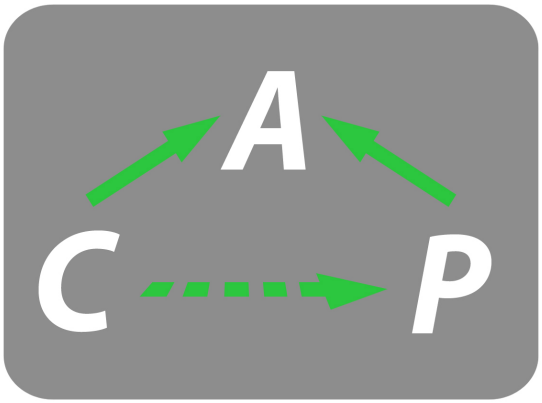
A constructive approach to finitely presented functors

Sebastian Posur

University of Siegen

September 28, 2017



Finitely presented functors

Finitely presented functors

Q: Is it possible to model the category of finitely presented functors on the computer?

Finitely presented functors

Q: Is it possible to model the category of finitely presented functors on the computer?

What are finitely presented functors?

Finitely presented functors

Let **A** be an abelian category

Finitely presented functors

Let **A** be an abelian category (e.g., R-Mod for a ring R).

Finitely presented functors

Let **A** be an abelian category (e.g., R-Mod for a ring R).

Definition

A functor $F : \mathbf{A} \to \mathbf{Ab}$ is called **finitely presented** if there exist $A, B \in \mathbf{A}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(B, -) \longleftarrow \mathrm{Hom}(A, -)$$

Finitely presented functors

Let **A** be an abelian category (e.g., R-Mod for a ring R).

Definition

A functor $F : \mathbf{A} \to \mathbf{Ab}$ is called **finitely presented** if there exist $A, B \in \mathbf{A}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(B, -) \longleftarrow \mathrm{Hom}(A, -)$$

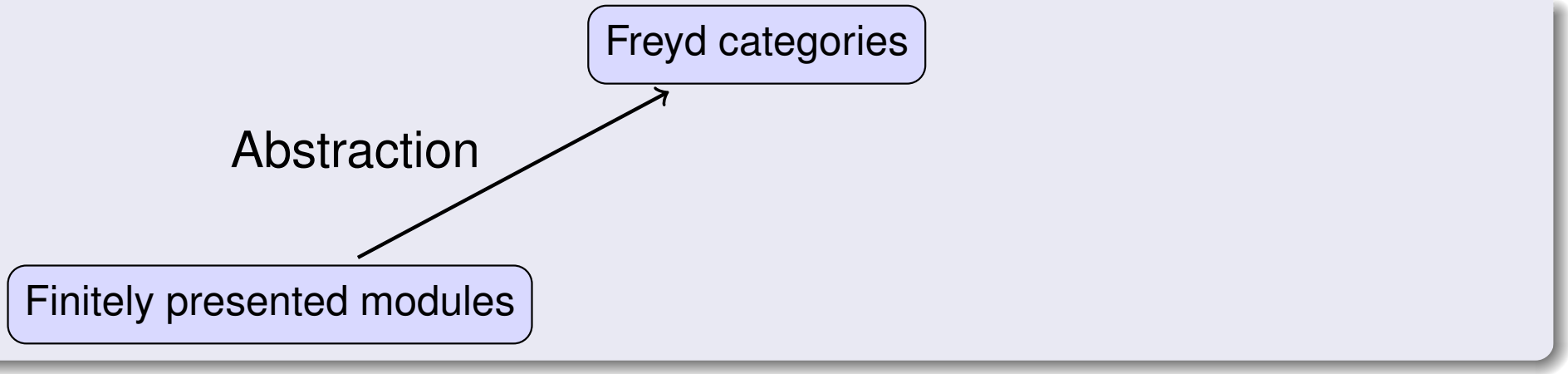
Applications in representation theory of finite dimensional algebras, algebraic geometry, model theory, homological algebra.

Route

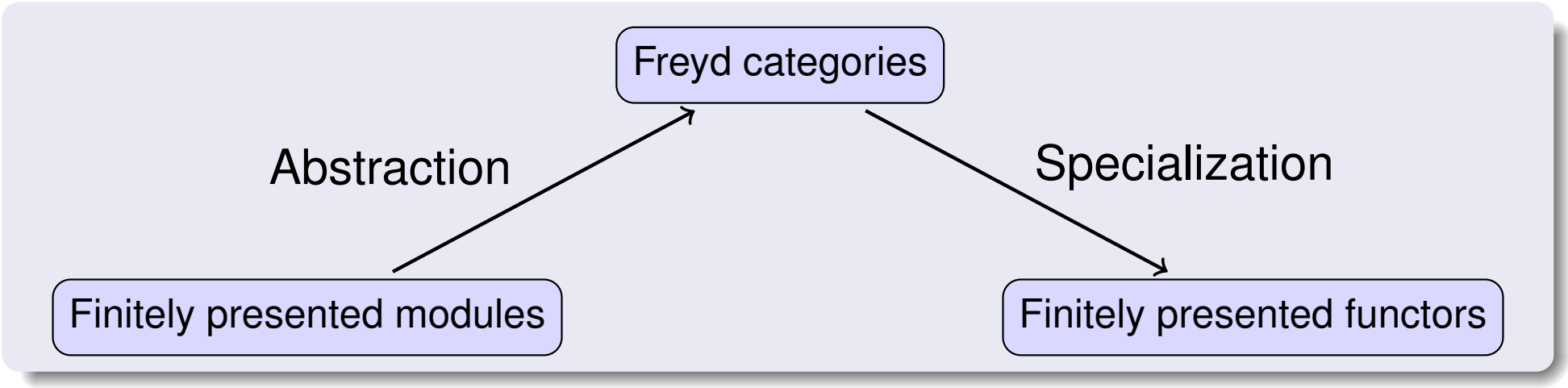
Route

Finitely presented modules

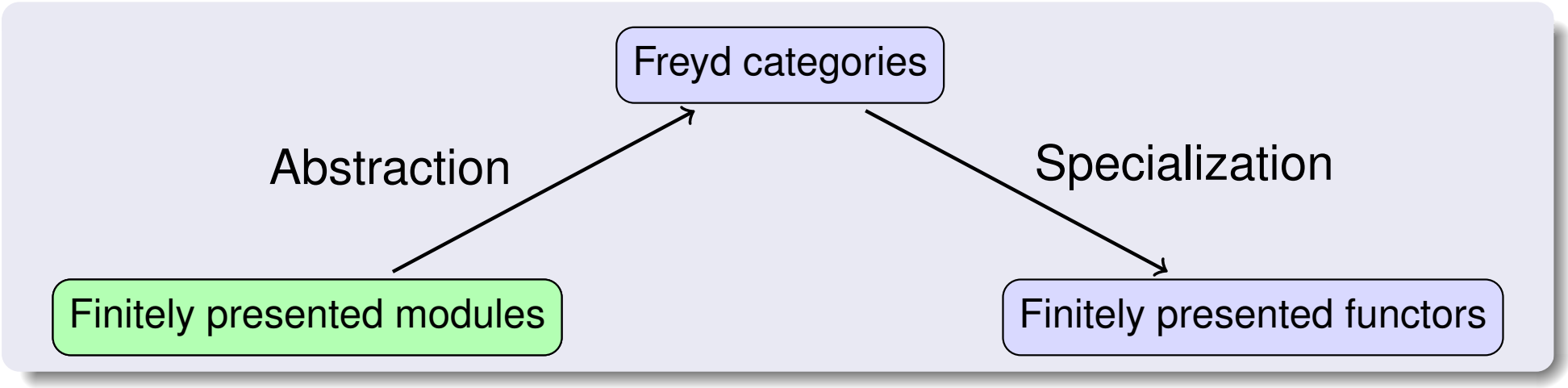
Route


Freyd categories
Abstraction
Finitely presented modules

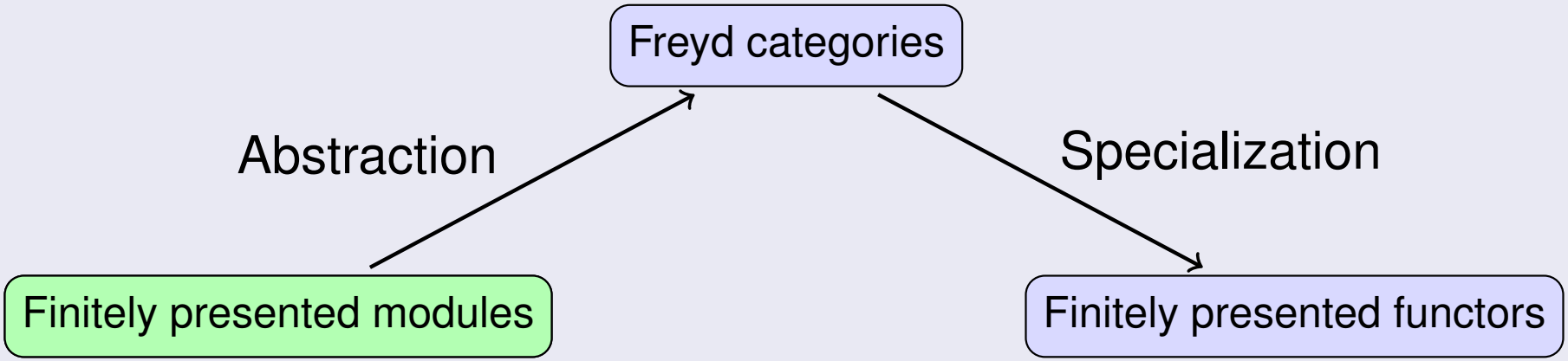
Route


Freyd categories
Abstraction
Specialization
Finitely presented modules
Finitely presented functors

Route

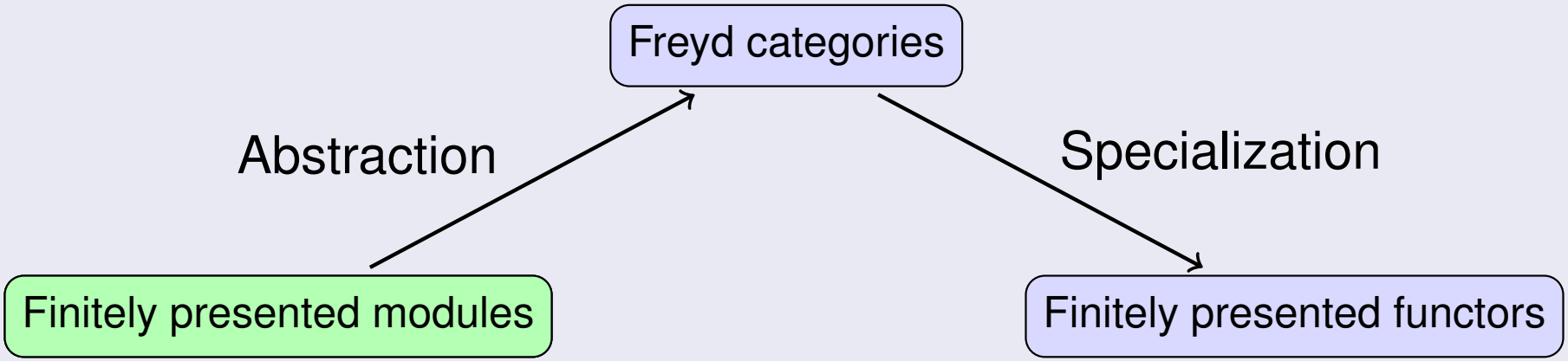

Freyd categories
Abstraction
Specialization
Finitely presented modules
Finitely presented functors

Route



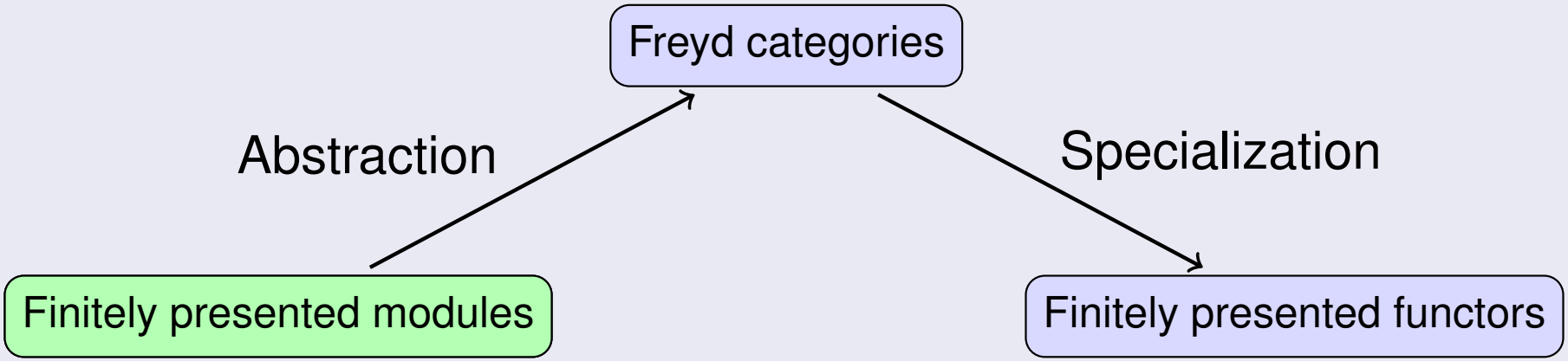
1. Data structures

Route



1. Data structures
2. Algorithms

Route



1. Data structures
2. Algorithms

Finitely presented modules: Data structures

Let R be a ring

Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod

Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

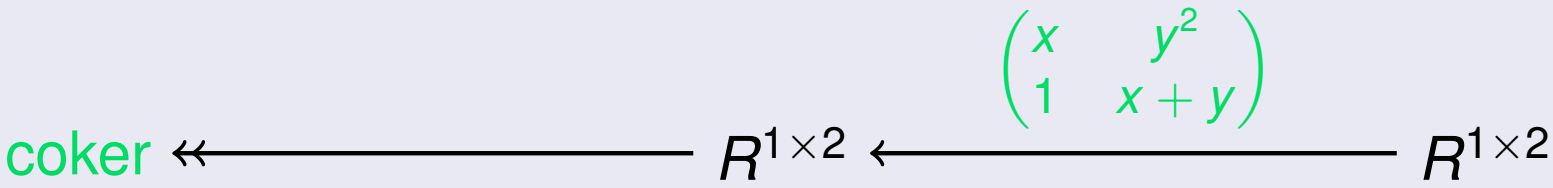
Category of finitely presented modules: R-fpmod

$$R^{1\times 2} \xleftarrow{\begin{pmatrix} x & y^2 \\ 1 & x+y \end{pmatrix}} R^{1\times 2}$$

Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod



Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod

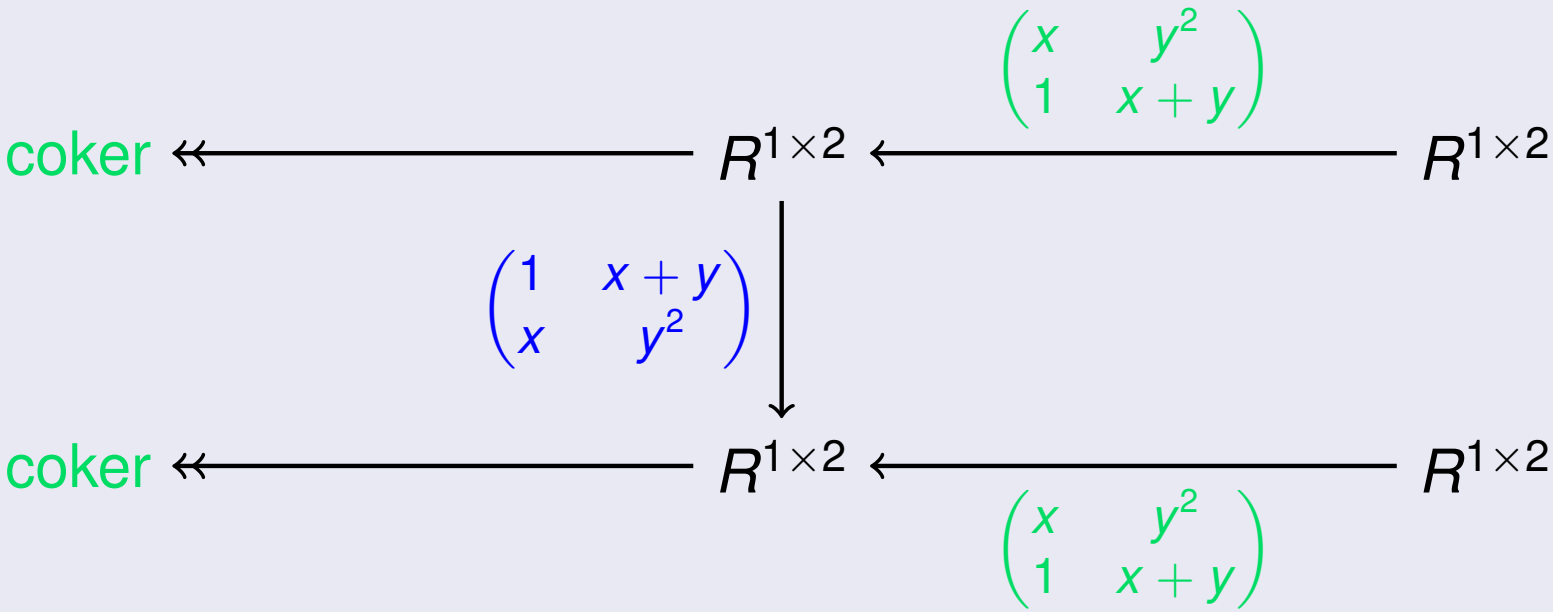
$$\text{coker} \xleftarrow{\quad} R^{1\times 2} \xleftarrow{\begin{pmatrix} x & y^2 \\ 1 & x+y \end{pmatrix}} R^{1\times 2}$$

$$\text{coker} \xleftarrow{\quad} R^{1\times 2} \xleftarrow{\begin{pmatrix} x & y^2 \\ 1 & x+y \end{pmatrix}} R^{1\times 2}$$

Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

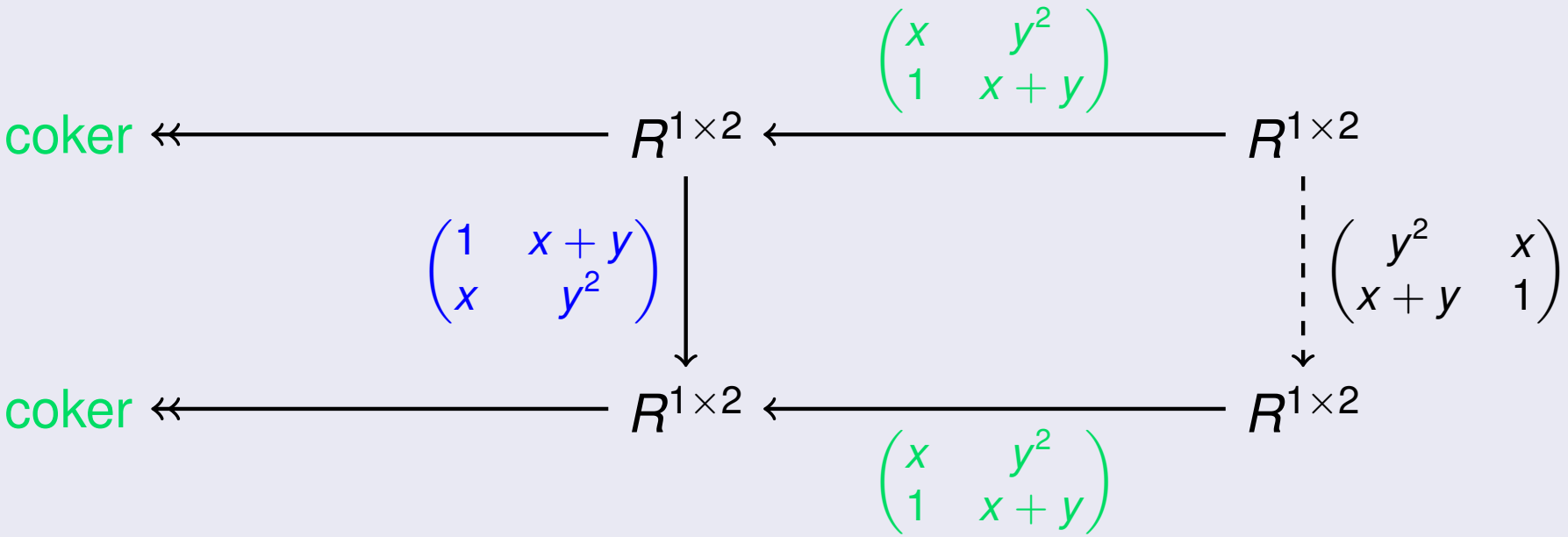
Category of finitely presented modules: R-fpmod



Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

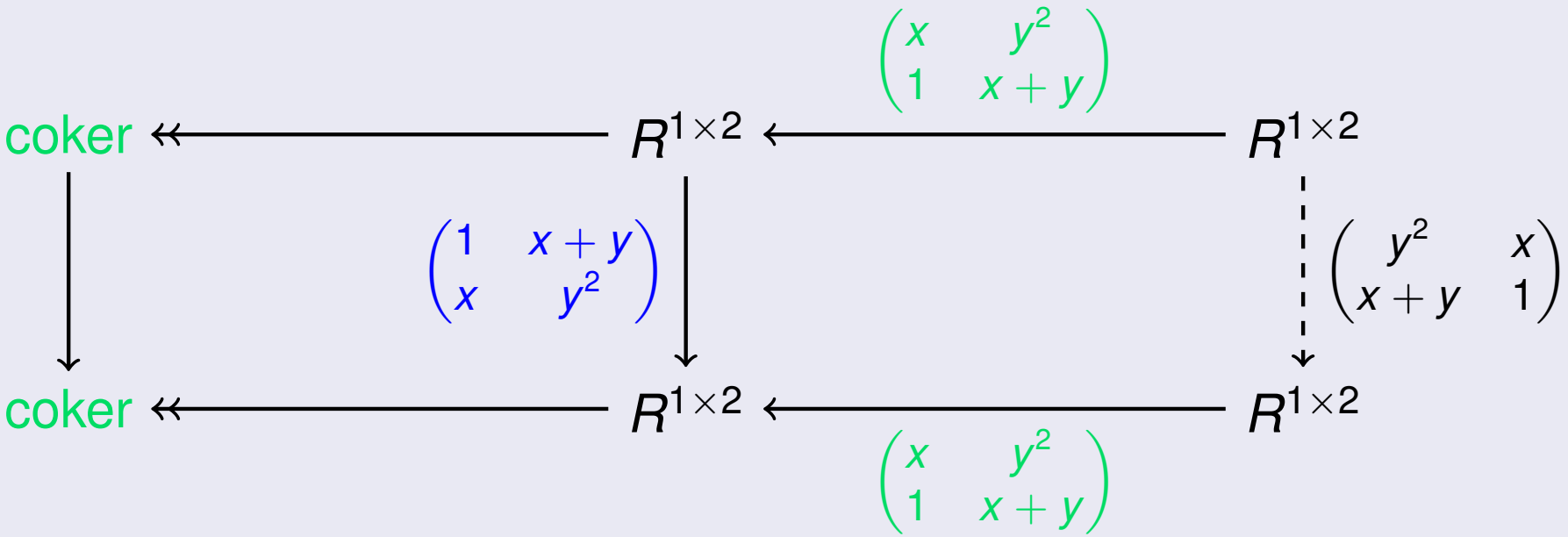
Category of finitely presented modules: R-fpmod



Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

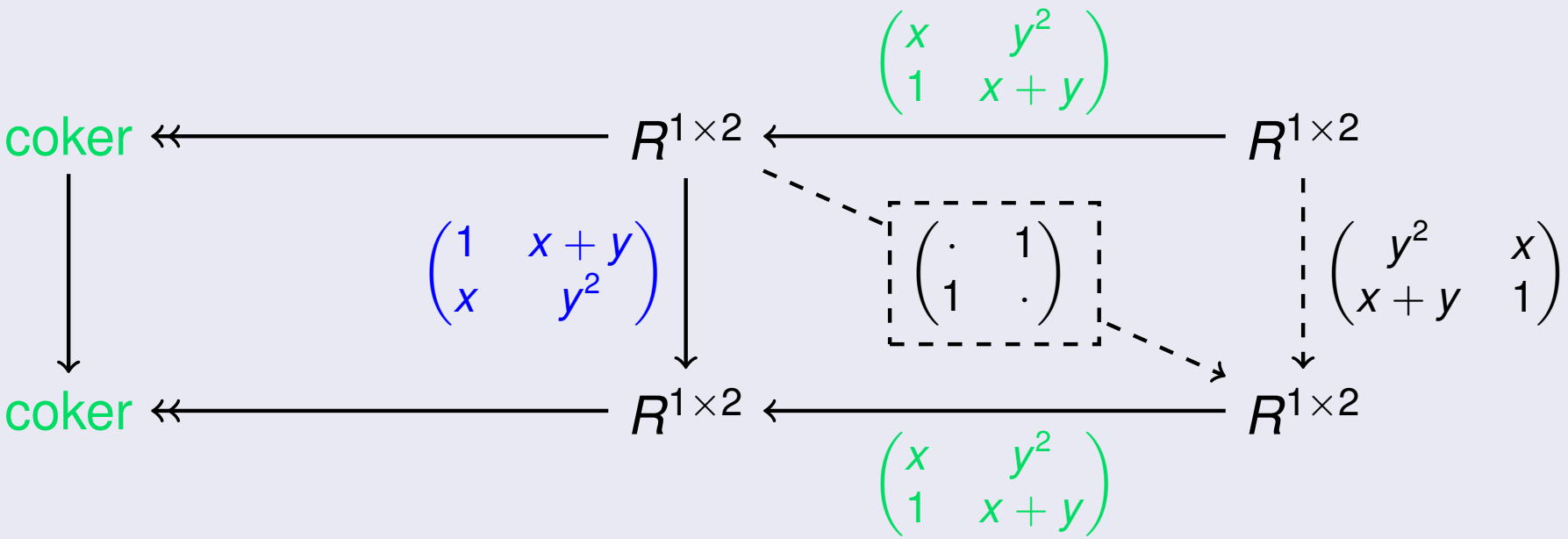
Category of finitely presented modules: R-fpmod



Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

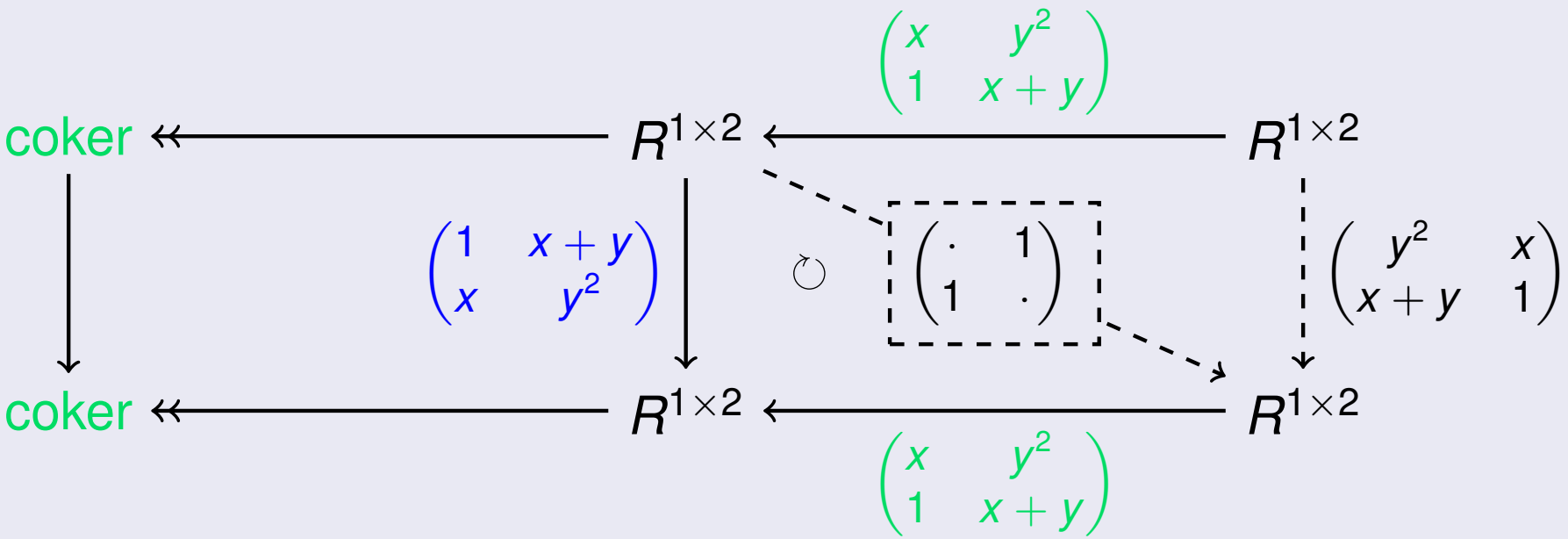
Category of finitely presented modules: R-fpmod



Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

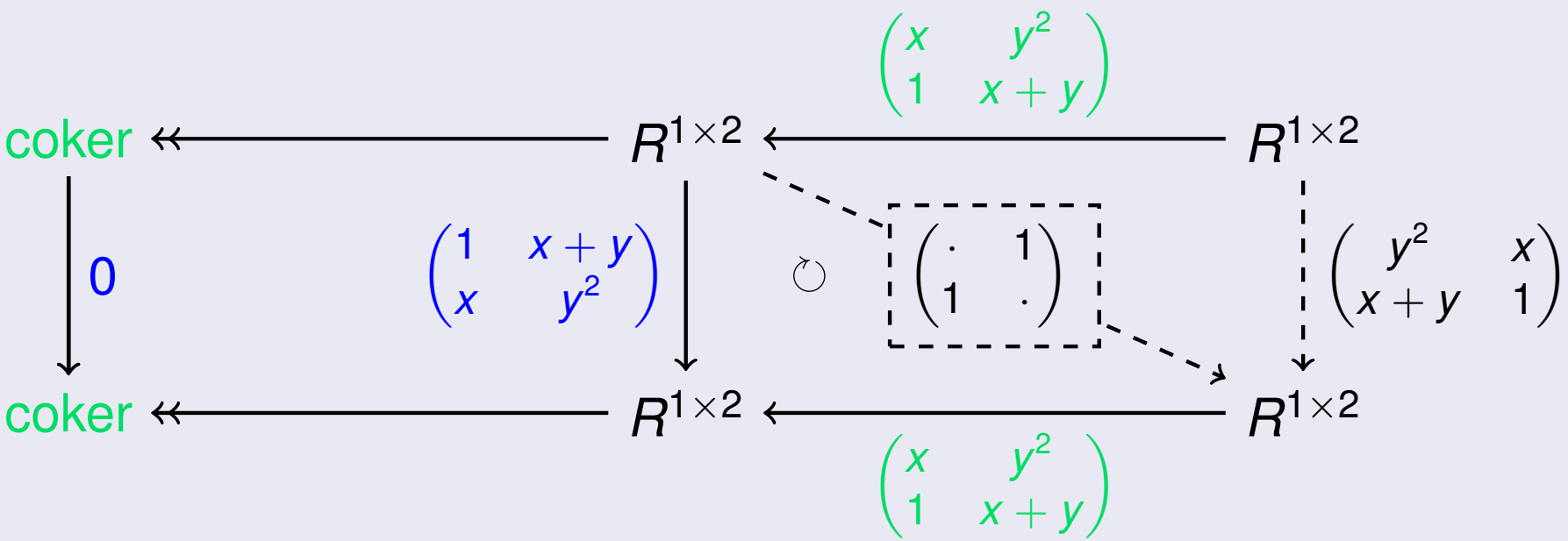
Category of finitely presented modules: R-fpmod



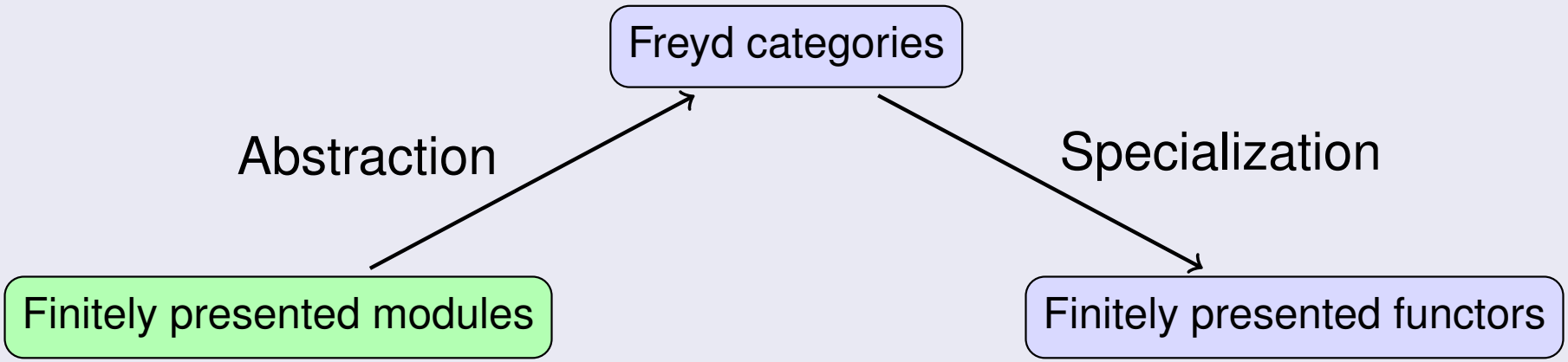
Finitely presented modules: Data structures

Let R be a ring, e.g., $R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod

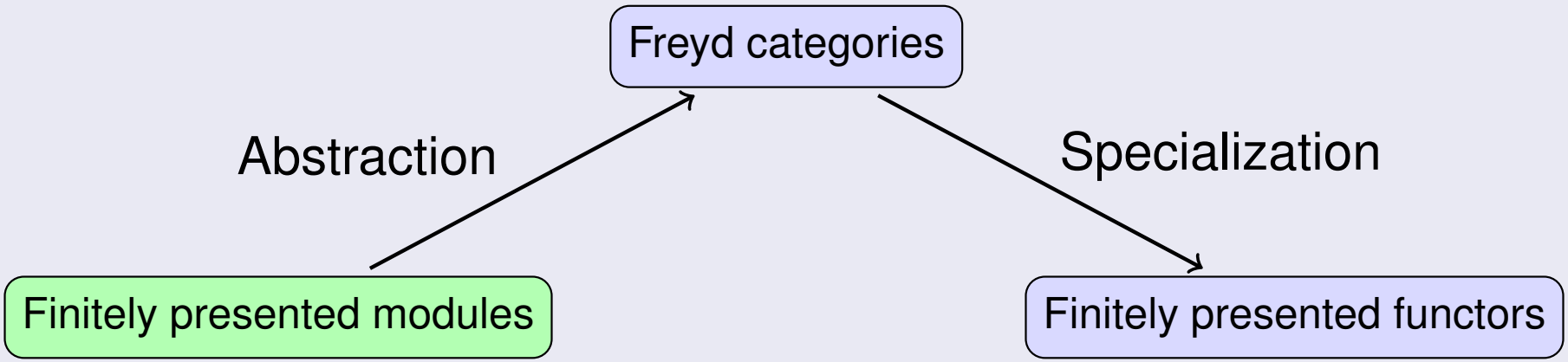


Route



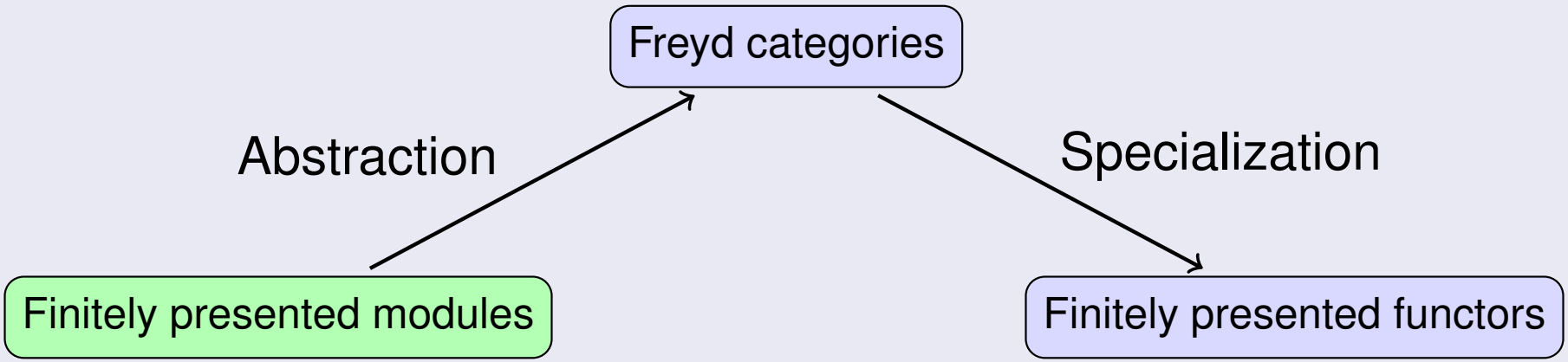
1. Data structures
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Finitely presented modules: Algorithms

Algorithms in R-fpmod

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition
- Addition

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition
- Addition
- Direct sums

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition
- Addition
- Direct sums
- Kernels

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition
- Addition
- Direct sums
- Kernels
- Cokernels

Finitely presented modules: Algorithms

Algorithms in R-fpmod

- Composition
- Addition
- Direct sums
- Kernels
- Cokernels

Algorithms for the defining axioms of an abelian category.

Finitely presented modules: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

Finitely presented modules: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.

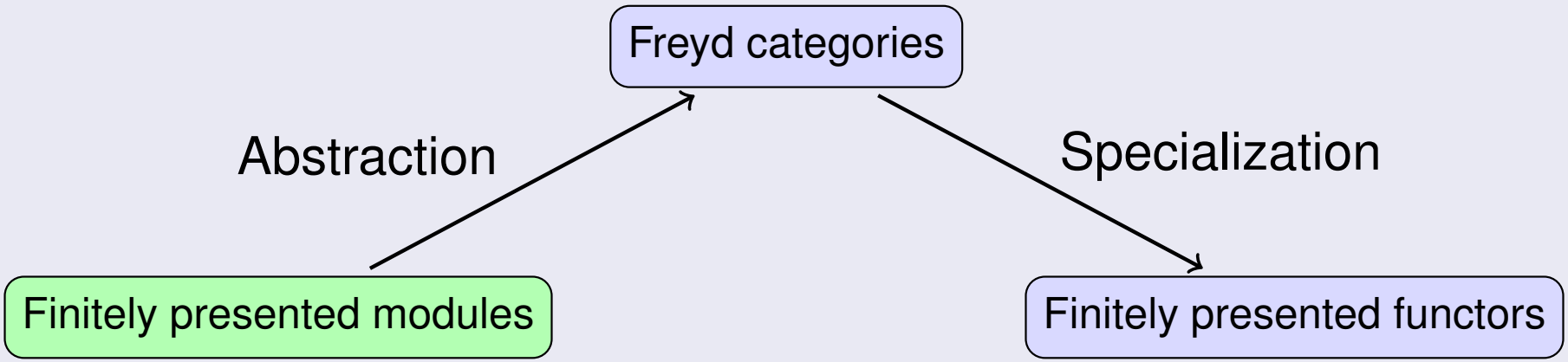
Finitely presented modules: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

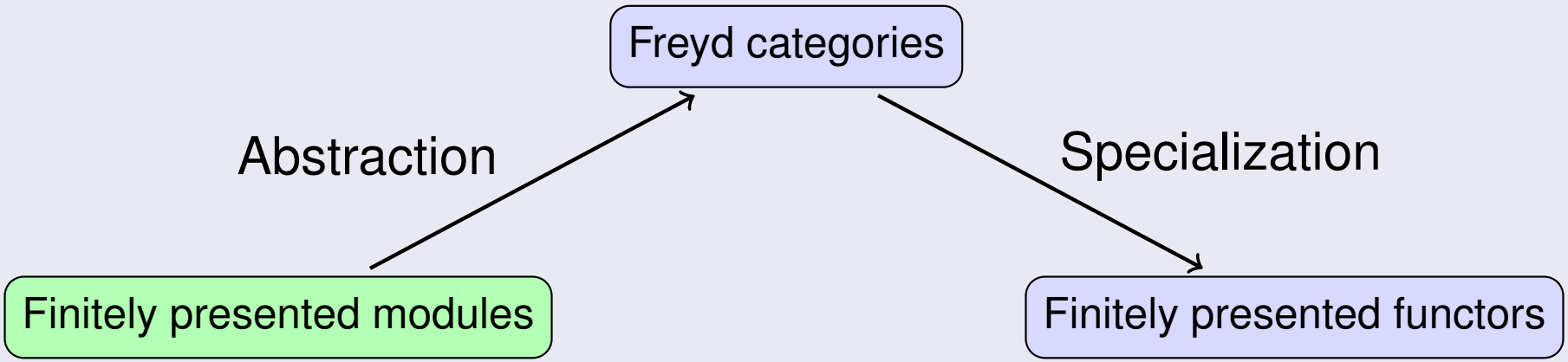
- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Route



