# CCHM Part I

## Week 8, Spring 2023

Chenchao Ding, Mar 6

# Core language

## syntax of terms and types

```
t,u,A,B := x | λx : A. t | t u | (x : A) → B          Π-types
         | (t, u) | t.1 | t.2 | (x : A) × B             Σ-types
         | 0 | s u | natrec t u | ℕ                     Natural numbers
         | <i> t | t r | Path A t u                     Path types
         | [φ₁ t₁, φ₂ t₂, ..., φₙ tₙ]                   Systems
         | compⁱ A [φ ↦ u] a₀                           Compositions
```

- path abstractions
- path applications
- systems
- compositions

# 𝕀 as a distributive lattice

## syntax of interval elements

### Syntax of 𝕀 elements

```
r, s := 0 | 1 | i | 1 − r | r ∧ s | r ∨ s
```

$1 - 0 = 1$
$1 - 1 = 0$
$1 - (r \vee s) = (1 - r) \wedge (1 - s)$
$1 - (r \wedge s) = (1 - r) \vee (1 - s)$

- $r \wedge s$ represents $\min(r, s)$
- $r \vee s$ represents $\max(r, s)$

# Path types

## dots, lines, cubes, hypercubes, …

$$
\begin{array}{rcll}
t, u, A, B & ::= & \ldots & \\
& | & \mathsf{Path}\ A\ t\ u \ \mid\ \langle i \rangle\ t \ \mid\ t\ r & \text{Path types}
\end{array}
$$

A type in a context with n *names* corresponds to an n-dimensional cube

<table>
<tr><td>$() \vdash A$</td><td>$\bullet\ A$</td></tr>
<tr><td>$i : \mathbb{I} \vdash A$</td><td>$A(i0) \xrightarrow{A} A(i1)$</td></tr>
<tr><td>$i : \mathbb{I}, j : \mathbb{I} \vdash A$</td><td>$$\begin{tikzcd} A(i0)(j1) \arrow[r, "A(j1)"] & A(i1)(j1) \\ A(i0)(j0) \arrow[u, "A(i0)"] \arrow[r, "A(j0)"'] & A(i1)(j0) \arrow[u, "A(i1)"'] \end{tikzcd}$$</td></tr>
<tr><td>$\vdots$</td><td>$\vdots$</td></tr>
</table>

# Path types

## substitution of intervals

$$\begin{tikzcd} a \arrow[r, "p"] & b \end{tikzcd}$$

$$\begin{tikzcd} b \arrow[r, "p(i/1-i)"] & a \end{tikzcd}$$

inversion

$$\begin{tikzcd} a \arrow[r, "p"] & b \\ a \arrow[u, "p(i0)"] \arrow[r, "p(i0)"'] \arrow[ur, phantom, "p(i/i \wedge j)"] & a \arrow[u, "p(i/j)"'] \end{tikzcd} \qquad \begin{tikzcd} b \arrow[r, "p(i1)"] & b \\ a \arrow[u, "p(i/j)"] \arrow[r, "p"'] \arrow[ur, phantom, "p(i/i \vee j)"] & b \arrow[u, "p(i1)"'] \end{tikzcd} \qquad \begin{tikzcd} j \\ {} \arrow[u] \arrow[r] & i \end{tikzcd}$$

connections

$l:\ p(i/i\wedge j)(i0) = p(i/0\wedge j) = p(i/0) = p(i0)$

$r:\ p(i/i\wedge j)(i1) = p(i/1\wedge j) = p(i/j) \stackrel{?}{=} p$

$b:\ p(i/i\wedge j)(j0) = p(i/i\wedge 0) = p(i/0) = p(i0)$

$t:\ p(i/i\wedge j)(j1) = p(i/i\wedge 1) = p(i/i) \doteq p$

$diag:\ p(i/i\wedge j)(j/i) = p(i/i\wedge i) = p(i/i) \doteq p$

# Context restriction

## syntax of contexts

### Syntax of contexts

```
Γ, Δ := ()
      | Γ, x : A
      | Γ, i : ▯
      | Γ, φ               Restrictions
```

context restriction allow us to describe new geometrical shapes corresponding to “sub-polyhedra” of a cube

