The Curry-Howard Correspondence, and beyond

Formulas	Types	Objects	Games
Proofs	Terms	Morphisms	Strategies

Further Reading: *Proofs and Types* by Girard, Lafont and Taylor, *Basic Simple Type Theory* by Hindley, both published by Cambridge University press.

Formal Proofs

Proof of A from **assumptions** $A_1, \ldots, A_n$:

$$A_1, \ldots, A_n \vdash A$$

We use Γ, Δ to range over finite sets of formulas, writing $\Gamma \vdash A$ *etc.*

We shall focus on the fragment of propositional logic based on **conjunction** $A \wedge B$ and **implication** $A \supset B$.

Natural Deduction system for $\wedge$, $\supset$

Identity

$$\frac{}{\Gamma, A \vdash A}\ \mathsf{Id}$$

Conjunction

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge\text{-intro} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge\text{-elim-1} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge\text{-elim-2}$$

Implication

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\ \supset\text{-intro} \qquad \frac{\Gamma \vdash A \supset B \qquad \Gamma \vdash A}{\Gamma \vdash B}\ \supset\text{-elim}$$

Structural Proof Theory

The idea is to study the ‘space of formal proofs’ as a mathematical structure in its own right, rather than to focus only on

$$\text{Provability} \longleftrightarrow \text{Truth}$$

(i.e. the usual notions of ‘soundness and completeness’).

Why? One motivation comes from trying to understand and use the **computational content of proofs**. To make this precise, we look at the ‘Curry-Howard correspondence’.

Terms

λ-calculus: a pure calculus of functions.

Variables x, y, z, ...

Terms

$$t \quad ::= \quad x \;\mid\; \underbrace{tu}_{\text{application}} \;\mid\; \underbrace{\lambda x.\, t}_{\text{abstraction}}$$

Examples

$\lambda x.\, x+1$	successor function
$\lambda x.\, x$	identity function
$\lambda f.\, \lambda x.\, fx$	application
$\lambda f.\, \lambda x.\, f(fx)$	double application
$\lambda f.\, \lambda g.\, \lambda x.\, g(f(x))$	composition $g \circ f$

Conversion and Reduction

The basic equation governing this calculus is β-**conversion**:

$$(\lambda x.\, t)u = t[u/x]$$

E.g.

$$(\lambda f.\, \lambda x.\, f(fx))(\lambda x.\, x+1)0 = \cdots 2.$$

By orienting this equation, we get a 'dynamics' - β-**reduction**

$$(\lambda x.\, t)u \to t[u/x]$$

From type-free to typed

'Pure' λ-calculus is **very** unconstrained.

For example, it allows terms like $\omega \equiv \lambda x.\, xx$ — self-application.

Hence $\Omega \equiv \omega\omega$, which **diverges**:

$$\Omega \to \Omega \to \cdots$$

Also, $\mathbf{Y} \equiv \lambda f.\, (\lambda x.\, f(xx))(\lambda x.\, f(xx))$ — recursion.

$$\mathbf{Y}t \to (\lambda x.\, t(xx))(\lambda x.\, t(xx)) \to t((\lambda x.\, t(xx))(\lambda x.\, t(xx))) = t(\mathbf{Y}t).$$

Historically, Curry extracted $\mathbf{Y}$ from an analysis of Russell's Paradox.

Simply-Typed λ-calculus

Base types

$$B \quad ::= \quad \iota \mid \ldots$$

General Types

$$T \quad ::= \quad B \mid T \to T \mid T \times T$$

Examples

$\iota \to \iota \to \iota$	first-order function type
$(\iota \to \iota) \to \iota$	second-order function type

In general, any simple type built purely from base types and function types can be written as

$$T_1 \to T_2 \to \cdots T_k \to B$$

where the T_i are again of this form.

Rank and Order

We can define the **rank** of a type:

$$\begin{array}{lcl}\rho(B) & = & 0\\ \rho(T\times U) & = & \max(\rho(T),\rho(U))\\ \rho(T\to U) & = & \max(\rho(T)+1,\rho(U))\end{array}$$

$\rho(T)=1$ means that T is 'first-order'.

Typed terms

Typing judgement:

$$x_1 : T_1, \ldots x_k : T_k \vdash t : T$$

the term t has type T **under the assumption** (or: **in the context**) that the variable x_1 has type T_1, …, x_k has type T_k.