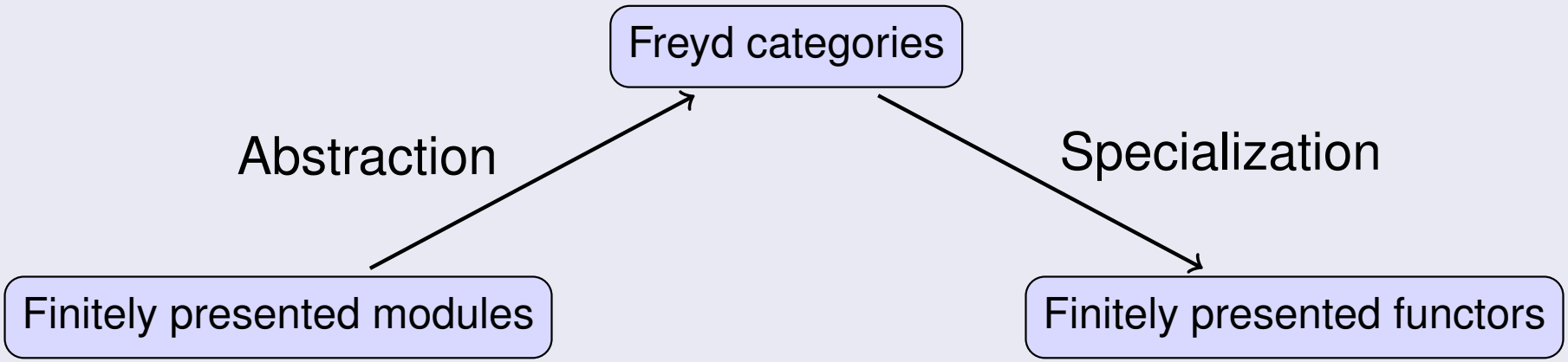
1. Data structures ✓
2. Algorithms

Route



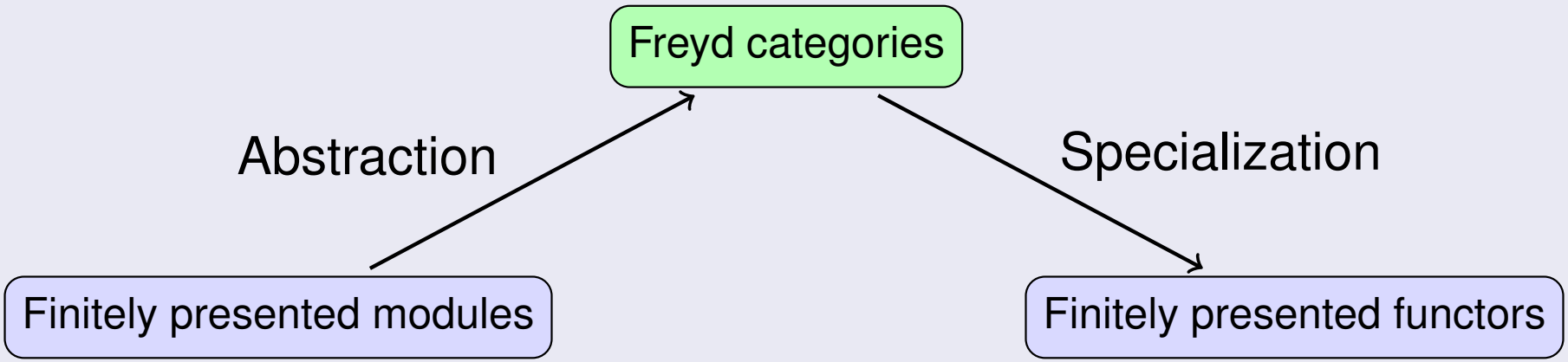
1. Data structures ✓
2. Algorithms ✓

Route



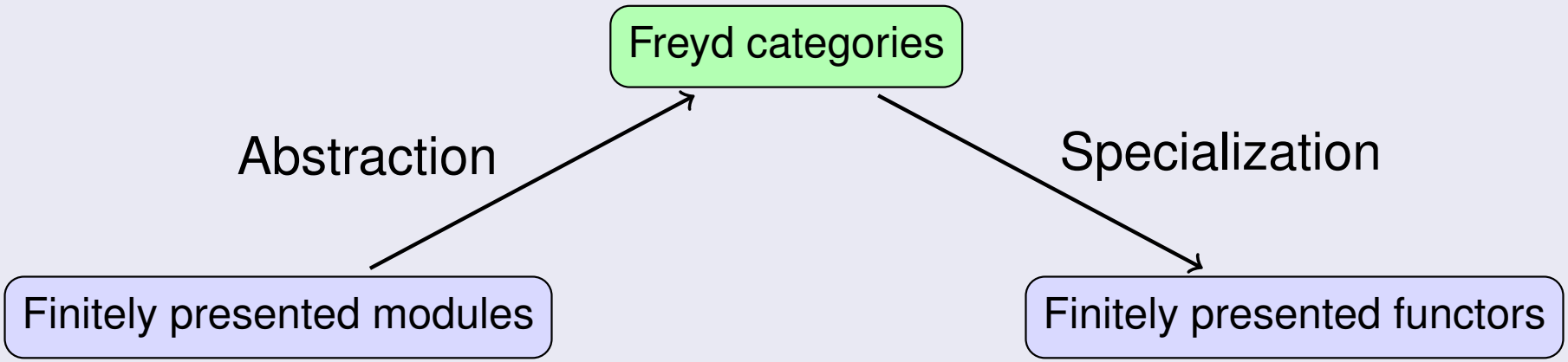
1. Data structures ✓
2. Algorithms ✓

Route



1. Data structures
2. Algorithms

Route

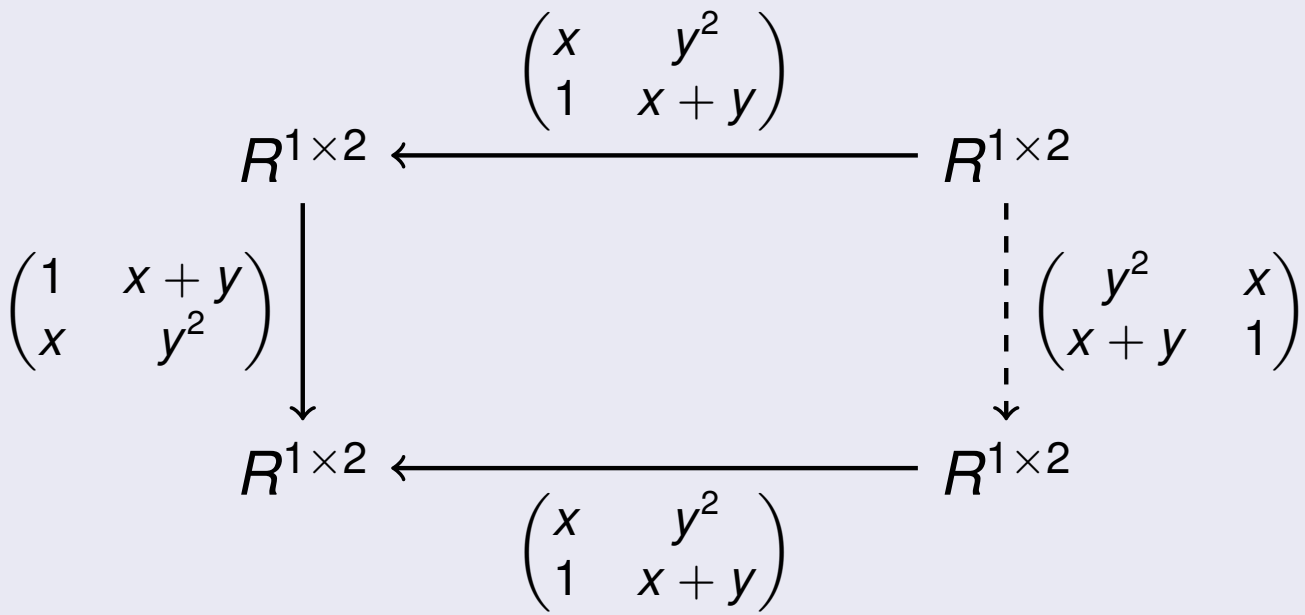


1. Data structures
2. Algorithms

Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

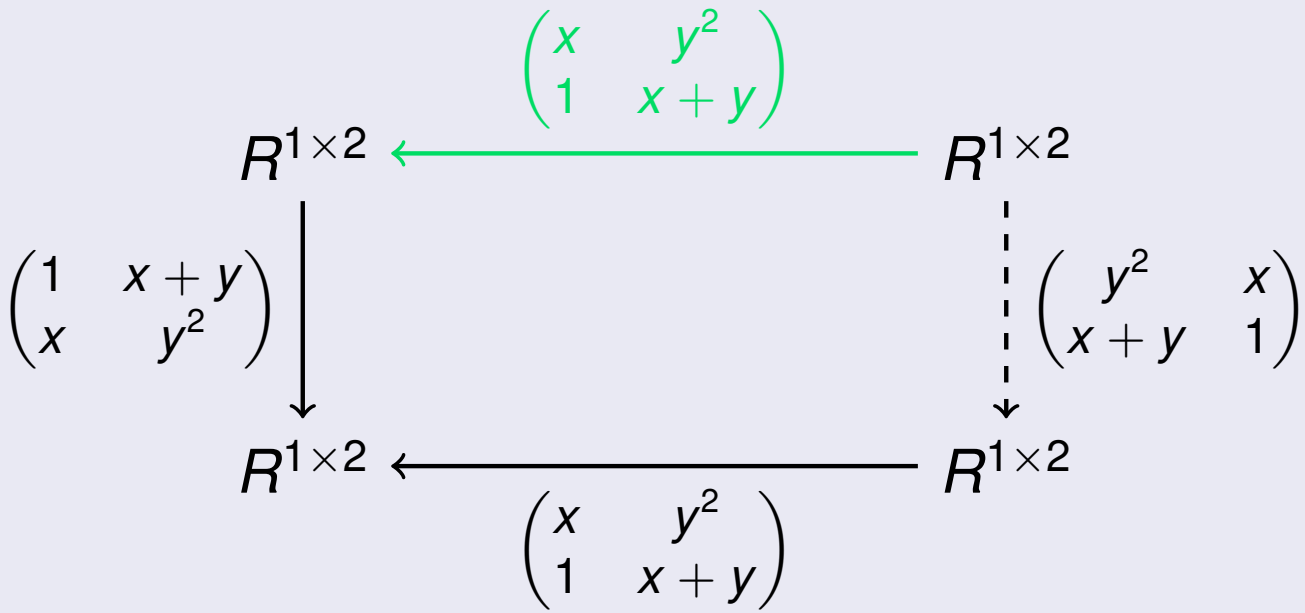
Category of finitely presented modules: R-fpmod



Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

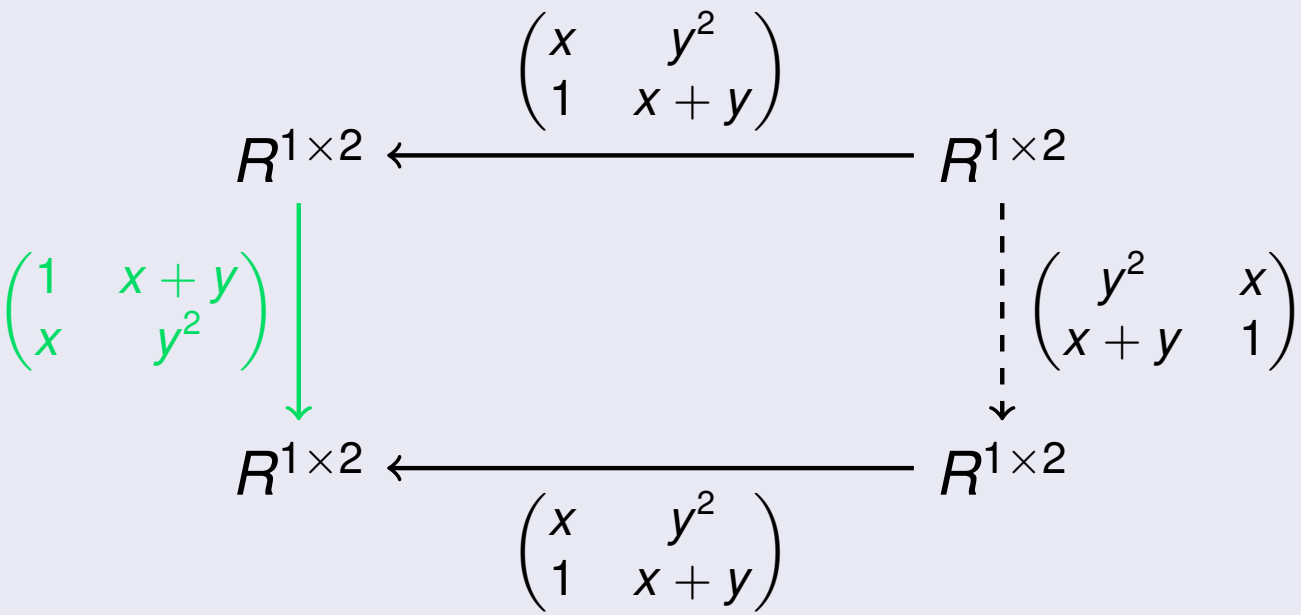
Category of finitely presented modules: R-fpmod



Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

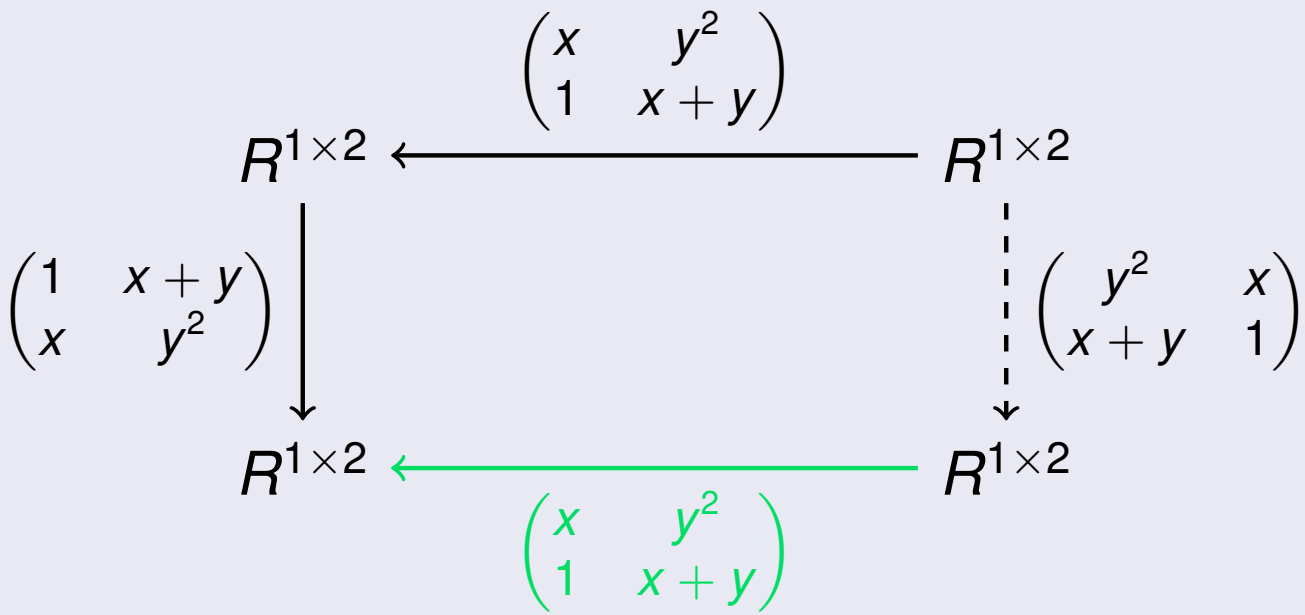
Category of finitely presented modules: R-fpmod



Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

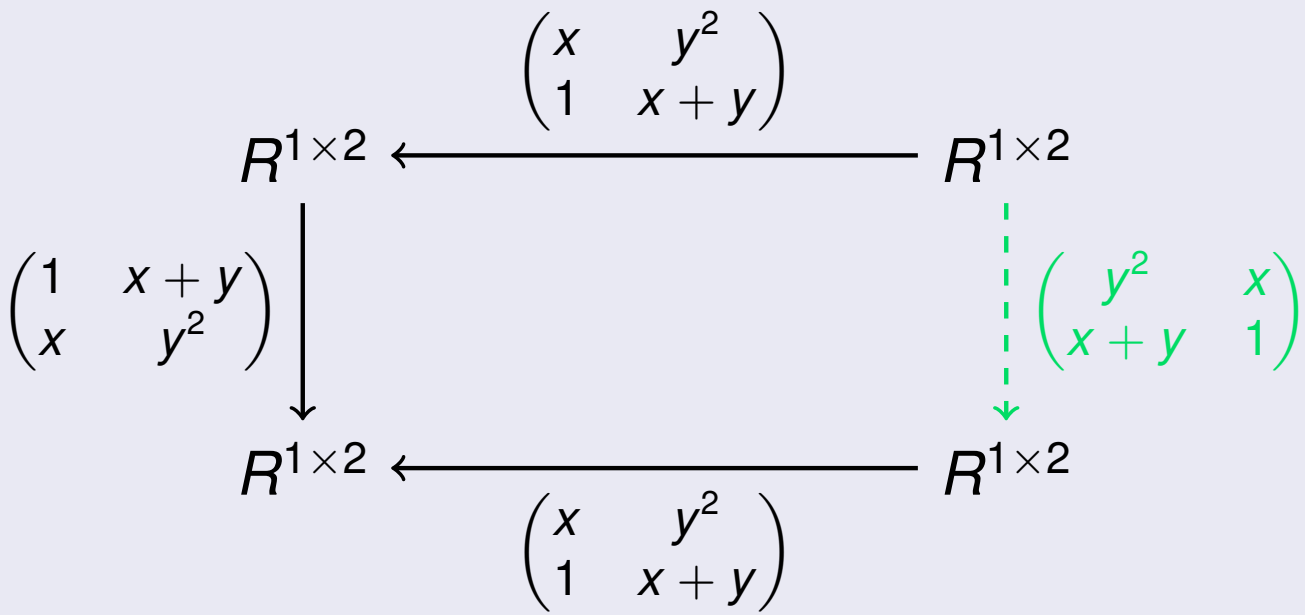
Category of finitely presented modules: R-fpmod



Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

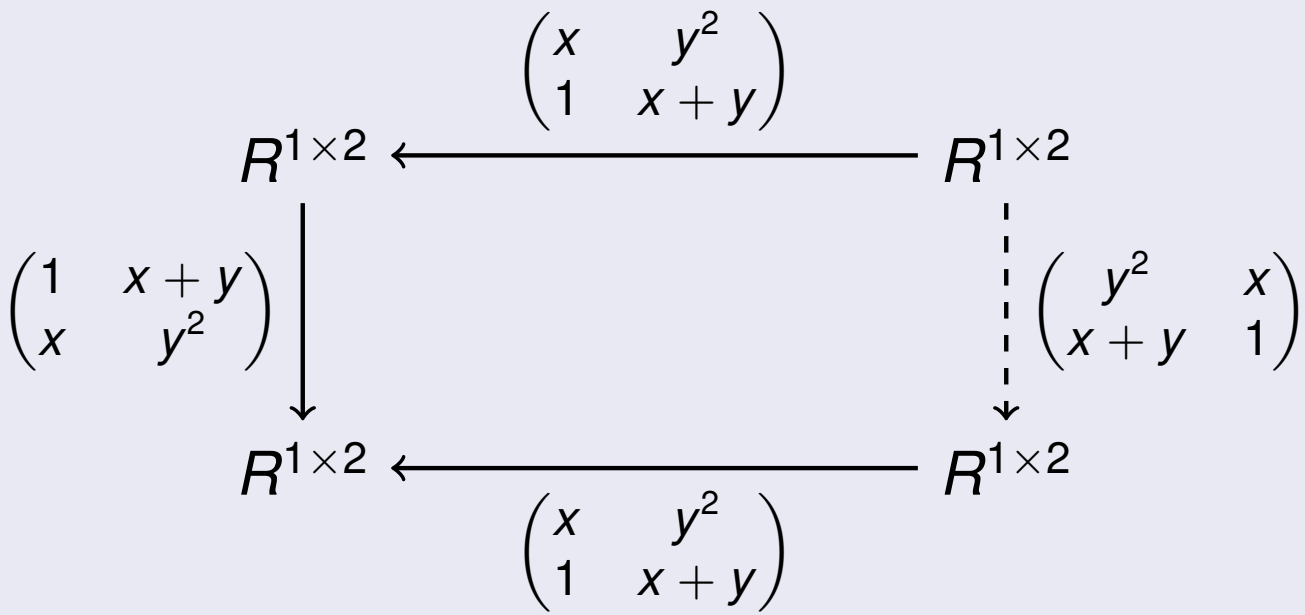
Category of finitely presented modules: R-fpmod



Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod



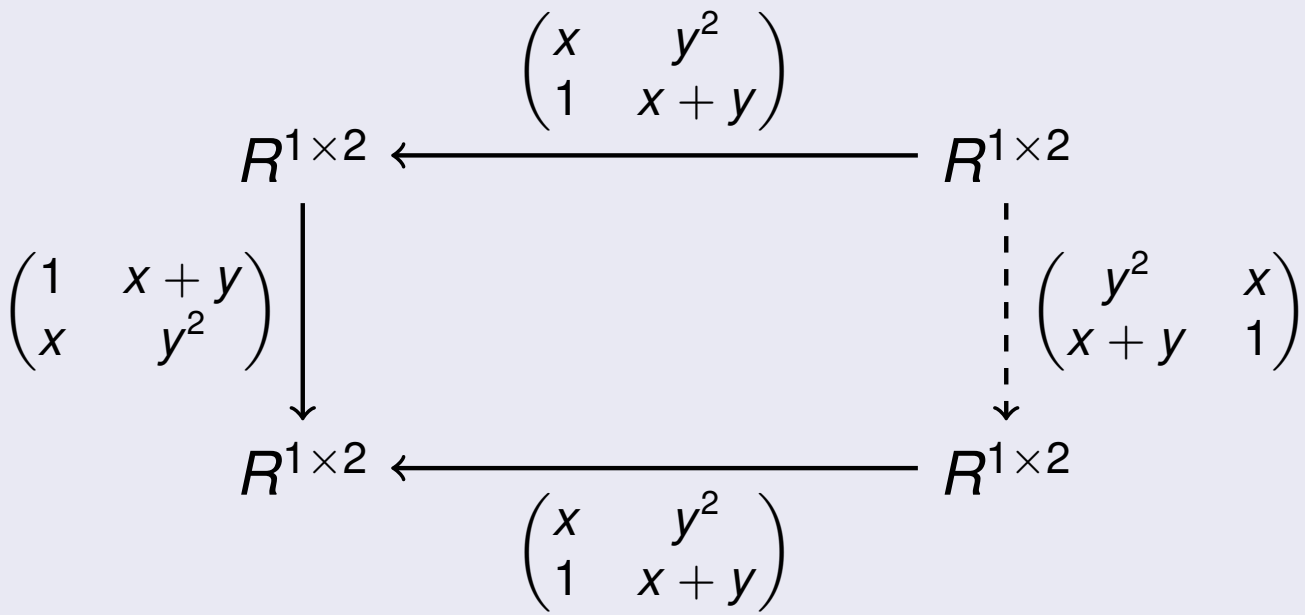
Observation: Data structures are built from objects and morphisms in

$$\mathrm{Rows}_R := \{R^{1\times n} \xleftarrow{A} R^{1\times m} \mid m, n \in \mathbb{N}_0, A \in R^{m\times n}\}.$$

Freyd categories: Data structures

$R = \mathbb{Q}[x, y]$.

Category of finitely presented modules: R-fpmod



Observation: Data structures are built from objects and morphisms in

$$\mathrm{Rows}_R := \{R^{1\times n} \xleftarrow{A} R^{1\times m} \mid m, n \in \mathbb{N}_0, A \in R^{m\times n}\}.$$

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.
- Morphisms:

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.

$$A \xleftarrow{\rho_A} R_A$$

- Morphisms:

Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.

$$A \xleftarrow{\rho_A} R_A$$

- Morphisms:

$$B \xleftarrow{\rho_B} R_B$$

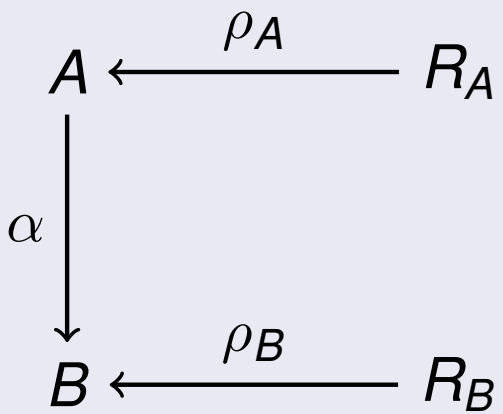
Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.
- Morphisms:



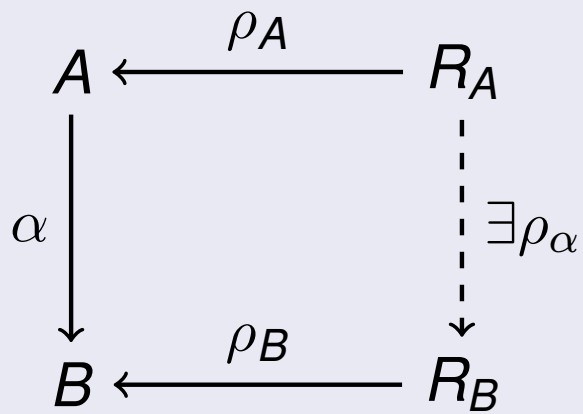
Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.
- Morphisms:



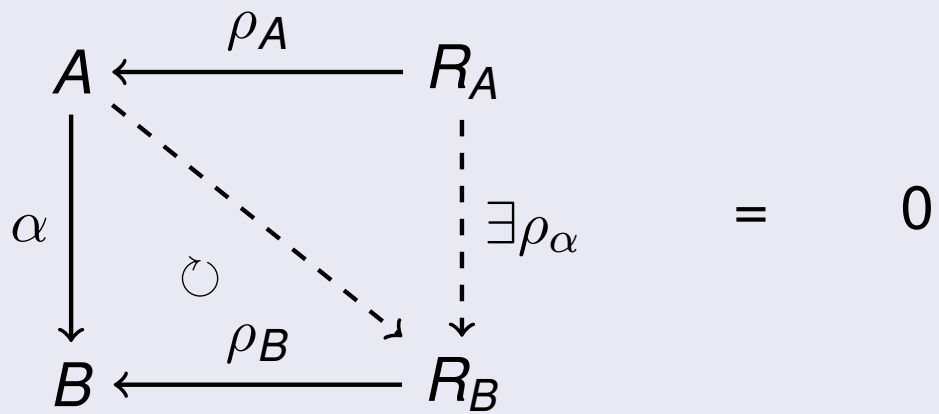
Freyd categories: Data structures

Abstraction: Instead of Rows_R, take an arbitrary additive category **P** (category with $+$, $\oplus$).

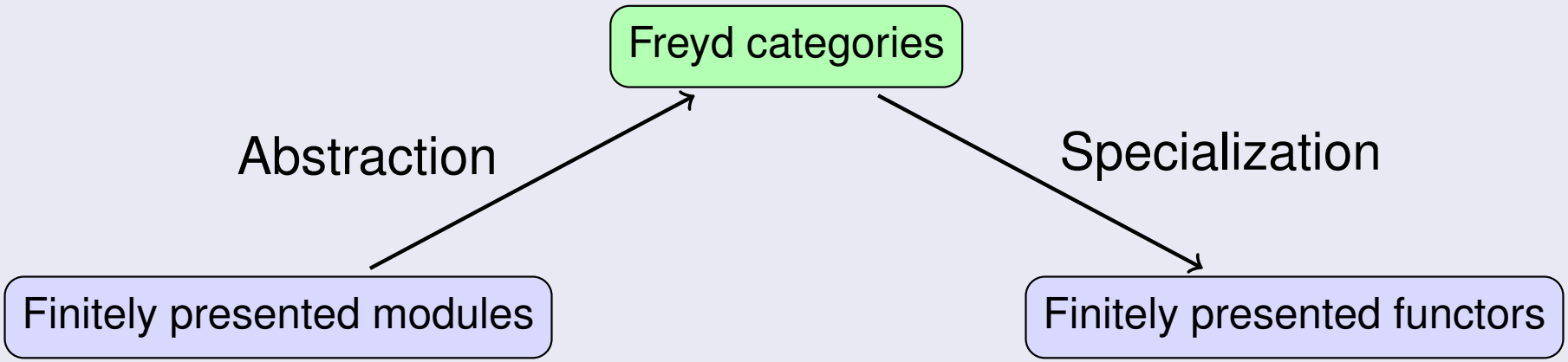
Definition

The **Freyd category** $\mathcal{A}(\mathbf{P})$ is given by the following data:

- An object in $\mathcal{A}(\mathbf{P})$ is simply a morphism $(A \xleftarrow{\rho_A} R_A)$ in **P**.
- Morphisms:

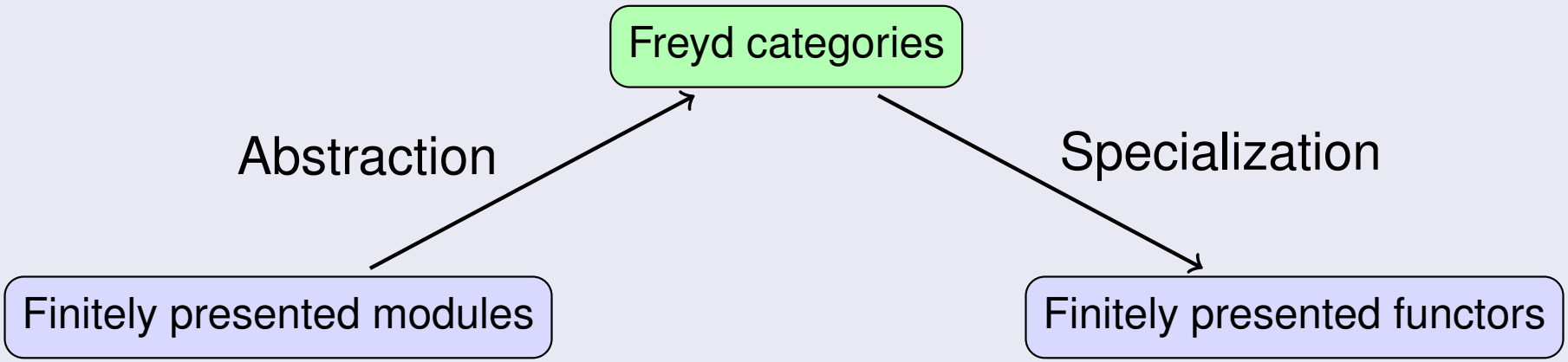


Route



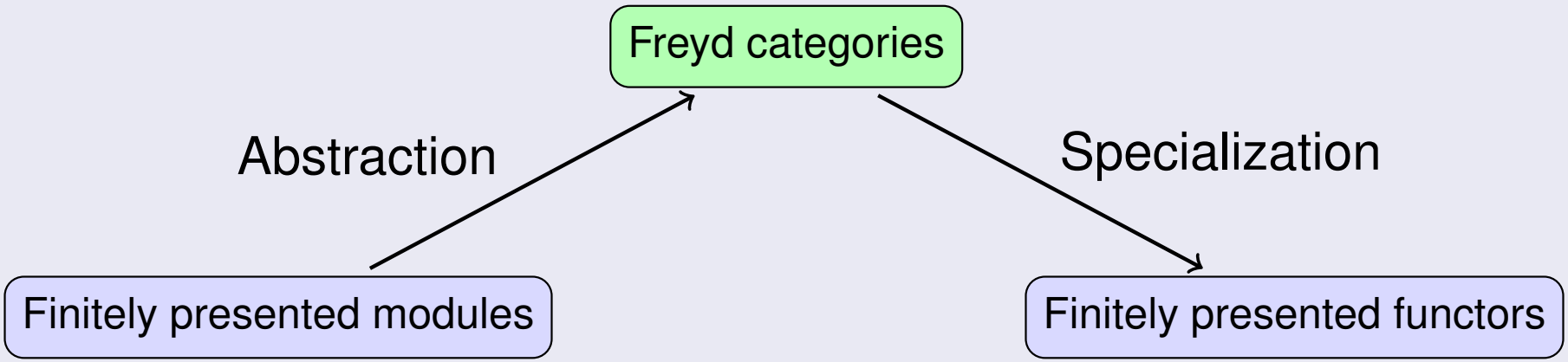
1. Data structures
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

What are syzygies in Rows_R?

Syzygies

Let $A \in R^{m \times n}$.

