.25)
\Ppoint0 \Pwd:2,4
\Ppoint0 \Pbd:1,3
\Ppoint1 \Pwd:1,2,3
\Ppoint1 \Pbd:4
}
```