The System of Simply-Typed λ-calculus

Variable

$$\frac{}{\Gamma, x : t \vdash x : T}$$

Product

$$\frac{\Gamma\vdash t:T \qquad \Gamma\vdash u:U}{\Gamma\vdash\langle t,u\rangle : T\times U} \qquad \frac{\Gamma\vdash v:T\times U}{\Gamma\vdash \pi_1 v : T} \qquad \frac{\Gamma\vdash v:T\times U}{\Gamma\vdash \pi_2 v : U}$$

Function

$$\frac{\Gamma, x:U\vdash t:T}{\Gamma\vdash\lambda x.\,t : U\to T} \qquad \frac{\Gamma\vdash t:U\to T \qquad \Gamma\vdash u:U}{\Gamma\vdash tu : T}$$

Reduction rules

Computation rules (β-reductions):

$$\begin{array}{lcl}(\lambda x.\,t)u & \to & t[u/x]\\ \pi_1\langle t,u\rangle & \to & t\\ \pi_2\langle t,u\rangle & \to & u\end{array}$$

Also, η-laws (extensionality principles):

$$\begin{array}{lcll}t & = & \lambda x.\,tx & x \text{ not free in } t\text{, at function types}\\ v & = & \langle\pi_1 v,\pi_2 v\rangle & \text{at product types}\end{array}$$

Compare the Simple Type system to the Natural Deduction system for $\wedge$, $\supset$.

If we equate

$$\begin{array}{ccc} \wedge & \equiv & \times \\ \supset & \equiv & \rightarrow \end{array}$$

they are the same!

This is the **Curry-Howard correspondence** (sometimes: 'Curry-Howard isomorphism').

It works on three levels:

Formulas	Types
Proofs	Terms
Proof transformations	Term reductions

Constructive reading of formulas

The 'Brouwer-Heyting-Kolmogorov interpretation'.

- A proof of an implication $A \supset B$ is a construction which transforms any proof of A into a proof of B.
- A proof of $A \wedge B$ is a pair consisting of a proof of A and a proof of B.

Thse readings motivate identifying $A \wedge B$ with $A \times B$, and $A \supset B$ with $A \rightarrow B$.

Moreover, these ideas have strong connections to computing. The λ-calculus is a 'pure' version of functional programming languages such as Haskell and SML. So we get a reading of

Proofs as Programs

Three Theorems on Simple Types

- Proofs **about** proofs or terms — **meta**-mathematics.
- Exploring the structure of formal systems — their behaviour under 'dynamics', *i.e.* reduction.
- Main proof technique: induction on syntax.

Induction on Syntax

Since proofs have been formalized as 'concrete objects', *i.e.* trees, we can assign numerical measures such as **height** or **size** to them, and use mathematical induction on these quantities.

Height of a term:

$$\begin{array}{lcl} \mathsf{height}(x) & = & 1 \\ \mathsf{height}(\lambda x.\, t) & = & \mathsf{height}(t) + 1 \\ \mathsf{height}(tu) & = & \max(\mathsf{height}(t), \mathsf{height}(u)) + 1 \end{array}$$

Draw pictures!

Reduction revisited

β-reduction:

$$(\lambda x.\, u)v \to u[v/x]$$

A **redex** of a term t is a subexpression of the form of the left-hand-side of the above rule, to which β-reduction can be applied. A term is in **normal form** of it contains no redexes. We write $t \twoheadrightarrow u$ if u can be obtained from t by a number of applications of β-reduction. Thus $\twoheadrightarrow$ is a reflexive and transitive relation.

Substitution:

$$x[t/x] = t \qquad y[t/x] = y \quad (x \neq y)$$

$$(\lambda z.\, u)[t/x] = \lambda z.\, (u[t/x]) \qquad (*)$$

$$(uv)[t/x] = (u[t/x])(v[t/x])$$

Three Theorems

1. The Church-Rosser Theorem If $t \twoheadrightarrow u$ and $t \twoheadrightarrow v$ then for some w, $u \twoheadrightarrow w$ and $v \twoheadrightarrow w$.

(Proved in Lambda Calculus course).

2. The Subject Reduction Theorem ‘Typing is invariant under reduction’. If $\Gamma \vdash t : T$ and $t \twoheadrightarrow u$, then $\Gamma \vdash u : T$.