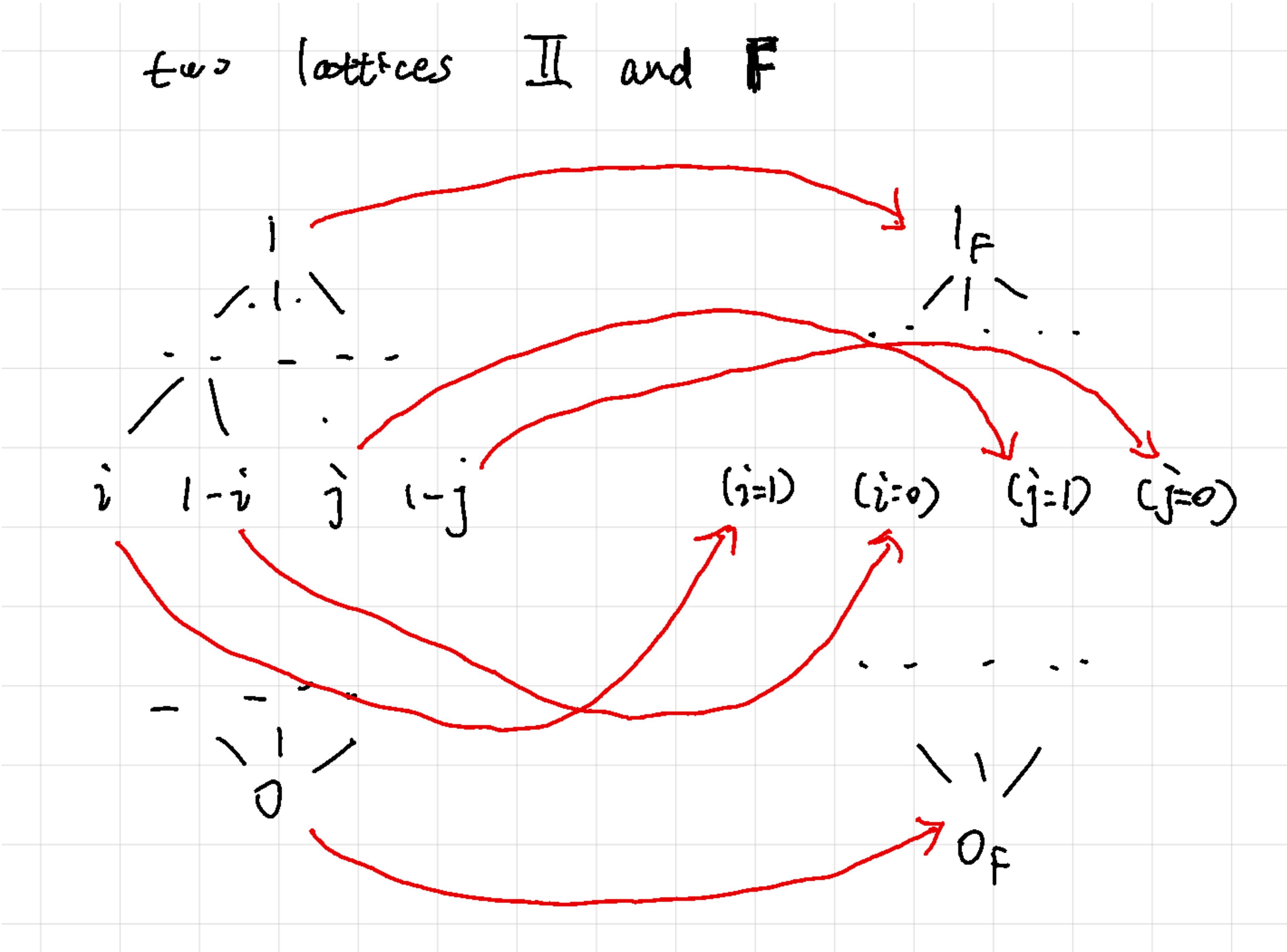
# Face Lattice $\mathbb{F}$

## face formula and "sub-polyhedra"

### Syntax of face formula

```
φ, ψ := 0𝔽 | 1𝔽
      | (i = 0)
      | (i = 1)
      | φ ∧ ψ
      | φ ∨ ψ
```

# A lattice map $\mathbb{I} \to \mathbb{F}$



# Context restriction: examples

## compatible union of faces

$$i : \mathbb{I}, (i = 0) \vee (i = 1) \vdash A$$

$$A(i0) \bullet \qquad A(i1) \bullet$$

$$i : \mathbb{I}, j : \mathbb{I}, (i = 0) \vee (j = 1) \vdash A$$

$$\begin{tikzcd}
A(i0)(j1) \arrow[r, "A(j1)"] & A(i1)(j1) \\
A(i0)(j0) \arrow[u, "A(i0)"] &
\end{tikzcd}$$

$$i : \mathbb{I}, j : \mathbb{I}, (i = 0) \vee (i = 1) \vee (j = 0) \vdash A$$

$$\begin{tikzcd}
A(i0)(j1) & A(i1)(j1) \\
A(i0)(j0) \arrow[u, "A(i0)"] \arrow[r, "A(j0)"'] & A(i1)(j0) \arrow[u, "A(i1)"']
\end{tikzcd}$$

