The DCpic package

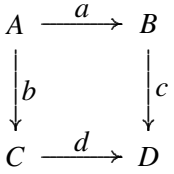
This package makes use of Pictex (rather, its more modern variant pictexwd) to produce commutative diagrams. The file dcpic.sty and a manual for DCpic can be found on CTAN or at http://hilbert.mat.uc.pt/~pedro/DCpic/man/.

The package provides 11 different types of arrows, and allows you to curve arrows. Arrows don't stretch to match their labels, and there is no switch to change the size of all labels. It is not possible to have labels on both sides of an arrow (but see at end).

For this file, I loaded it using \usepackage{pictexwd,dcpic}

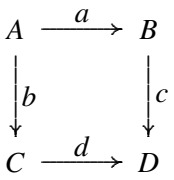
An object is given as \obj(1,2)[3]{4} where 1,2 are the integer coordinates of the object, 3 is an optional label, and 4 is the object itself.

For example:




```
\begindc{\commdiag}[50]
\obj(0,1)[aa]{$A$}
\obj(1,1)[bb]{$B$}
\obj(0,0)[cc]{$C$}
\obj(1,0){$D$}
\mor{aa}{bb}{$a$}
\mor{aa}{cc}{$b$}
\mor{bb}{$D$}{$c$}
\mor{cc}{$D$}{$d$}
\enddc
```

This code puts the object A at position $(0,1)$ and labels it aa (for internal purposes). It draws an arrow from the object A to the object B and puts a label a in the default position. When an object has no label, the object itself can be used. The "50" is the magnification factor. The same diagram is produced by the following code:




```
\begindc{\commdiag}[5]
\obj(0,10){$A$}
\obj(10,10){$B$}
...
\mor{cc}{dd}{$d$}
\enddc
```

Specifying a magnification of 5 allows you to position the nodes more finely.

There are the following arrows:

$A \xrightarrow{0} B$ `\solidarrow`

$A \dashrightarrow{1} B$ `\dashArrow`

$A \xrightarrow{2} B$ `\solidline`

$A \dashrightarrow{3} B$ `\dashline`

$A \xrightarrow{4} B$ `\dotline`

$A \xrightarrow{5} B$ `\injectionarrow`

$A \xrightarrow{6} B$ `\aplicationarrow`

$A \xrightarrow{7} B$ `\surjectivearrow`

$A \xrightarrow{8} B$ `\equalline`

$A \xrightarrow{9} B$ `\doublearrow`

$A \xrightarrow{10} B$ `10`

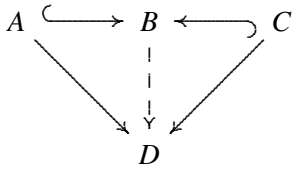
An arrow is invoked by

`\mor{source}{target}{label}[labelposition,type]`

where `labelposition` can be +1 (=`\atright`) or -1 (=`\atleft`) and `type` is 0,1,2,...,10 or the corresponding command in the above table. Alternatively, the arrow can be invoked by

`\mor(w,x)(y,z){label}[labelposition,type]`

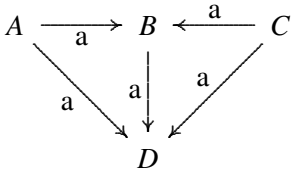
where (w,x) and (y,z) are the integer coordinates of the source and target. For example, both samples of code below produce the diagram:

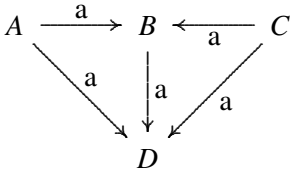



```
\[\begindc{\commdiag}[50]
\obj(0,1){$A$}
\obj(1,1){$B$}
\obj(2,1){$C$}
\obj(1,0){$D$}
\mor(0,1)(1,1){}[+1,3]
\mor(2,1)(1,1){}[+1,3]
\mor(0,1)(1,0){}
\mor(1,1)(1,0){}[+1,1]
\mor(2,1)(1,0){}
\enddc\]
```

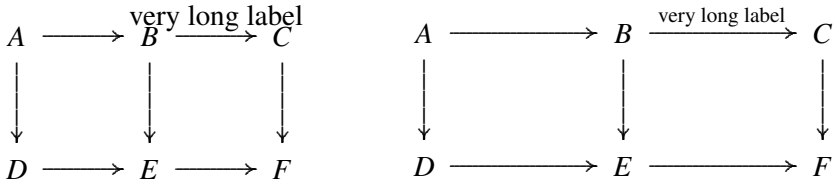
```
\[\begindc{\commdiag}[50]
\obj(0,1)[a]{$A$}
\obj(1,1)[b]{$B$}
\obj(2,1)[c]{$C$}
\obj(1,0)[d]{$D$}
\mor{a}{b}{}[\atright,\injectionarrow]
\mor{c}{b}{}[\atright,\injectionarrow]
\mor{a}{d}{}
\mor{b}{d}{}[\atright,\dashArrow]
\mor{c}{d}{}
\enddc\]
```

In the diagram at right below, all arrows have the (default) `\atright` option; in the diagram at left, they have the `\atleft` option.