3. Weak Normalization If t is typable in Simple Types, then t has a normal form (necessarily unique by Church-Rosser).

Key Lemma for Subject Reduction

Lemma The following ‘Cut Rule’ is admissible in Simple Types; *i.e.* whenever we can prove the premises of the rule, we can also prove the conclusion.

$$\frac{\Gamma, x : U \vdash t : T \qquad \Gamma \vdash u : U}{\Gamma \vdash t[u/x] : T}$$

The proof is by induction on the derivation of $\Gamma, x : U \vdash t : T$. (Equivalently, by induction on $\mathsf{height}(t)$).

Normalization in simple types is non-elementary

Define $e(m,n)$ by $e(m,0) = m$, $e(m,n+1) = 2^{e(m,n)}$. Thus $e(m,n)$ is an exponential ‘stack’ of n 2’s with an m at the top:

$$e(m,n) \quad = \quad 2^{2^{\cdot^{\cdot^{\cdot^{2^m}}}}}$$

We can prove that a term of degree d and height h has a normal form of height bounded by $e(h,d)$. (Details in next Exercise Sheet). However, there is no **elementary** bound (*i.e.* an exponential stack of fixed height).

The connection to Categories

Let $\mathcal{C}$ be a category. We shall interpret Formulas (or Types) as Objects of $\mathcal{C}$.

A morphism $f : A \longrightarrow B$ will then correspond to a **proof of** B **from assumption** A, *i.e.* a proof of $A \vdash B$. Note that the bare structure of a category only supports proofs from a single assumption.

Now suppose $\mathcal{C}$ has finite products. A proof of

$$A_1, \ldots, A_k \vdash A$$

will correspond to a morphism

$$f : A_1 \times \cdots \times A_k \longrightarrow A.$$

Axiom

$$\frac{}{\Gamma, A \vdash A}\ \mathsf{Id} \qquad\qquad \frac{}{\pi_2 : \Gamma \times A \longrightarrow A}$$

Conjunction

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge\text{-intro} \qquad\qquad \frac{f : \Gamma \longrightarrow A \qquad g : \Gamma \longrightarrow B}{\langle f, g \rangle : \Gamma \longrightarrow A \times B}$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge\text{-elim-1} \qquad\qquad \frac{f : \Gamma \longrightarrow A \times B}{\pi_1 \circ f : \Gamma \longrightarrow A}$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge\text{-elim-2} \qquad\qquad \frac{f : \Gamma \longrightarrow A \times B}{\pi_2 \circ f : \Gamma \longrightarrow B}$$

Implication

Now let $\mathcal{C}$ be cartesian closed.

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\ \supset\text{-intro} \qquad\qquad \frac{f : \Gamma \times A \longrightarrow B}{\Lambda(f) : \Gamma \longrightarrow (A \Rightarrow B)}$$

$$\frac{\Gamma \vdash A \supset B \qquad \Gamma \vdash A}{\Gamma \vdash B}\ \supset\text{-elim} \qquad\qquad \frac{f : \Gamma \longrightarrow (A \Rightarrow B) \qquad g : \Gamma \longrightarrow A}{\mathsf{Ap}_{A,B} \circ \langle f, g \rangle : \Gamma \longrightarrow B}$$

Moreover, the β- and η-equations are all then **derivable** from the equations of cartesian closed categories.

So cartesian closed categories are **models** of $\wedge$, $\supset$-logic, at the level of **proofs** and **proof transformations**, and of simply typed λ-calculus, at the level of **terms** and **equations between terms**.

Linearity

Implicit in our treatment of assumptions

$$A_1, \ldots, A_n \vdash A$$

is that we can use them as many times as we want (including not at all).

To make these more visible, we now represent the assumptions as a **list** (possibly with repetitions) rather than a set, and use explicit structural rules to control copying and deletion of assumptions.

Thus we replace the identity by

$$\frac{}{A \vdash A}\ \mathsf{Id}$$

and introduce the structural rules

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}\ \text{Exchange}$$

$$\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ \text{Contraction} \qquad \frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ \text{Weakening}$$

In terms of the product structure we use using for the categorical intepretation of lists of assumptions, these structural rules have clear meanings.