# Partial elements

## extensibility

$\Gamma \vdash a : A[\varphi \mapsto u]$

- $\Gamma \vdash a : A$
- $\Gamma, \varphi \vdash a = u : A$

It can be read as "in the restricted context $\varphi$, $a$ agrees with $u$".
In other words, $a$ is a evidence that $u$ (defined on $\varphi$) is *extensible*.

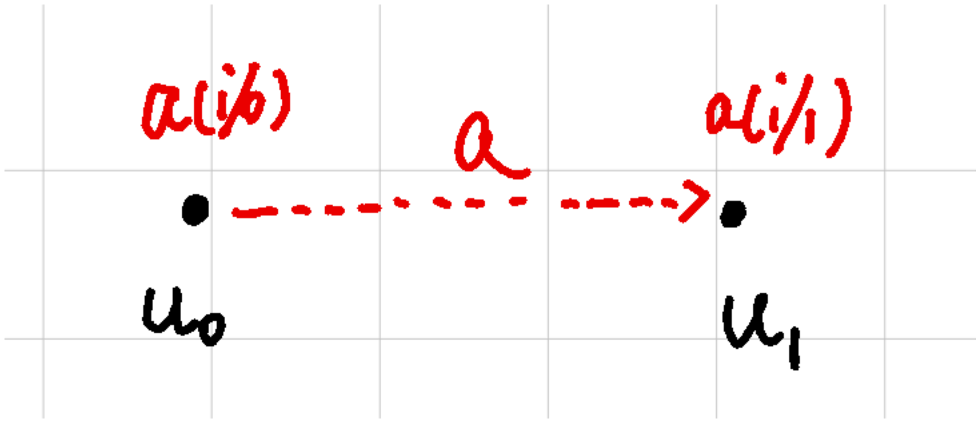
# Partial elements

## extensibility

It can be read as "in the restricted context `φ`, `a` agrees with `u`".
In other words, `a` is a evidence that `u` (defined on `φ`) is *extensible*.

```
Γ, i : I ⊢ a : A[(i = i0) ↦ u0 ; (i = i1) ↦ u1]
```

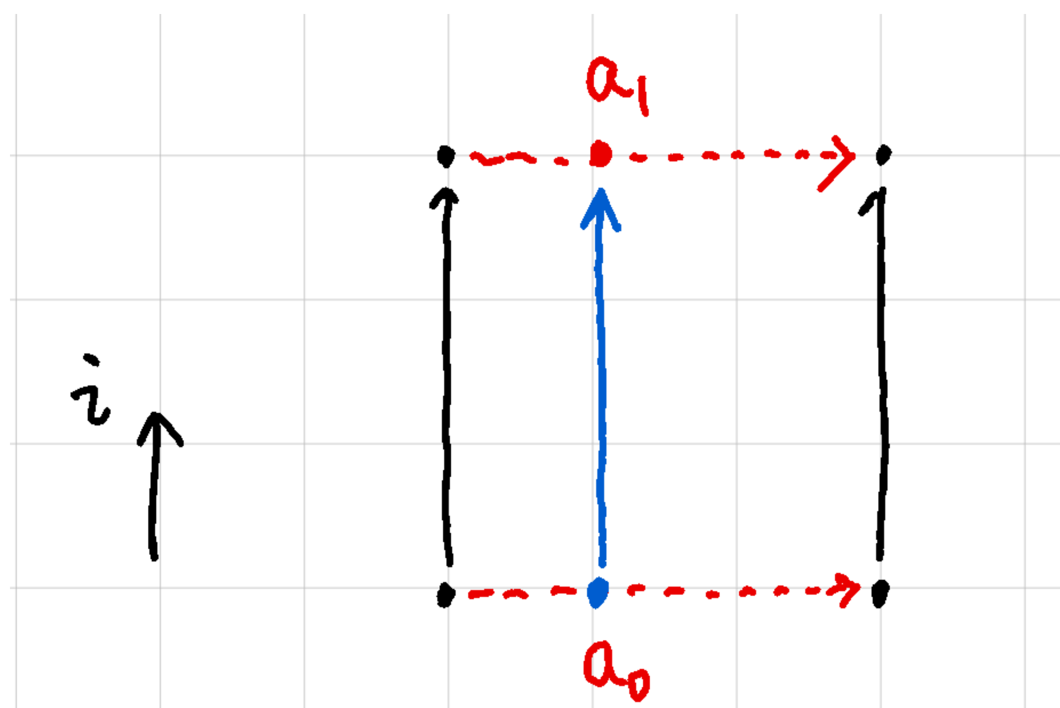
- `Γ, i : I ⊢ A`
- `Γ, i : I ⊢ a : A`
- `Γ, i : I, (i = i0) ⊢ a(i0) = u0 : A(i0)`
- `Γ, i : I, (i = i1) ⊢ a(i1) = u1 : A(i1)`