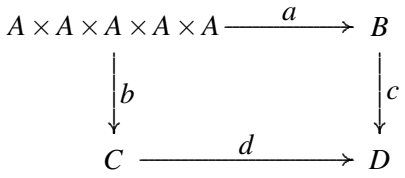


Arrows do not automatically stretch to match long labels:



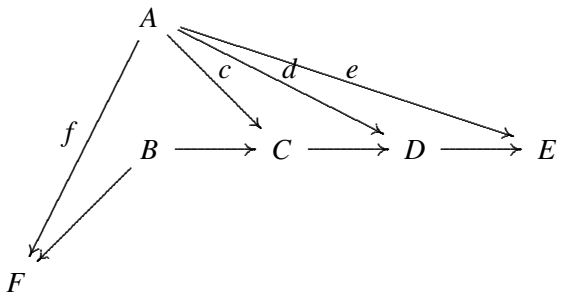
To fix this, replace `\begindc{0}[50]` with `\begindc{0}[5]`), put $A,B,C\ldots$ at `(0,10),(15,10),(30,10)` rather than `(0,1),(1,1),(2,1)` and shrink the label (`\scriptsize{very long label}`), as in the diagram at right.

Arrows do now stretch (or contract) to match large objects provided you use labels (rather than coordinates) to determine the start and end of the arrow:



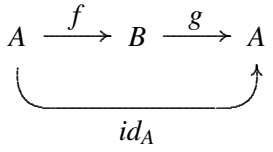

```
\begindc{\commdiag}[50]
\obj(0,1)[a]{$A\times A\times A\times A\times A$}
\obj(2,1)[b]{$B$}
\obj(0,0)[c]{$C$}
\obj(2,0)[d]{$D$}
\mor{a}{b}{$a$}
\mor{a}{c}{$b$}
\mor{b}{d}{$c$}
\mor{c}{d}{$d$}
\enddc
```

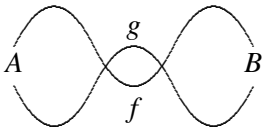
While the syntax of DCpic seems a little clumsy for simple diagrams, it makes it easy to construct complicated diagrams — once the objects are correctly placed, it is easy to add the arrows, as illustrated by,

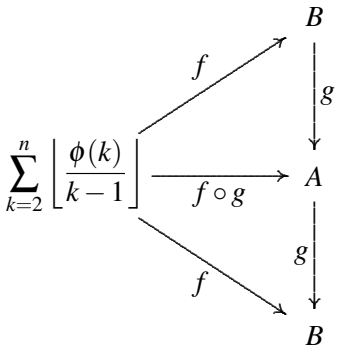



```
\begindc{\commdiag}[50]
\obj(1,2)[aa]{$A$}
\obj(1,1)[bb]{$B$}
\obj(2,1)[cc]{$C$}
\obj(3,1)[dd]{$D$}
\obj(4,1)[ee]{$E$}
\obj(0,0)[ff]{$F$}
\mor{aa}{ff}{$f$}[-1,0]
\mor{aa}{cc}{$c$}
\mor{aa}{dd}{$d$}
\mor{aa}{ee}{$e$}
\mor{bb}{ff}{}
\mor{bb}{cc}{}
\mor{cc}{dd}{}
\mor{dd}{ee}{}
\enddc
```

One of DCpic's strengths is its ability to draw curved arrows. Here are some examples from the "examples" files with the package.

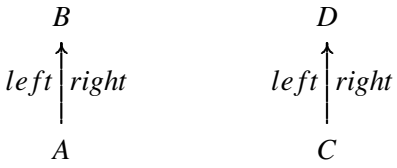






To use DCpic to draw "direct graphs" or "unidirect graph" instead of commutative diagrams, invoke it with `\begindc{digraph}` or `\begindc{unidigraph}`. There is an optional argument `placement of the object` for objects, which (apparently) has no effect for commmutative diagrams, and an option argument `correction factors` which, when invoked with numbers other than [10,10], will move the starting or ending point of the arrow.

Stuart Ambler pointed out to me that it is possible to put labels on both sides of an arrow by making two coincident arrows between a pair of objects, or by making arrows between two coincident pairs of objects.




```
\begindc{\commdiag}[5]
\obj(00,00)[objA]{$A$}
\obj(00,10)[objB]{$B$}
\mor{objA}{objB}{$left$}[\atleft,\solidarrow]
\mor{objA}{objB}{$right$}[\atright,\solidarrow]
\obj(20,00)[objC]{$C$}
\obj(20,10)[objD]{$D$}
\obj(20,00)[objCC]{}
\obj(20,10)[objDD]{}
\mor{objC}{objD}{$left$}[\atleft,\solidarrow]
\mor{objCC}{objDD}{$right$}[\atright,\solidarrow]
\enddc
```