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}\ \text{Exchange}$$

$$\frac{f : \Gamma \times A \times B \times \Delta \longrightarrow C}{f \circ (\mathsf{id}_\Gamma \times s_{A,B} \times \mathsf{id}_\Delta) : \Gamma \times B \times A \times \Delta \longrightarrow C}$$

$$\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ \text{Contraction} \qquad \frac{f : \Gamma \times A \times A \longrightarrow B}{f \circ (\mathsf{id}_\Gamma \times \Delta_A) : \Gamma \times A \longrightarrow B}$$

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ \text{Weakening} \qquad \frac{f : \Gamma \longrightarrow B}{f \circ \pi_1 : \Gamma \times A \longrightarrow B}$$

What happens if we **drop** the Contraction and Weakening rules (but keep the Exchange rule)?

It turns out we can still make good sense of the resulting proofs, terms and categories, but now in the setting of a different, 'resource-sensitive' logic:

Linear Logic

Formulas: $A \otimes B$, $A \multimap B$.

Sequents are still written $\Gamma \vdash A$ but Γ is now a **multiset**.

Linear Logic: Proofs

Axiom

$$\frac{}{A \vdash A}$$

Tensor

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \qquad \frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C}$$

Linear Implication

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B} \qquad \frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B}$$

Cut Rule

$$\frac{\Gamma \vdash A \qquad A, \Delta \vdash B}{\Gamma, \Delta \vdash B}$$

Note the following:

- The use of **disjoint** (*i.e.* non-overlapping) contexts.
- In the presence of Contraction and Weakening, the rules given for $\otimes$ and $\multimap$ are equivalent to those previously given for $\wedge$ and $\supset$.
- The system given was chosen to emphasize the parallels with the system for $\wedge$, $\supset$. However, to obtain a system in which 'Cut-elimination' holds, one should replace the 'elimination rule' given for Linear implication by the following '$\multimap$-left' rule:

$$\frac{\Gamma \vdash A \qquad B, \Delta \vdash C}{\Gamma, A \multimap B, \Delta \vdash C}$$

Linear Logic: terms

Judgements will look much the same as previously, but term formation is now highly constrained by the form of the typing judgements. In particular,

$$x_1 : A_1, \ldots, x_k : A_k \vdash t : A$$

will now imply that each x_i occurs **exactly once** (free) in t.

Linear Logic: Term Assignment for Proofs

Axiom

$$\frac{}{x : A \vdash x : A}$$

Tensor

$$\frac{\Gamma \vdash t : A \qquad \Delta \vdash u : B}{\Gamma, \Delta \vdash t \otimes u : A \otimes B} \qquad \frac{\Gamma, x : A, y : B \vdash v : C}{\Gamma, z : A \otimes B \vdash \mathbf{let}\ z\ \mathsf{be}\ x \otimes y\ \mathbf{in}\ v : C}$$

Linear Implication

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.\, t : A \multimap B} \qquad \frac{\Gamma \vdash t : A \multimap B \qquad \Delta \vdash u : A}{\Gamma, \Delta \vdash tu : B}$$

Cut Rule

$$\frac{\Gamma \vdash t : A \qquad x : A, \Delta \vdash u : B}{\Gamma, \Delta \vdash u[t/x] : B}$$

Reductions

$$\begin{array}{lcl} (\lambda x.\, t)u & \to & t[u/x] \\ \mathbf{let}\ t \otimes u\ \mathsf{be}\ x \otimes y\ \mathbf{in}\ v & \to & v[t/x, u/y] \\ & \vdots & \end{array}$$

Term assignment for $\multimap$-left

$$\frac{\Gamma \vdash t : A \qquad x : B, \Delta \vdash u : C}{\Gamma, f : A \multimap B, \Delta \vdash u[ft/x] : C}$$

Monoidal Categories

A monoidal category is a structure $(\mathcal{C}, \otimes, I, a, l, r)$ where

- $\mathcal{C}$ is a category
- $\otimes : \mathcal{C} \times \mathcal{C} \longrightarrow \mathcal{C}$ is a functor
- a, l, r are natural isomorphisms

$$a_{A,B,C} : A \otimes (B \otimes C) \xrightarrow{\cong} (A \otimes B) \otimes C$$

$$l_A : I \otimes A \xrightarrow{\cong} A \qquad\qquad r_A : A \otimes I \xrightarrow{\cong} A$$

such that the following equations hold for all A, B, C, D:

$$a_{A,I,B} ; r_A \otimes \mathsf{id}_B = \mathsf{id}_A \otimes l_B$$

$$\mathsf{id}_A \otimes a_{B,C,D} ; a_{A,B \otimes C,D} ; a_{A,B,C} \otimes \mathsf{id}_D = a_{A,B,C \otimes D} ; a_{A \otimes B,C,D}.$$

