Reverse engineerable LaTeXexamples:

Beamer presentations

Aarhus University

March 15, 2020

This is a very simple slide, which contains some math symbols such as σ, and also a math equation such as

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

Here is a theorem with a proof

Theorem (Euler)

If n and a are coprime positive integers, then $a^{\phi(n)} \equiv 1 \mod n$

Proof.

Left to the reader. □

It is quite common to have your slide divided into two or more parts. This is done using columns.

Here is some text in the left column

Here is some text in the right column

You can have code snippets on your slides just the same way you do in the documents. You only have to mark the frame *fragile*, such that overflows do not immediately break the compilation.

```
1   Theorem strong_induction :
2     forall P : nat → Prop,
3     (forall n : nat, (forall m : nat, m < n → P m) → P n) →
4     forall n : nat, P n.
5   Proof.
6     intros P IH_strong n.
7     assert (H : forall k, k <= n → P k).
8     { ... }
9     now apply H.
10  Qed.
```

Listing 1: Exercise on proving strong induction in [Pie+]

Tables

Tables are just as simple as you are used to.

A cell	Another one
γ	β

Table: A table

Graphics

Figures are also exactly as easy. That means, you can actually have the whole figure in another file, that you can include in both an article and in a presentation!

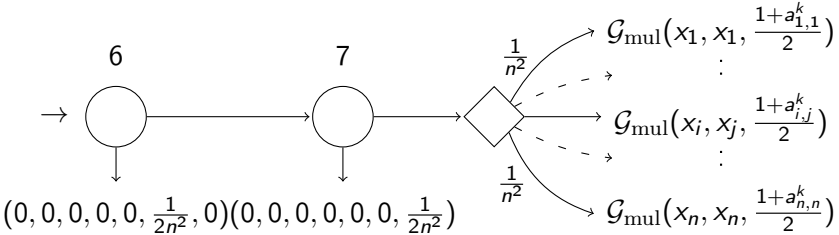


Figure: The label for the figure.

pause command

To animate slides it is luckily not necessary to copy paste all of the code and change one thing after the other! Otherwise Bærbak would definitely have been very sad. The simplest animations you need is slowly revealing the slide, such as the bullet points below. This is done using the `\pause` command.

- An item, that is shown at the very beginning.

pause command

To animate slides it is luckily not necessary to copy paste all of the code and change one thing after the other! Otherwise Bærbak would definitely have been very sad.
The simplest animations you need is slowly revealing the slide, such as the bullet points below. This is done using the `\pause` command.

- An item, that is shown at the very beginning.
- An item, that is first shown on the "next slide"

pause command

To animate slides it is luckily not necessary to copy paste all of the code and change one thing after the other! Otherwise Bærbak would definitely have been very sad. The simplest animations you need is slowly revealing the slide, such as the bullet points below. This is done using the `\pause` command.

- An item, that is shown at the very beginning.
- An item, that is first shown on the "next slide"

It of course also works for enumeration

1. To not make the LaTeX too heavy on commands the items support a shorthand for the onslide notation on the next slide onslide command further explained on the next slide

pause command

To animate slides it is luckily not necessary to copy paste all of the code and change one thing after the other! Otherwise Bærbak would definitely have been very sad. The simplest animations you need is slowly revealing the slide, such as the bullet points below. This is done using the `\pause` command.

- An item, that is shown at the very beginning.
- An item, that is first shown on the "next slide"

It of course also works for enumeration

1. To not make the LaTeX too heavy on commands the items support a shorthand for the onslide notation on the next slide onslide command further explained on the next slide
2. Finally, if you need to compile handouts without animations, just add `handout` in the `documentclass` above

onslide command

To do more complicated animations you want to trigger parts of the slide at different times. You do this using the `\onslide<a-b>` where *a*, and *b* are optional slide numbers.

This paragraph is shown on the first two steps of the slides

onslide command

To do more complicated animations you want to trigger parts of the slide at different times. You do this using the `\onslide<a-b>` where *a*, and *b* are optional slide numbers.

This paragraph is shown on the second step and forwards

This paragraph is shown on the second and third steps of the slide

This paragraph is shown on the first two steps of the slides

onslide command

To do more complicated animations you want to trigger parts of the slide at different times. You do this using the `\onslide<a-b>` where *a*, and *b* are optional slide numbers.
This paragraph is shown on the second step and forwards
This paragraph is shown on the second and third steps of the slide

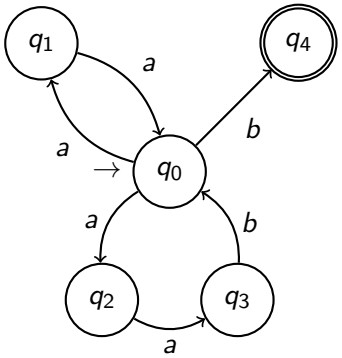
onslide command

To do more complicated animations you want to trigger parts of the slide at different times. You do this using the `\onslide<a-b>` where *a*, and *b* are optional slide numbers. This paragraph is shown on the second step and forwards

This paragraph is shown on the fourth (last) step of the slides

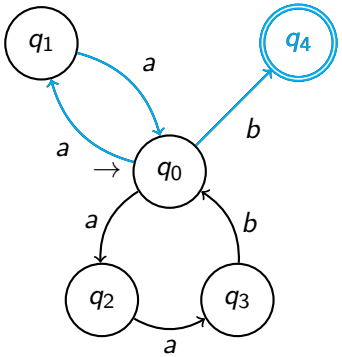
Animated Tikz

Here is a simple example using the `\onslide` command to trigger different parts of the slide at different times.