# Composition operation

## extensibility is preserved along paths

```
Γ ⊢ φ : 𝔽
Γ, (i : 𝕀) ⊢ A
Γ, φ, (i : 𝕀) ⊢ u : A
Γ ⊢ a₀ : A(i0) [φ ↦ u(i0)]
------------------------------------------
Γ ⊢ compⁱ A [φ ↦ u] a₀ : A(i1) [φ ↦ u(i1)]
```



# Composition operation

## extensibility is preserved along paths

```
Γ ⊢ φ : 𝔽
Γ, (i : 𝕀) ⊢ A
Γ, φ, (i : 𝕀) ⊢ u : A
Γ ⊢ a₀ : A(i0) [φ ↦ u(i0)]
------------------------------------
Γ ⊢ compⁱ A [φ ↦ u] a₀ : A(i1) [φ ↦ u(i1)]
```

- `u` is called a “partial path”
- `u(i0)` and `u(i1)` are partial elements

# Composition operation

## extensibility is preserved along paths

```
Γ ⊢ φ : 𝔽
Γ, (i : 𝕀) ⊢ A
Γ, φ, (i : 𝕀) ⊢ u : A
Γ ⊢ a₀ : A(i0) [φ ↦ u(i0)]
------------------------------------------
Γ ⊢ compⁱ A [φ ↦ u] a₀ : A(i1) [φ ↦ u(i1)]
```

```
postulate
  comp' : ∀ {ℓ}
          → (A : ∀ i → Type ℓ)
          → (φ : I)
          → (u : ∀ i → Partial φ (A i))
          → A i0 [ φ ↦ u i0 ]
          -------------------------
          → A i1 [ φ ↦ u i1 ]
```

# Transport

## a special case of composition

### Two special cases

1. When `φ = 1𝔽`, `u(i1)` becomes a "total element" (no context restrictions):

```
Γ ⊢ compⁱ A [1𝔽 ↦ u] a₀ = u(i1) : A(i1)
```