The Pentagon

$$\begin{tikzcd}
A \otimes (B \otimes (C \otimes D)) \arrow[r, "a"] \arrow[d, "\mathsf{id} \otimes a"'] & (A \otimes B) \otimes (C \otimes D) \arrow[r, "a"] & ((A \otimes B) \otimes C) \otimes D \\
A \otimes ((B \otimes C) \otimes D) \arrow[rr, "a"] & & (A \otimes (B \otimes C)) \otimes D \arrow[u, leftarrow, "a \otimes \mathsf{id}"']
\end{tikzcd}$$

$$\begin{tikzcd}
A \otimes (I \otimes B) \arrow[r, "a"] \arrow[d, "\mathsf{id} \otimes l"'] & (A \otimes I) \otimes B \arrow[dl, "r \oplus \mathsf{id}"] \\
A \otimes B &
\end{tikzcd}$$

Examples

- Both products and coproducts give rise to monoidal structures — which are the common denominator between them. (But in addition, products have **diagonals** and **projections**).
- $(\mathbb{N}, \leq, +, 0)$ is a monoidal category.
- **Rel**, the category of sets and relations, with cartesian product (which is **not** the categorical product).
- **Vect** with the tensor product.

Symmetric Monoidal Categories

A symmetric monoidal category is a monoidal category $(\mathcal{C}, \otimes, I, a, l, r)$ with an additional natural isomorphism

$$s_{A,B} : A \otimes B \xrightarrow{\cong} B \otimes A$$

such that the following equations hold for all A, B, C:

$$s_{A,B} ; s_{B,A} = \mathsf{id}_{A \otimes B} \qquad\qquad s_{A,I} ; l_A = r_A$$

$$a_{A,B,C} ; s_{A \otimes B,C} ; a_{C,A,B} = \mathsf{id}_A \otimes s_{B,C} ; a_{A,C,B} ; s_{A,C} \otimes \mathsf{id}_B.$$

Symmetric Monoidal Closed categories

A symmetric monoidal closed category is a symmetric monoidal category $(\mathcal{C}, \otimes, I, a, l, r, s)$ such that, for each object A, the is a couniversal arrow to the functor

$$- \otimes A : \mathcal{C} \longrightarrow \mathcal{C}$$

This means that for all A and B there is an object $A \multimap B$ and a morphism

$$\mathsf{Ap}_{A,B} : (A \multimap B) \otimes A \longrightarrow B$$

Moreover, for every morphism $f : C \otimes A \longrightarrow B$, there is a unique morphism $\Lambda(f) : C \longrightarrow (A \multimap B)$ such that

$$\mathsf{Ap}_{A,B} \circ (\Lambda(f) \otimes \mathsf{id}_A) = f.$$

Examples

- **Vect**$_k$. Here $\otimes$ is the tensor product of vector spaces, and $A \multimap B$ is the vector space of linear maps.
- **Rel**, the category with objects sets and morphisms **relations**. Here we take $\otimes$ to be cartesian product (which is **not** the categorical product in **Rel**).
- A cartesian closed category is a special case of a symmetric monoidal closed category, where $\otimes$ is taken to be the product.

Linear Logic: Categories

Just as cartesian closed categories correspond to Simply-typed λ-calculus/$(\wedge, \supset)$–logic, so **symmetric monoidal closed categories** correspond to Linear λ-calculus/$(\otimes, \multimap)$–logic.

Let $(\mathcal{C}, \otimes, \ldots)$ be a symmetric monoidal closed category.