Syzygies

Let $A \in R^{m \times n}$.

$$R^{1 \times m} \xrightarrow{A} R^{1 \times n}$$

Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

$$R^{1 \times m} \xrightarrow{A} R^{1 \times n}$$

Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

... an object $R^{1 \times o}$,

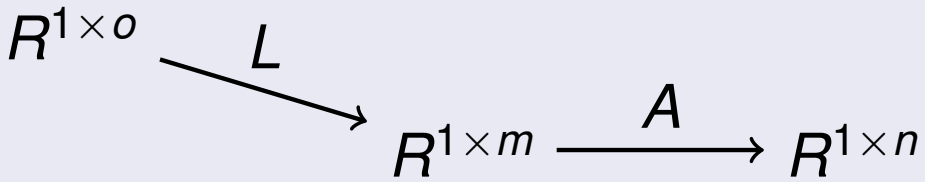
$$R^{1 \times o}$$

$$R^{1 \times m} \xrightarrow{A} R^{1 \times n}$$

Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

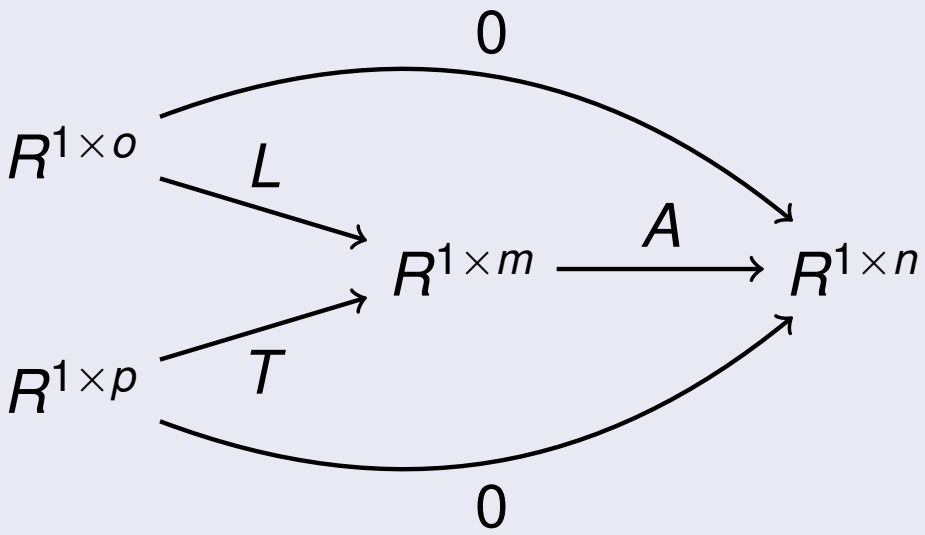
... an object $R^{1 \times o}$,
a morphism L,



Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

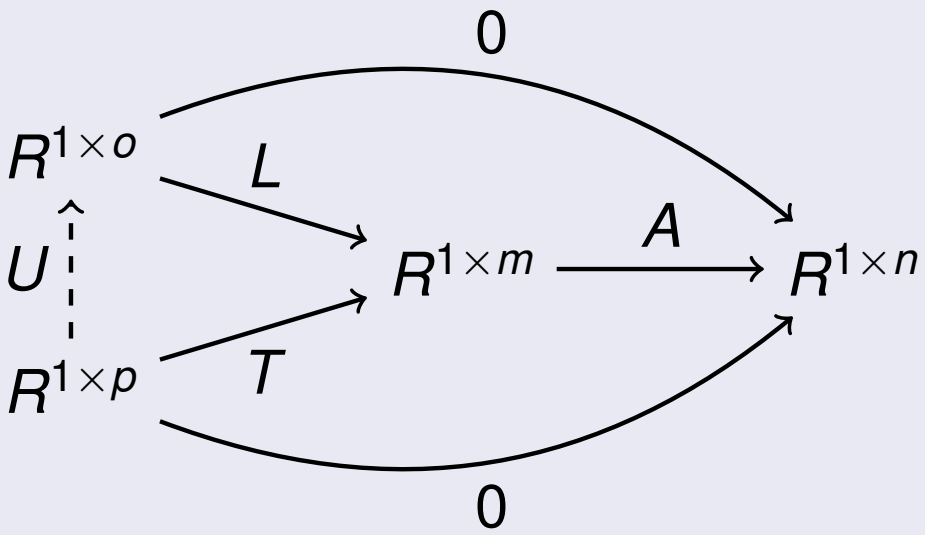
... an object $R^{1 \times o}$,
a morphism L,
and for every test morphism T



Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

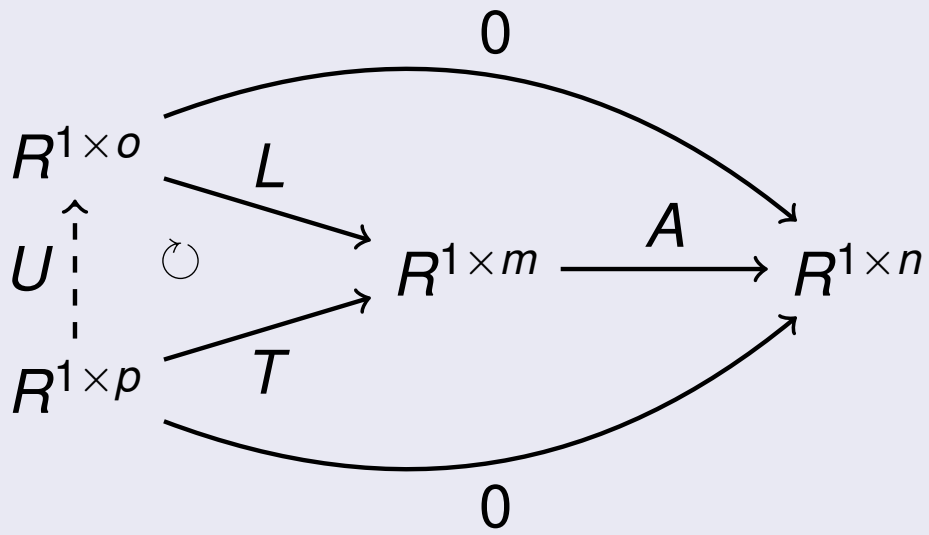
... an object $R^{1 \times o}$,
a morphism L,
and for every test morphism T
a morphism U



Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

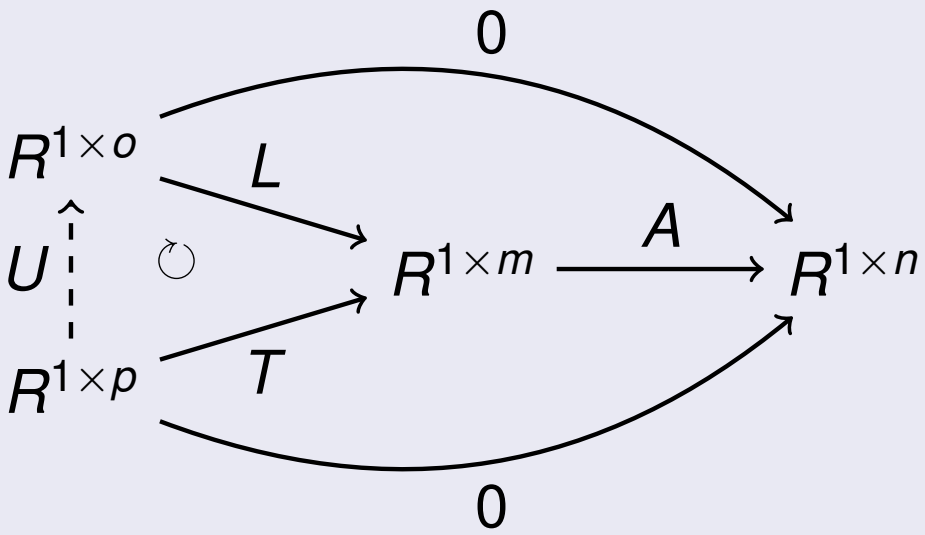
... an object $R^{1 \times o}$,
a morphism L,
and for every test morphism T
a morphism U, such that



Syzygies

Let $A \in R^{m \times n}$. Syzygies of A in the category Rows_R are given by ...

... an object $R^{1 \times o}$,
a morphism L,
and for every test morphism T
a morphism U, such that



Syzygies = Weak kernels

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **syzygies**: $L \cdot A = 0$.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **weak kernel** of A.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **weak kernel** of A.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Solution

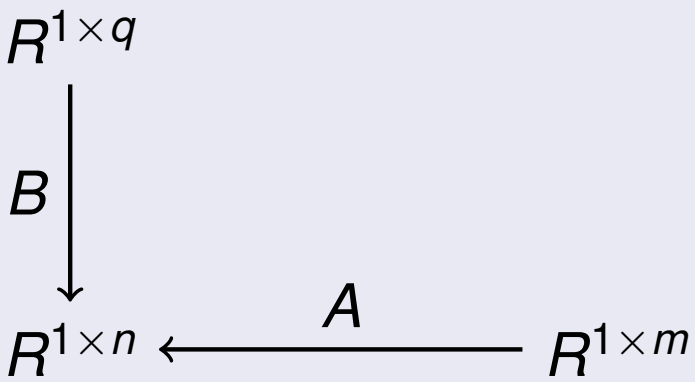
Solution

Given $A \in R^{m \times n}$

$$R^{1 \times n} \xleftarrow{A} R^{1 \times m}$$

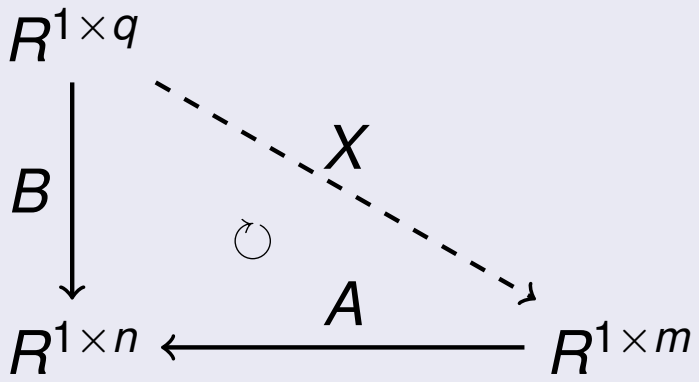
Solution

Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$.



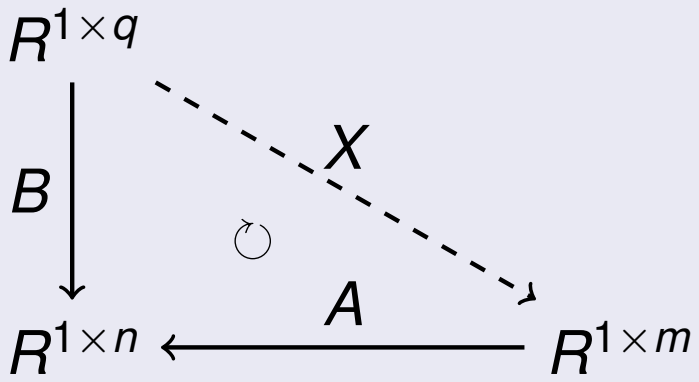
Solution

Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Find solution: $X \cdot A = B$.



Solution

Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Find solution: $X \cdot A = B$.



Solution = Lift of B along A.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **weak kernel** of A.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **solution**: $X \cdot A = B$.

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **weak kernel** of A.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **lift of B along A.**

Freyd categories: Algorithms

Theorem (Barakat, Lange-Hegermann)

R-fpmod is computable as an abelian category if we have algorithms for the following two problems:

- Given $A \in R^{m \times n}$. Find **weak kernel** of A.
- Given $A \in R^{m \times n}$ and $B \in R^{q \times n}$. Decide existence and in the affirmative case find **lift of B along A.**

This theorem generalizes to Freyd categories.

Freyd categories: Algorithms

Theorem

Let **P** be a computable additive $(+, \oplus)$ category. The Freyd category $\mathcal{A}(\mathbf{P})$ is computable as an abelian category if and only if we have algorithms for the following two problems:

- Given $\alpha \in \mathrm{Mor}(\mathbf{P})$. Find **weak kernel** of α.
- Given $M \xrightarrow{\alpha} N$ and $Q \xrightarrow{\beta} N$ in $\mathrm{Mor}(\mathbf{P})$. Decide existence and in the affirmative case find **lift** of β along α.

Freyd categories: Algorithms

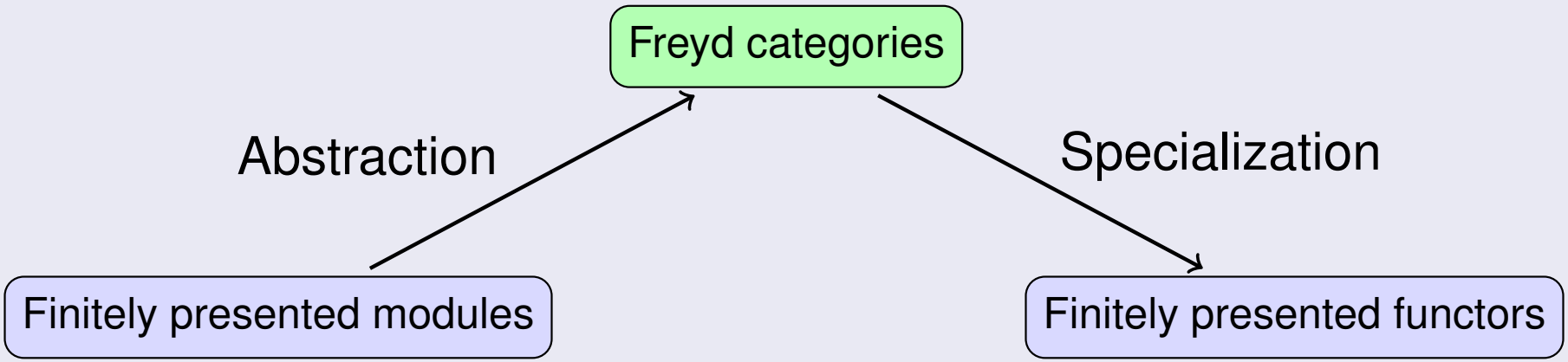
Theorem

Let **P** be a computable additive $(+, \oplus)$ category. The Freyd category $\mathcal{A}(\mathbf{P})$ is computable as an abelian category if and only if we have algorithms for the following two problems:

- Given $\alpha \in \mathrm{Mor}(\mathbf{P})$. Find **weak kernel** of α.
- Given $M \xrightarrow{\alpha} N$ and $Q \xrightarrow{\beta} N$ in $\mathrm{Mor}(\mathbf{P})$. Decide existence and in the affirmative case find **lift** of β along α.

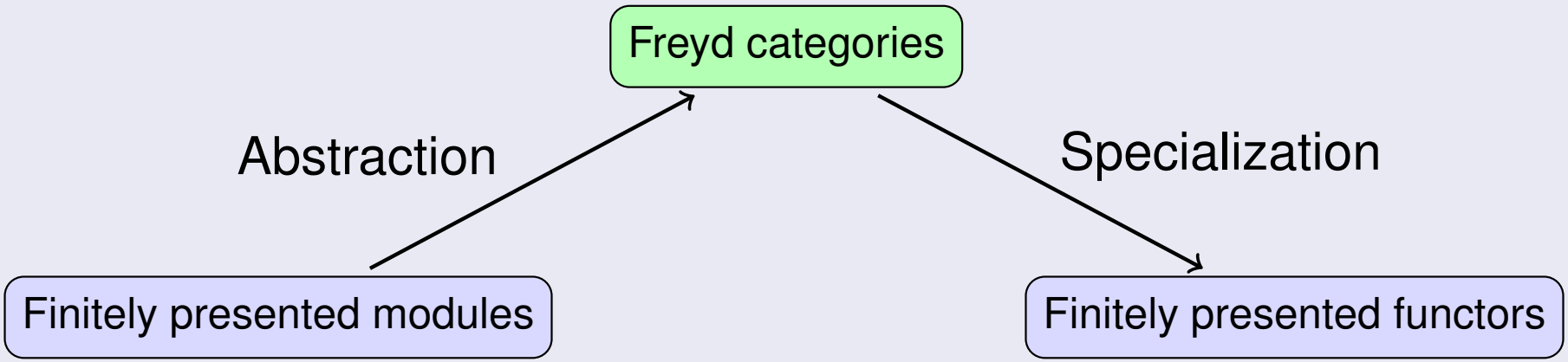
This is a constructive version of a theorem by Peter Freyd.