2. When `φ = 0𝔽`, composition corresponds to transport:

```
Γ ⊢ transpⁱ A a = compⁱ A [] a : A(i1)
```

# Kan filling operation

## defined with composition

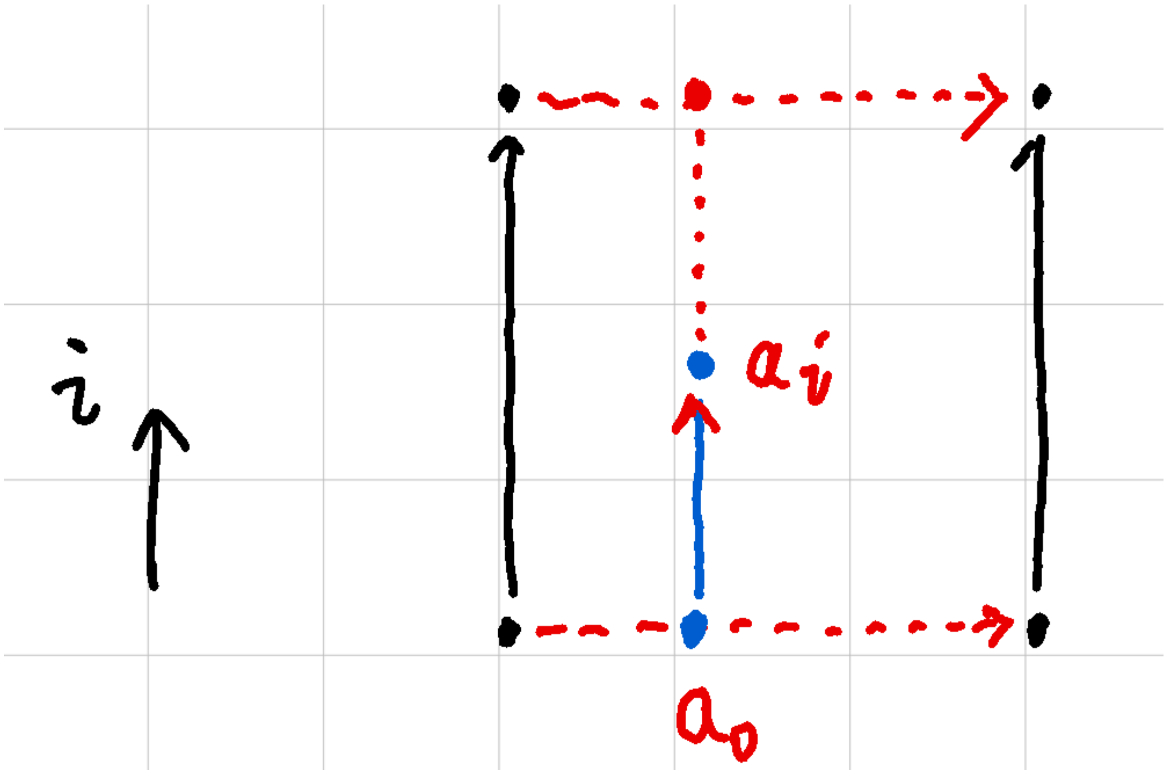
$$\Gamma, i : \mathbb{I} \vdash \mathsf{fill}^i\ A\ [\varphi \mapsto u]\ a_0 = \mathsf{comp}^j\ A(i/i \wedge j)\ [\varphi \mapsto u(i/i \wedge j), (i = 0) \mapsto a_0]\ a_0 : A$$

1. when `i=0`, it's just identity function.

```
Γ ⊢ v(i0) = a0 : A
```

2. when `i=1`, it's the "full composition".

```
Γ ⊢ v(i1) = comp^i A [φ ↦ u] a0 : A(i1)
```