The interpretation of a Linear inference

$$A_1, \ldots, A_k \vdash A$$

will be a morphism

$$f : A_1 \otimes \cdots \otimes A_k \longrightarrow A.$$

To be precise in our interpretation, we will treat contexts as **lists** of formulas, and explicitly interpret the Exchange rule:

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C} \qquad \frac{f : \Gamma \otimes A \otimes B \otimes \Delta \longrightarrow C}{f \circ (\mathsf{id}_\Gamma \otimes s_{A,B} \otimes \mathsf{id}_\Delta) : \Gamma \otimes B \otimes A \otimes \Delta \longrightarrow C}$$

Categorical interpretation of Linear proofs (I)

Axiom

$$\frac{}{A \vdash A} \qquad \frac{}{\mathsf{id}_A : A \longrightarrow A}$$

Tensor

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \qquad \frac{f : \Gamma \longrightarrow A \qquad g : \Delta \longrightarrow B}{f \otimes g : \Gamma \otimes \Delta \longrightarrow A \otimes B}$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C} \qquad \frac{f : (\Gamma \otimes A) \otimes B \longrightarrow C}{f \circ a_{A,B,C} : \Gamma \otimes (A \otimes B) \longrightarrow C}$$

Categorical interpretation of Linear proofs (II)

Linear Implication

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B} \qquad \frac{f : \Gamma \otimes A \longrightarrow B}{\Lambda(f) : \Gamma \longrightarrow (A \multimap B)}$$

$$\frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B} \qquad \frac{f : \Gamma \longrightarrow (A \multimap B) \qquad g : \Delta \longrightarrow A}{\mathsf{Ap} \circ (f \otimes g) : \Gamma \otimes \Delta \longrightarrow B}$$

Cut Rule

$$\frac{\Gamma \vdash A \qquad A, \Delta \vdash B}{\Gamma, \Delta \vdash B} \qquad \frac{f : \Gamma \longrightarrow A \qquad g : A \otimes \Delta \longrightarrow B}{g \circ (f \otimes \mathsf{id}_\Delta) : \Gamma \otimes \Delta \longrightarrow B}$$

Linear Logic: beyond the multiplicatives

Linear Logic has three 'levels' of connectives:

- The **multiplicatives**, e.g. $\otimes$, $\multimap$
- The **additives**: additive conjunction & and disjunction $\oplus$
- the **exponentials**, allowing controlled access to copying and discarding

Additive Conjunction

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \& B} \qquad \frac{\Gamma, A \vdash C}{\Gamma, A \& B \vdash C} \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \& B \vdash C}$$

The additive conjunction can be interpreted in any symmetric monoidal closed category with products (*e.g.* our category of games).

Note that, since by linearity an argument of type $A \& B$ can only be used once, each use of a left rule for & makes a once-and-for-all **choice** of a projection.

Term assignment for additive conjunction

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : B}{\Gamma \vdash \langle t, u \rangle : A \& B}$$

$$\frac{\Gamma, x : A \vdash t : C}{\Gamma, z : A \& B \vdash \mathbf{let}\ z = \langle x, - \rangle\ \mathbf{in}\ t : C}$$

$$\frac{\Gamma, B \vdash C}{\Gamma, z : A \& B \vdash \mathbf{let}\ z = \langle -, y \rangle\ \mathbf{in}\ t : C}$$

Reduction rules

$$\begin{array}{lcl} \mathbf{let}\ \langle t, u \rangle = \langle x, - \rangle\ \mathbf{in}\ v & \to & v[t/x] \\ \mathbf{let}\ \langle t, u \rangle = \langle -, y \rangle\ \mathbf{in}\ v & \to & v[u/y] \end{array}$$

Exponential

$!A$: a kind of modality (*cf.* $\Box A$)

Rules:

$$\frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B} \qquad \frac{\Gamma \vdash B}{\Gamma, !A \vdash B} \qquad \frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B} \qquad \frac{!\Gamma \vdash A}{!\Gamma \vdash !A}$$

Interpreting standard Natural Deduction

We can use the exponential to recover the 'expressive power' of the usual logical connectives $\wedge$, $\supset$. If we interpret

$$\begin{array}{lcl} A \supset B & \triangleq & !A \multimap B \\ A \wedge B & \triangleq & A \& B \end{array}$$

and an inference

$$\Gamma \vdash A$$

in standard Natural Deduction for $\wedge$, $\supset$-logic as

$$!\Gamma \vdash A$$

in Linear Logic, then each proof rule of Natural Deduction for $\wedge$, $\supset$ can be interpreted in Linear Logic (and exactly the same formulas of $\wedge$, $\supset$-logic are provable).

Note in particular that the interpretation

$$A \supset B \triangleq !A \multimap B$$

decomposes the fundamental notion of **implication** into finer notions — like 'splitting the atom of logic'!