Route



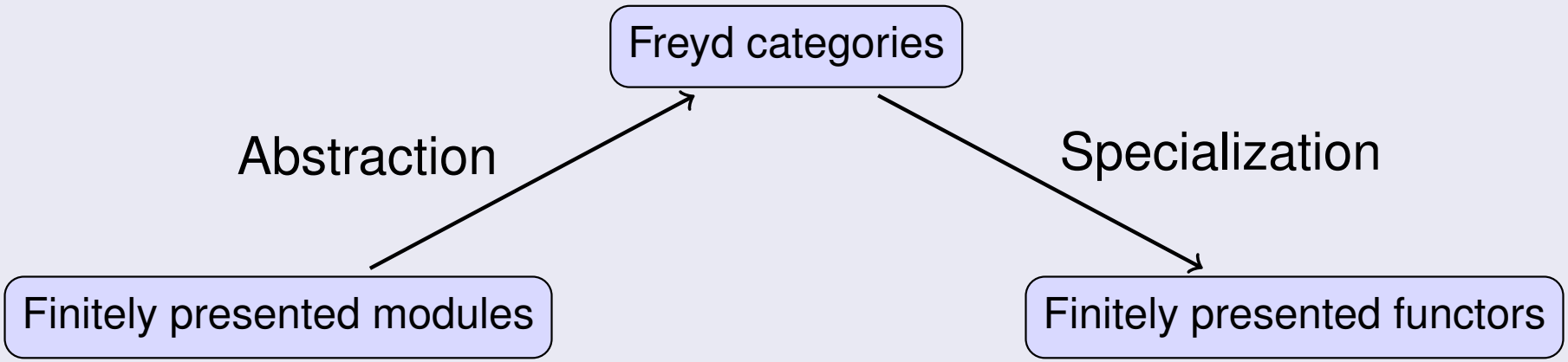
1. Data structures ✓
2. Algorithms

Route



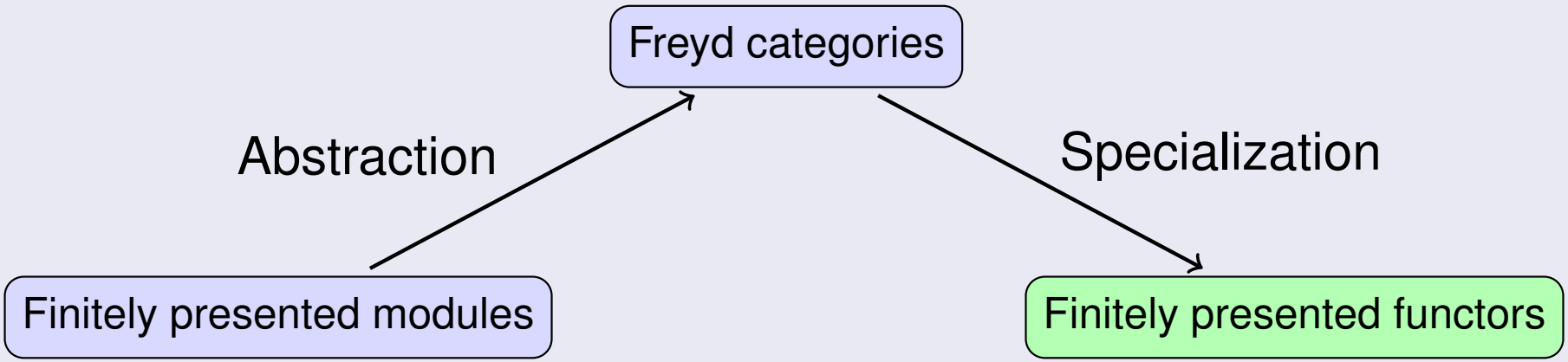
1. Data structures ✓
2. Algorithms ✓

Route



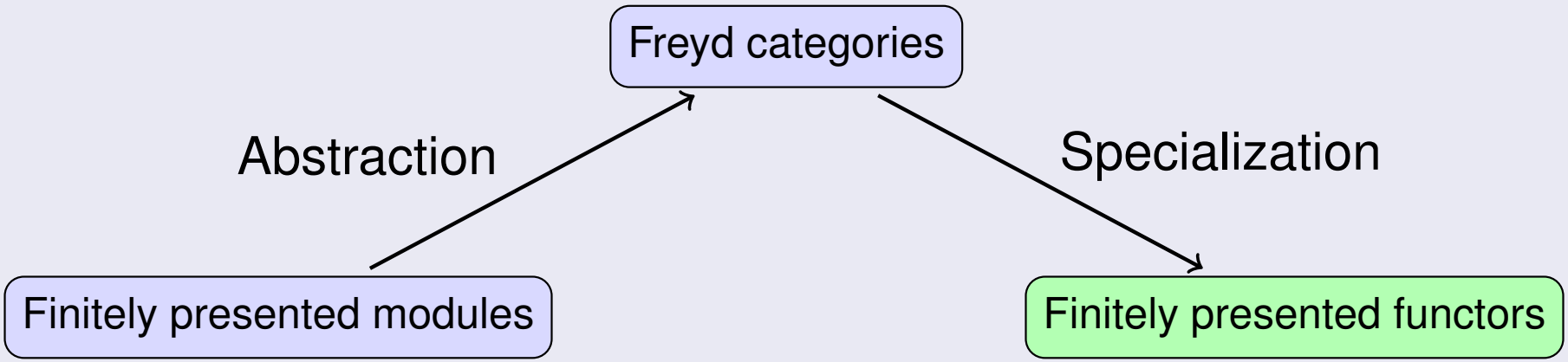
1. Data structures ✓
2. Algorithms ✓

Route



1. Data structures
2. Algorithms

Route



1. Data structures
2. Algorithms

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq$

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq$

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq R\text{-fpgrmod}$ (for a graded ring R)

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq R\text{-fpgrmod}$ (for a graded ring R)
- $\mathcal{A}(R\text{-fpmod}) \simeq$

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq R\text{-fpgrmod}$ (for a graded ring R)
- $\mathcal{A}(R\text{-fpmod}) \simeq$ f.p. **contra**variant functors

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq R\text{-fpgrmod}$ (for a graded ring R)
- $\mathcal{A}(R\text{-fpmod}) \simeq$ f.p. **contra**variant functors
- $\mathcal{A}(R\text{-fpmod}^{\mathrm{op}}) \simeq$

Finitely presented functors: Data structures

Plug additive categories **P** into $\mathcal{A}(-)$:

- $\mathcal{A}(\mathrm{Rows}_R) \simeq R\text{-fpmod}$
- $\mathcal{A}(\{\text{rows with a grading}\}) \simeq R\text{-fpgrmod}$ (for a graded ring R)
- $\mathcal{A}(R\text{-fpmod}) \simeq$ f.p. **contra**variant functors
- $\mathcal{A}(R\text{-fpmod}^{\mathrm{op}}) \simeq$ f.p. **co**variant functors

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -)$$

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \xleftarrow{\nu} \mathrm{Hom}(M, -)$$

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \xleftarrow{\nu} \mathrm{Hom}(M, -)$$

$$F \cong \mathrm{coker}(\nu)$$

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \xleftarrow{\nu} \mathrm{Hom}(M, -)$$

$$F \cong \mathrm{coker}(\nu)$$

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \xleftarrow{\nu} \mathrm{Hom}(M, -)$$

$$F \cong \mathrm{coker}(\nu)$$

$$\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$$

Finitely presented functors: Data structures

Definition

A **co**variant functor $F : R\text{-fpmod} \to \mathbf{Ab}$ is called **finitely presented** if there exist $M, N \in R\text{-fpmod}$ and an exact sequence of functors

$$0 \longleftarrow F \longleftarrow \mathrm{Hom}(N, -) \xleftarrow{\nu} \mathrm{Hom}(M, -)$$

$$F \cong \mathrm{coker}(\nu)$$

$$\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$$

$$\mathcal{A}(\mathbf{P}) \simeq \text{f.p. covariant functors}$$

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

$$N \xrightarrow{\alpha} M$$

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

$$N \xrightarrow{\alpha} M$$

$$\downarrow$$

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

$$N \xrightarrow{\alpha} M$$

$$\downarrow$$

$$\mathrm{Hom}(N, -) \xleftarrow{\mathrm{Hom}(\alpha, -)} \mathrm{Hom}(M, -)$$

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

$$N \xrightarrow{\alpha} M$$

$$\updownarrow \quad \text{(1:1) Yoneda Lemma}$$

$$\mathrm{Hom}(N, -) \xleftarrow{\mathrm{Hom}(\alpha, -)} \mathrm{Hom}(M, -)$$

Finitely presented functors: Data structures

How to describe $\mathbf{P} = \{\mathrm{Hom}(N, -) \longleftarrow \mathrm{Hom}(M, -) \mid M, N \in R\text{-fpmod}\}$?

$$N \xrightarrow{\alpha} M$$

$$\updownarrow \quad \text{(1:1) Yoneda Lemma}$$

$$\mathrm{Hom}(N, -) \xleftarrow{\mathrm{Hom}(\alpha, -)} \mathrm{Hom}(M, -)$$

Thus, $\mathbf{P} \simeq R\text{-fpmod}^{\mathrm{op}}$.

Finitely presented functors: Data structures

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$:

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$:

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$: Colifts in R-fpmod

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$: Colifts in R-fpmod (possible for example if R is commutative computable)

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$: Colifts in R-fpmod (possible for example if R is commutative computable) ✓

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$: Colifts in R-fpmod (possible for example if R is commutative computable) ✓

Corollary

The abelian category of finitely presented functors has $\mathcal{A}(R\text{-}\mathrm{fpmod}^{\mathrm{op}})$ as a computable model.

Finitely presented functors: Data structures

- Weak kernels in R-$\mathrm{fpmod}^{\mathrm{op}}$: Cokernels in R-fpmod ✓
- Lifts in R-$\mathrm{fpmod}^{\mathrm{op}}$: Colifts in R-fpmod (possible for example if R is commutative computable) ✓

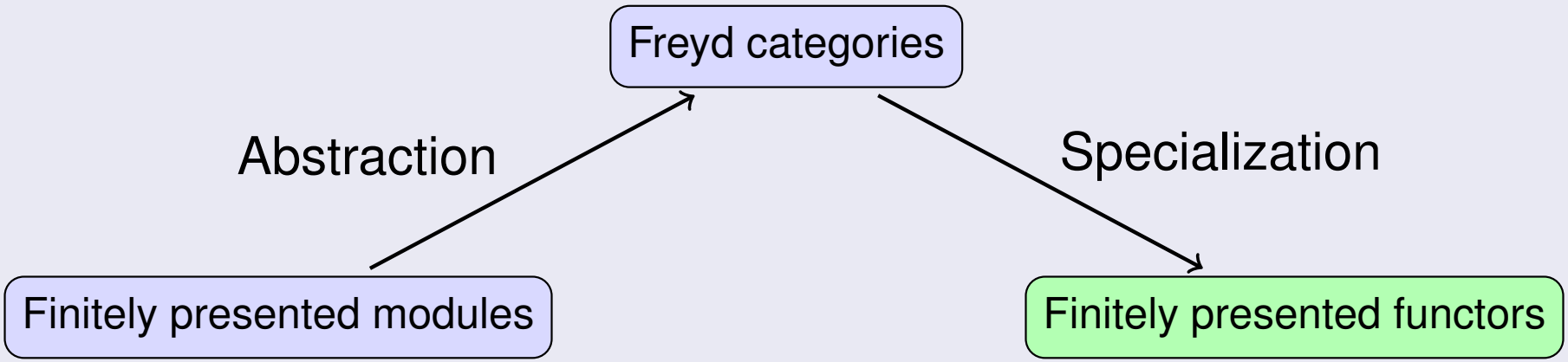
Corollary

The abelian category of finitely presented functors has $\mathcal{A}(R\text{-}\mathrm{fpmod}^{\mathrm{op}})$ as a computable model.

Remark

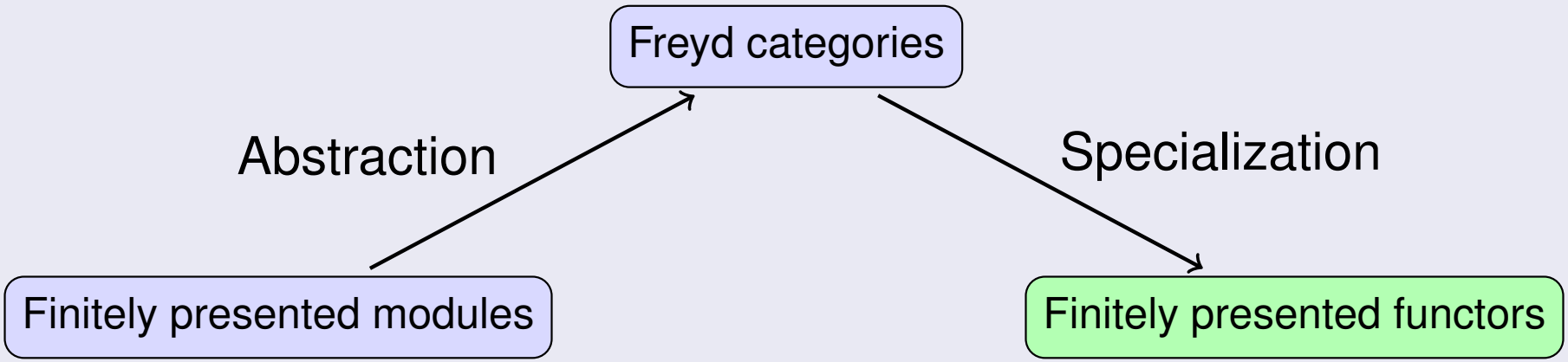
$$\mathcal{A}(R\text{-}\mathrm{fpmod}^{\mathrm{op}}) \simeq \mathcal{A}(\mathcal{A}(\mathrm{Rows}_R)^{\mathrm{op}})$$