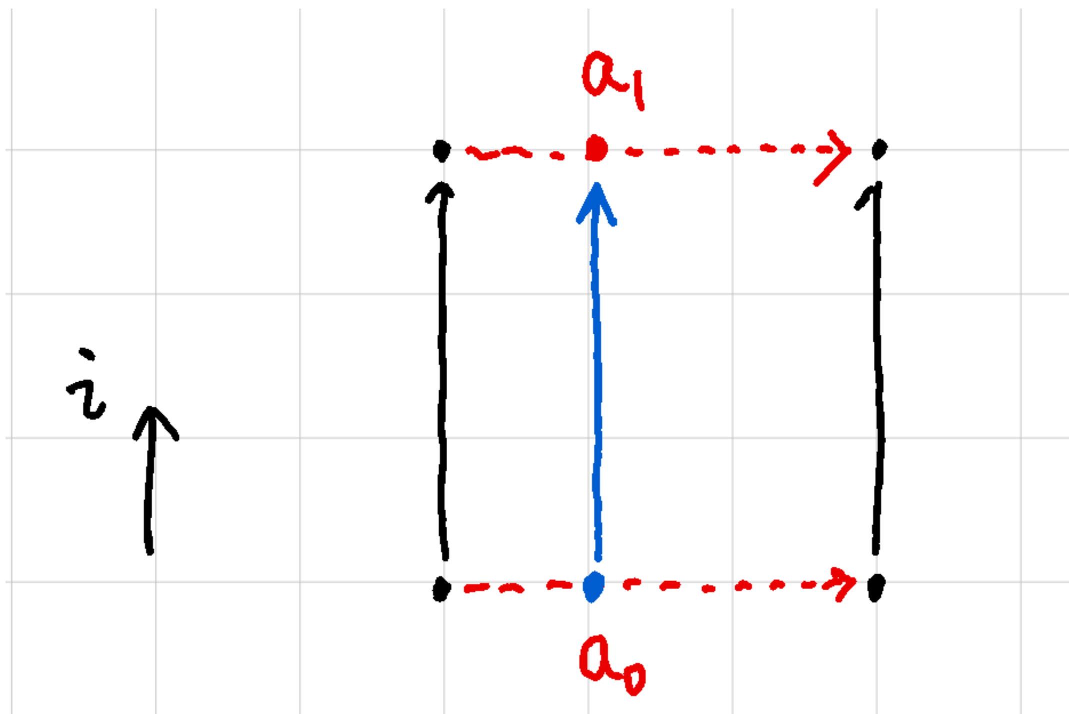
# Kan filling operation

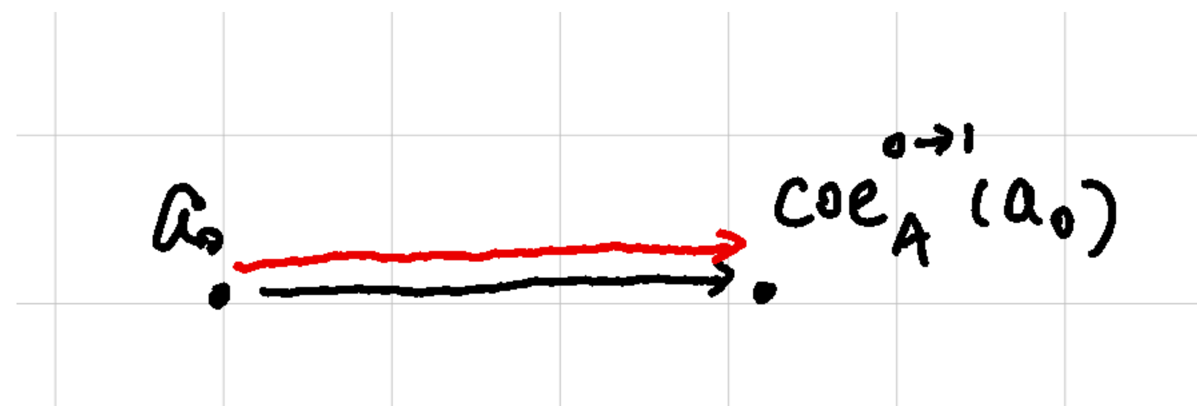
## defined with composition

$$\Gamma, i : \mathbb{I} \vdash \mathsf{fill}^i\ A\ [\varphi \mapsto u]\ a_0 = \mathsf{comp}^j\ A(i/i \wedge j)\ [\varphi \mapsto u(i/i \wedge j), (i = 0) \mapsto a_0]\ a_0 : A$$

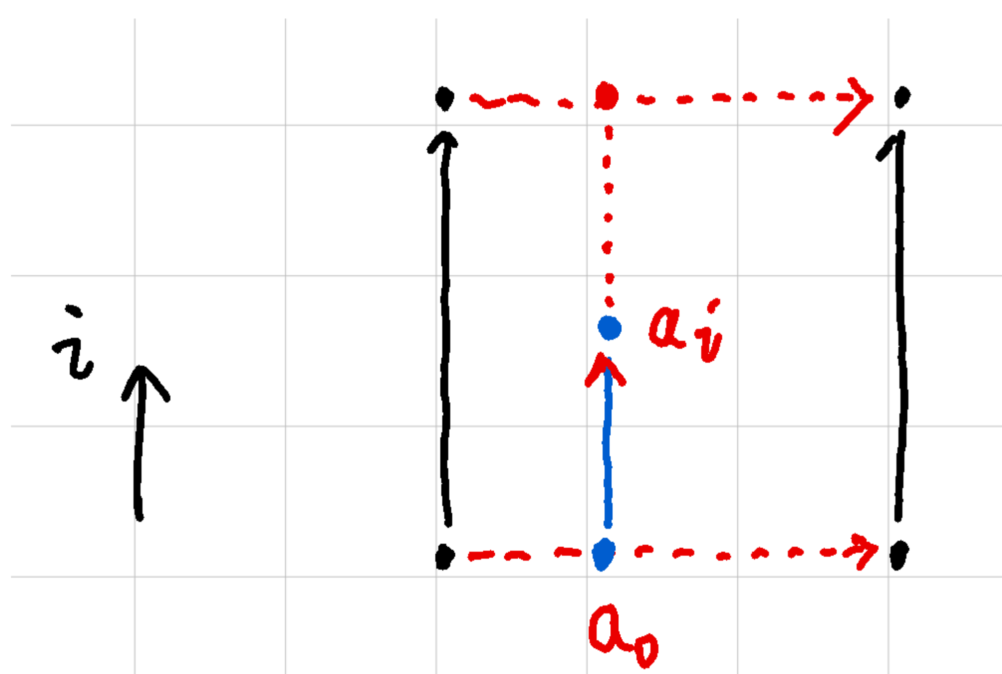
```
fill' : ∀ {ℓ}
      → (A : ∀ i → Type ℓ)
      → (φ : I)
      → (u : ∀ i → Partial φ (A i))
      → A i0 [ φ ↦ u i0 ]
      -------------------------------
      → (i : I) → A i
fill' A φ u a₀ i = outS (comp' A* (φ ∨ ~ i) u* (inS (outS a₀)))
  where
    A* : _
    A* = λ j → A (i ∧ j)

    u* : ∀ j → Partial (φ ∨ ~ i) _
    u* j (φ = i1) = u (i ∧ j) 1=1
    u* j (i = i0) = outS a₀
```