Animated Tikz

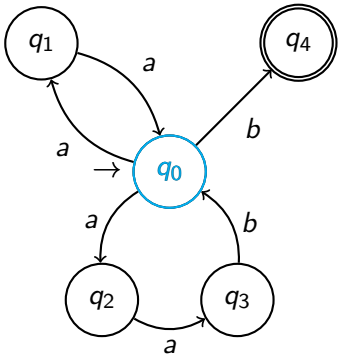
Here is a simple example using the `\onslide` command to trigger different parts of the slide at different times.



Animated Tikz

Here is a much more complicated example together with an animation in sync below.

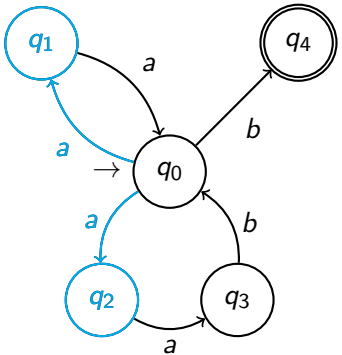
$\{q_0\}$



Animated Tikz

Here is a much more complicated example together with an animation in sync below.

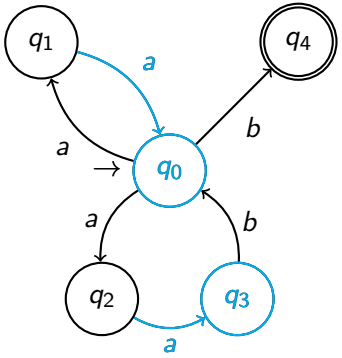
$$\{q_0\} \to \{q_1, q_2\}$$



Animated Tikz

Here is a much more complicated example together with an animation in sync below.

$$\{q_0\} \to \{q_1, q_2\}$$
$$\to \{q_0, q_3\}$$

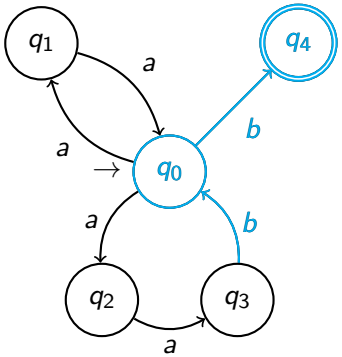


Animated Tikz

Here is a much more complicated example together with an animation in sync below.

$$\begin{aligned}\{q_0\} &\to \{q_1, q_2\} \\ &\to \{q_0, q_3\} \\ &\to \{q_0, q_4\}\end{aligned}$$

Are any of the final states accepting?

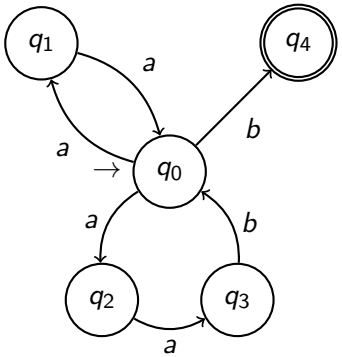


Animated Tikz

Here is a much more complicated example together with an animation in sync below.

$$\begin{aligned}\{q_0\} &\to \{q_1, q_2\} \\ &\to \{q_0, q_3\} \\ &\to \{q_0, q_4\}\end{aligned}$$

Are any of the final states accepting? Yes, q_4 is!



By escaping from lstlisting out to LaTeX using the *@...@* macro you can also slowly reveal code. This could be useful for showing a *Coq* proof as below

```
1  Theorem plus_n_O : forall n:nat, n = n + 0.
2  Proof.
```

By escaping from lstlisting out to LaTeX using the *@...@* macro you can also slowly reveal code. This could be useful for showing a *Coq* proof as below

```
1  Theorem plus_n_O : forall n:nat, n = n + 0.
2  Proof.
3    induction n as [| n' IHn'].
```

By escaping from lstlisting out to LaTeX using the *@...@* macro you can also slowly reveal code. This could be useful for showing a *Coq* proof as below

```
1  Theorem plus_n_O : forall n:nat, n = n + 0.
2  Proof.
3    induction n as [| n' IHn'].
4    - (* n = 0 *)
5      reflexivity.
```

By escaping from lstlisting out to LaTeX using the *@...@* macro you can also slowly reveal code. This could be useful for showing a *Coq* proof as below

```
1  Theorem plus_n_O : forall n:nat, n = n + 0.
2  Proof.
3    induction n as [| n' IHn'].
4    - (* n = 0 *)
5      reflexivity.
6    - (* n = S n' *)
7      simpl. rewrite ← IHn'. reflexivity.
```

By escaping from lstlisting out to LaTeX using the *@...@* macro you can also slowly reveal code. This could be useful for showing a *Coq* proof as below

```
1  Theorem plus_n_O : forall n:nat, n = n + 0.
2  Proof.
3    induction n as [| n' IHn'].
4    - (* n = 0 *)
5      reflexivity.
6    - (* n = S n' *)
7      simpl. rewrite ← IHn'. reflexivity.
8  Qed.
```

Benjamin C. Pierce, Arthur Azevedo de Amorim, Chris Casinghino, Marco Gaboardi, Michael Greenberg, Cătălin Hriţcu, Vilhelm Sjöberg, Brent Yorgey, Loris D'Antoni, Andrew W. Appel, Arthur Chargueraud, Anthony Cowley, Jeffrey Foster, Dmitri Garbuzov, Michael Hicks, Ranjit Jhala, Greg Morrisett, Jennifer Paykin, Mukund Raghothaman, Chung-chieh Shan, Leonid Spesivtsev, Andrew Tolmach, Philip Wadler, Stephanie Weirich, and Steve Zdancewic. *Software Foundations - Volume 1: Discrete Math in Coq (Alpha)*.