Route



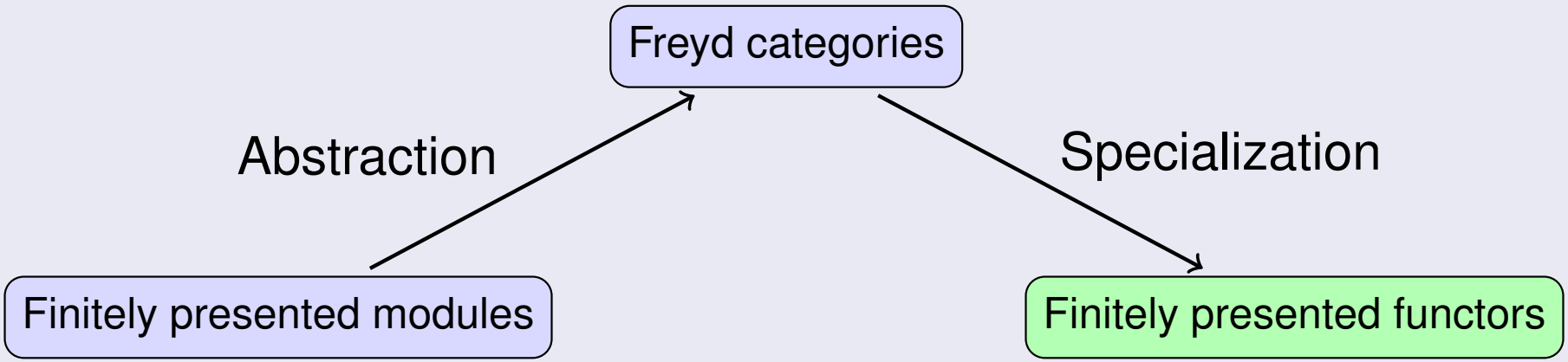
1. Data structures
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Route



1. Data structures ✓
2. Algorithms

Finitely presented functors: Algorithms

Finitely presented functors: Algorithms

- Operations in abelian categories

Finitely presented functors: Algorithms

- Operations in abelian categories
- Computing sets of natural transformations, e.g., Hom $\left(\mathrm{Tor}_j(M,-), \mathrm{Ext}^i(A,-)\right)$

Finitely presented functors: Algorithms

- Operations in abelian categories
- Computing sets of natural transformations, e.g., Hom $\left(\operatorname{Tor}_j(M,-), \operatorname{Ext}^i(A,-)\right)$
- Computing injective resolutions $(\operatorname{gldim} \leq 2)$

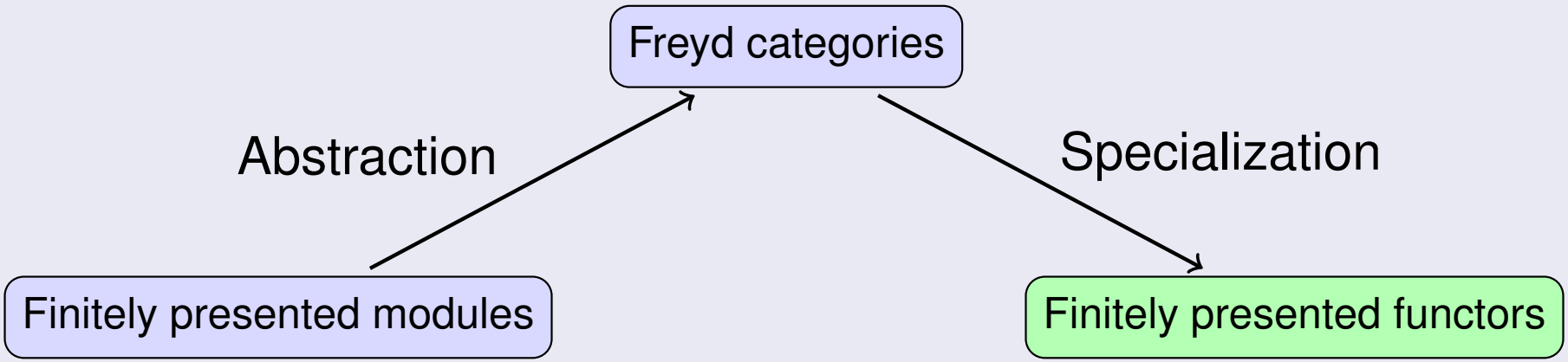
Finitely presented functors: Algorithms

- Operations in abelian categories
- Computing sets of natural transformations, e.g., Hom $\left(\mathrm{Tor}_j(M,-), \mathrm{Ext}^i(A,-)\right)$
- Computing injective resolutions ($\mathrm{gldim} \leq 2$)
- Deciding left exactness (representability)

Finitely presented functors: Algorithms

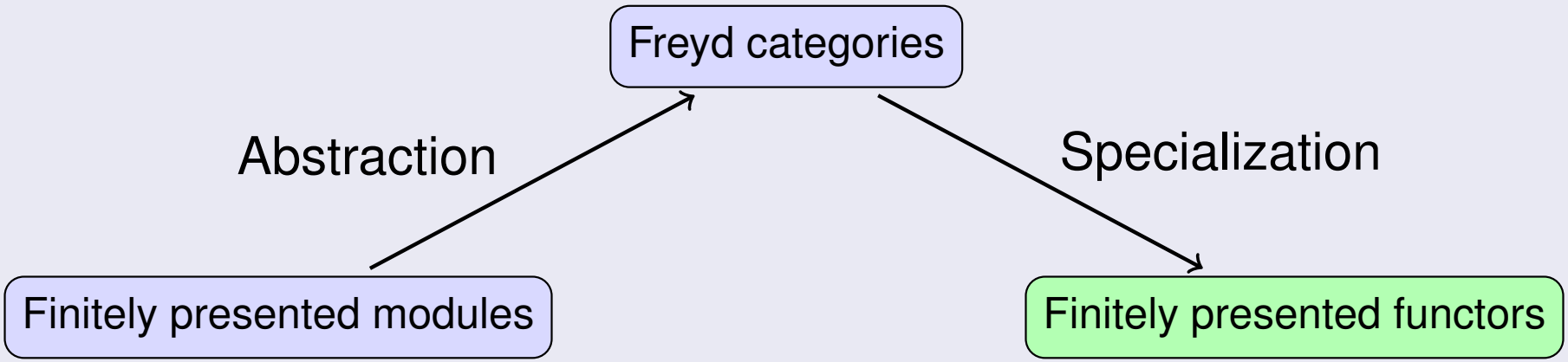
- Operations in abelian categories
- Computing sets of natural transformations, e.g., Hom $\left(\mathrm{Tor}_j(M,-), \mathrm{Ext}^i(A,-)\right)$
- Computing injective resolutions ($\mathrm{gldim} \leq 2$)
- Deciding left exactness (representability)
- Deciding right exactness

Route



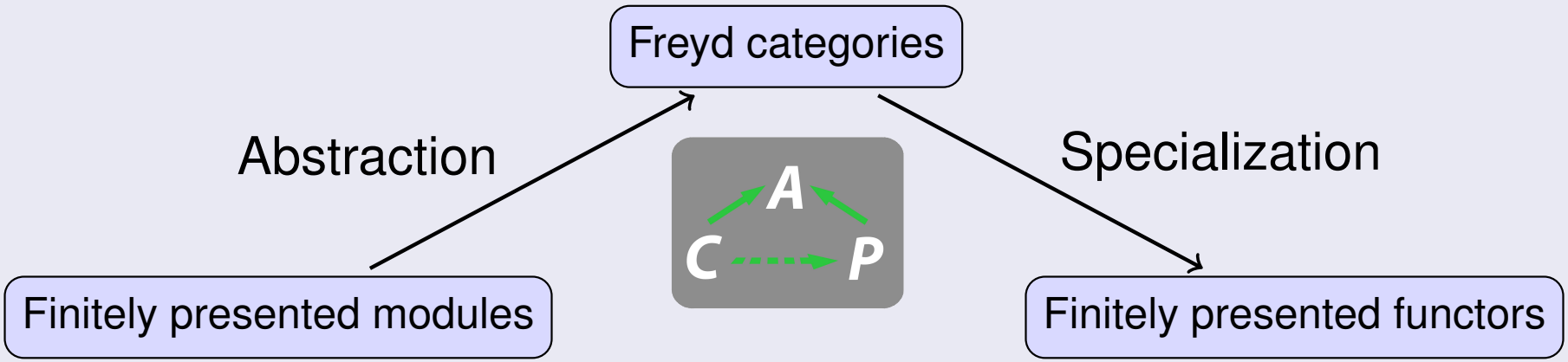
1. Data structures ✓
2. Algorithms

Route



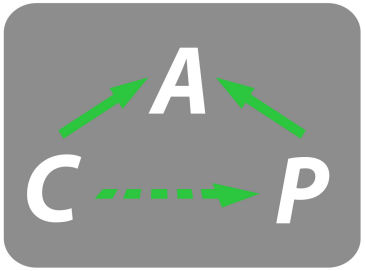
1. Data structures ✓
2. Algorithms ✓

Route

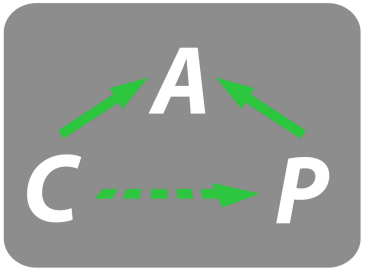


1. Data structures ✓
2. Algorithms ✓

The CAP-project



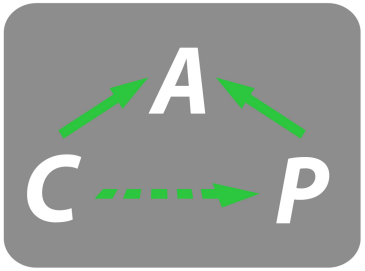
The CAP-project



CAP: categories, algorithms, programming

category constructor	application

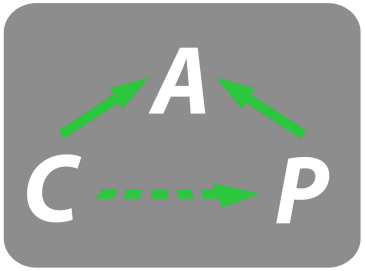
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	

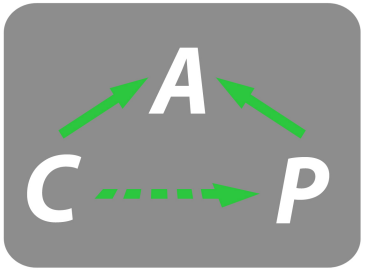
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties

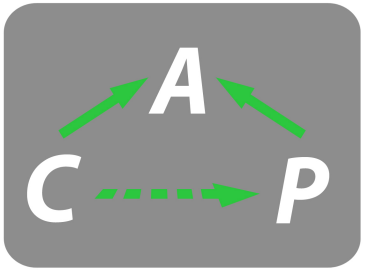
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	

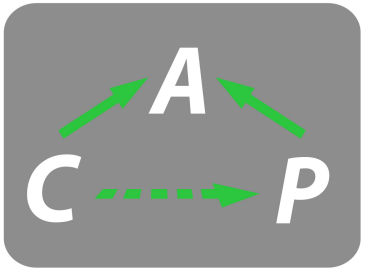
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories

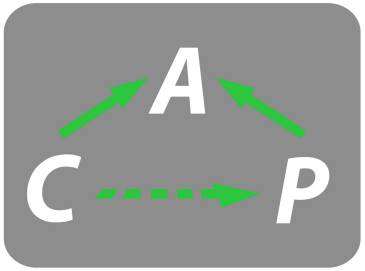
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	

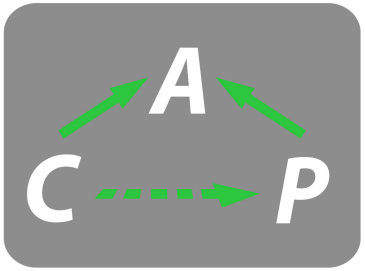
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing

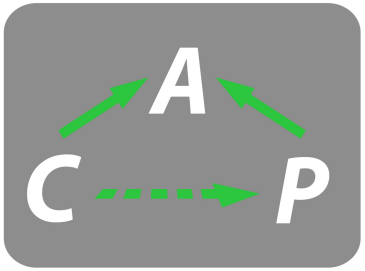
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing
$(\mathbf{A}, \mathbf{B}) \mapsto \mathrm{Hom}(\mathbf{A}, \mathbf{B})$	

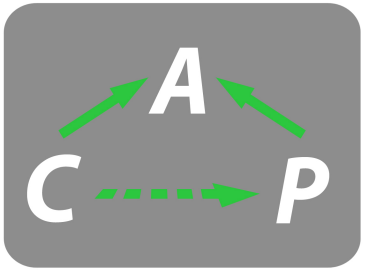
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing
$(\mathbf{A}, \mathbf{B}) \mapsto \mathrm{Hom}(\mathbf{A}, \mathbf{B})$	representation theory of f.d. algebras

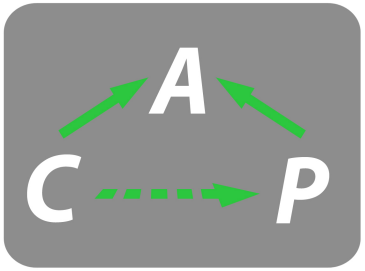
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing
$(\mathbf{A}, \mathbf{B}) \mapsto \mathrm{Hom}(\mathbf{A}, \mathbf{B})$	representation theory of f.d. algebras
$\mathbf{P} \mapsto \mathcal{A}(\mathbf{P})$	

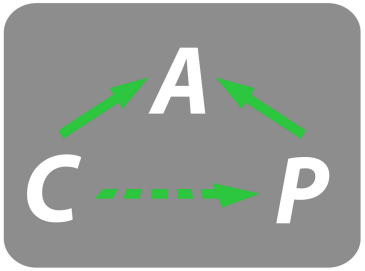
The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing
$(\mathbf{A}, \mathbf{B}) \mapsto \mathrm{Hom}(\mathbf{A}, \mathbf{B})$	representation theory of f.d. algebras
$\mathbf{P} \mapsto \mathcal{A}(\mathbf{P})$	f.p. functors

The CAP-project



CAP: categories, algorithms, programming

category constructor	application
$(\mathbf{A}, \mathbf{C}) \mapsto \mathbf{A}/\mathbf{C}$	coherent sheaves on toric varieties
$\mathbf{A} \mapsto \mathrm{Ch}^\bullet(\mathbf{A})$	towards dg categories
$\mathbf{A} \mapsto \mathbf{G}(\mathbf{A})$	diagram chasing
$(\mathbf{A}, \mathbf{B}) \mapsto \mathrm{Hom}(\mathbf{A}, \mathbf{B})$	representation theory of f.d. algebras
$\mathbf{P} \mapsto \mathcal{A}(\mathbf{P})$	f.p. functors

References

- Maurice Auslander, *Coherent functors*, Proc. Conf. Categorical Algebra (La Jolla, Calif., 1965), Springer, New York, 1966, pp. 189–231. MR MR0212070 (35 #2945)
- Mohamed Barakat and Markus Lange-Hegermann, *An axiomatic setup for algorithmic homological algebra and an alternative approach to localization*, J. Algebra Appl. **10** (2011), no. 2, 269–293, (`arXiv:1003.1943`). MR 2795737 (2012f:18022)
- Peter Freyd, *Representations in abelian categories*, Proc. Conf. Categorical Algebra (La Jolla, Calif., 1965), Springer, New York, 1966, pp. 95–120. MR 0209333
- Sebastian Posur, *A constructive approach to Freyd categories*, (in preparation) 2017.