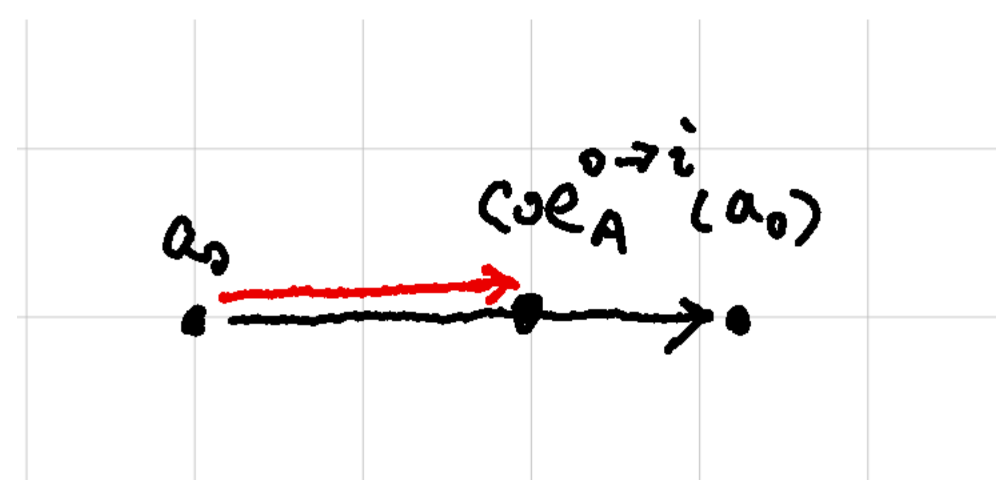
# Composition vs. Coercion?




```
coe0→1 : ∀ {ℓ} (A : I → Type ℓ) → A i0 → A i1
coe0→1 A a = transp (\ i → A i) i0 a
```




```
coe0→i : ∀ {ℓ} (A : I → Type ℓ) (i : I) → A i0 → A i
coe0→i A i a = transp (λ j → A (i ∧ j)) (~ i) a
```

# Composition vs. Coercion?

$$\frac{\begin{array}{c} A : \mathrm{Type} \\ x : \mathbb{I} \vdash a(x) : A \\ x : \mathbb{I} \vdash b(x) : A \\ q : \mathrm{Path}_A(a(0), b(0)) \end{array}}{\mathrm{coe}^{0 \rightsquigarrow 1}_{x.\mathrm{Path}_A(a(x), b(x))}(q) : \mathrm{Path}_A(a(1), b(1))}$$

$$\begin{tikzcd} a(0) \arrow[r, "a(x)"] \arrow[d, "q(y)"'] & a(1) \arrow[d, dashed] \\ b(0) \arrow[r, "b(x)"'] & b(1) \end{tikzcd}$$

Source: Carlo’s thesis slides

# Composition vs. Coercion?

$$\frac{\begin{array}{ll} & \Gamma \vdash M : A \\ (\forall i) & \Gamma, \xi_i, x : \mathbb{I} \vdash N_i : A \\ (\forall i, j) & \Gamma, \xi_i, \xi_j, x : \mathbb{I} \vdash N_i = N_j : A \\ (\forall i) & \Gamma, \xi_i \vdash N_i\langle r/x \rangle = M : A \end{array}}{\Gamma \vdash \mathrm{hcom}_A^{r \rightsquigarrow r'}(M; \overrightarrow{\xi_i \hookrightarrow x.N_i}) : A}$$

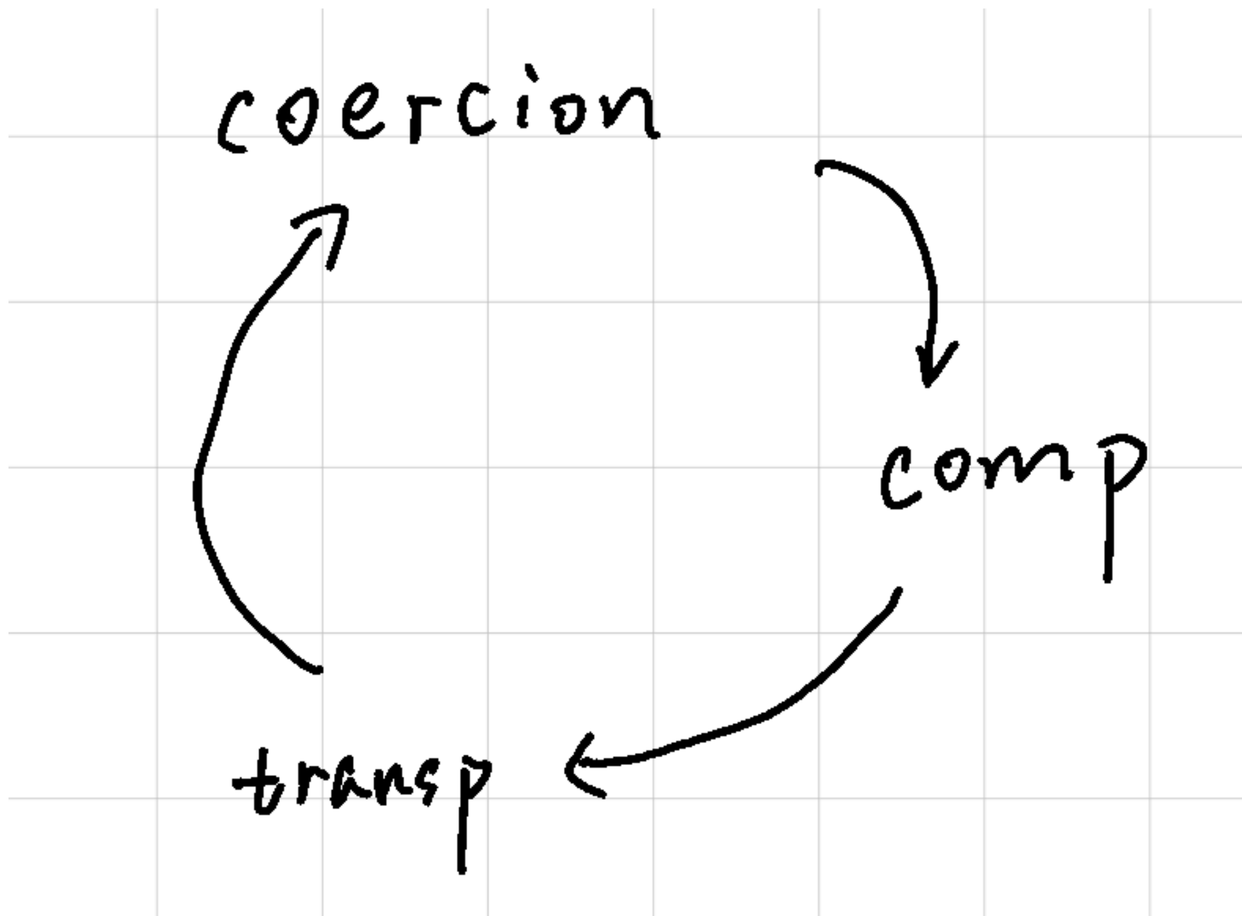
$$= \begin{cases} M & \text{when } r = r' \\ N_i\langle r'/x \rangle & \text{when } \xi_i \end{cases}$$

$$\begin{tikzcd}
a(0) \arrow[r, "a(x)"] \arrow[d, "q(y)"'] & a(1) \arrow[d, dashed] \\
b(0) \arrow[r, "b(x)"'] & b(1)
\end{tikzcd}$$

$$\mathrm{hcom}_A^{0 \rightsquigarrow 1}(q(y); y = 0 \hookrightarrow x.a(x), y = 1 \hookrightarrow x.b(x))$$

Source: Carlo’s thesis slides

# Composition vs. Coercion?



# Questions

Cubical sets, presheafs, Kan fibrations …

# To Do

- more comparison between CCHM and Cartesian
- gluing and univalence
- semantics models for cubical type theory
- differential; derivative of a program