# Substructural Logic
## (Linear Logic and Linear Type Systems)

Hype for Types

February 25, 2024

# What We'll Talk About

- What it means for a logic to be "substructural"

# What We'll Talk About

- What it means for a logic to be "substructural"
- A case study of a particular substructural logic (linear logic)

# What We'll Talk About

- What it means for a logic to be "substructural"
- A case study of a particular substructural logic (linear logic)
- How do servers work?

# Substructural Logic

# She Sub on my Structure

The constructive logic we have been working with so far has the following admissible rules, which we call "structural properties" of the logic:

$$\frac{\Gamma \vdash C}{\Gamma, A \vdash C}\ (\textsc{Weak}) \qquad\qquad \frac{\Gamma, A, A \vdash C}{\Gamma, A \vdash C}\ (\textsc{Cntr})$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, B, A \vdash C}\ (\textsc{Exch})$$

# She Sub on my Structure

What happens if you remove some of these structural properties?

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)
- Relevance Logic: no weakening (Use premises at least once)

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)
- Relevance Logic: no weakening (Use premises at least once)
- Linear Logic: no weakening or contraction (Use premises exactly once)

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)
- Relevance Logic: no weakening (Use premises at least once)
- Linear Logic: no weakening or contraction (Use premises exactly once)
- Ordered Logic: no weakening, contraction, or exchange (Use premises exactly once and order matters)

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)
- Relevance Logic: no weakening (Use premises at least once)
- Linear Logic: no weakening or contraction (Use premises exactly once)
- Ordered Logic: no weakening, contraction, or exchange (Use premises exactly once and order matters)

**Question**

What are the consequences of not having these structural properties?

# She Sub on my Structure

What happens if you remove some of these structural properties? You would get a new logical system!

- Affine Logic: no contraction (Use premises at most once)
- Relevance Logic: no weakening (Use premises at least once)
- Linear Logic: no weakening or contraction (Use premises exactly once)
- Ordered Logic: no weakening, contraction, or exchange (Use premises exactly once and order matters)

**Question**

What are the consequences of not having these structural properties?

Today, we'll be focusing on *linear logic*, and how we can use it to model a client-server protocol.

# Linear Logic

# Is Logic Logical?

*The moon is made of green cheese. Therefore, you come to hype for types today.*

**Question**

Is this logical?

# Different Interpretation of Implication

Constructive logic interprets $A \Rightarrow B$ as "If you give me $A$ is true, then I give you $B$ is true".
But what it really says is "If you give me as many copy of $A$ as I need, then I give you $B$ is true".

# Different Interpretation of Implication

Constructive logic interprets $A \Rightarrow B$ as "If you give me $A$ is true, then I give you $B$ is true".
But what it really says is "If you give me as many copy of $A$ as I need, then I give you $B$ is true".

**Idea**

The problem of previous example is that to prove the conclusion we only need zero copies of the assumption, hence lacking "relevance".

# Different Interpretation of Implication

Constructive logic interprets $A \Rightarrow B$ as "If you give me $A$ is true, then I give you $B$ is true".
But what it really says is "If you give me as many copy of $A$ as I need, then I give you $B$ is true".

**Idea**

The problem of previous example is that to prove the conclusion we only need zero copies of the assumption, hence lacking "relevance".

**Idea**

We need a logic that forces relevance.

# Malloc is Scary...

Consider the following C code:

```c
int main () {
  char *str;
  str = (char *) malloc(13);
  strcpy(str, "hypefortypes");
  free(str);
  return(0);
}
```

In C, we have to make sure we allocate and deallocate every memory cell exactly once.

# Malloc is Scary...

Consider the following C code:

```c
int main () {
  char *str;
  str = (char *) malloc(13);
  strcpy(str, "hypefortypes");
  free(str);
  return(0);
}
```

In C, we have to make sure we allocate and deallocate every memory cell exactly once.

**Question**

Is there a way to make our *types* guarantee correctness?

# The Problem With Constructive Logic

In "normal" constructive logic, we have no concept of *state*.

# The Problem With Constructive Logic

In “normal” constructive logic, we have no concept of *state*.

**Big Idea**

Proofs should no longer be *persistent*, but rather *ephemeral*.

# The Problem With Constructive Logic

In "normal" constructive logic, we have no concept of *state*.

**Big Idea**

Proofs should no longer be *persistent*, but rather *ephemeral*.

Persistence is due to implicit **structural rules**: weakening and contraction.

# Weakening

```c
int main() {
  int *x = (int *) malloc(sizeof(int));
  *x = 3;
  return 0;
}
```

# Weakening

```c
int main() {
  int *x = (int *) malloc(sizeof(int));
  *x = 3;
  return 0;
}
```

**Weakening**: we can "drop" assumptions

$$\frac{\Gamma \vdash e : \tau}{\Gamma, x : \tau' \vdash e : \tau} \; (\textsc{Weak})$$

# Contraction

```
void f(int *x) {
  free(x);
}

int main() {
  int *x = (int *) malloc(sizeof(int));
  *x = 3;
  f(x);
  f(x);
  return 0;
}
```

# Contraction

```
void f(int *x) {
  free(x);
}

int main() {
  int *x = (int *) malloc(sizeof(int));
  *x = 3;
  f(x);
  f(x);
  return 0;
}
```

**Contraction**: we can "duplicate" assumptions

$$\frac{\Gamma, x_1 : \tau, x_2 : \tau \vdash e : \tau'}{\Gamma, x : \tau \vdash [x, x/x_1, x_2]e : \tau'}\ (\textsc{Cntr})$$

# Introduction to Linear Logic

In **linear logic**, we have neither weakening nor contraction.

- Requirement that we use each piece of data *exactly* once - no duplication, no dropping
- Comes with an inherent idea of “resources” that are used up
- Allows us to write safe, stateful (imperative!) programs

# The Linear Rules

# Identity

**Constructive Logic**

$$\frac{A \in \Gamma}{\Gamma \vdash A}\ (\textsc{Hyp})$$

# Identity

**Constructive Logic**

$$\frac{A \in \Gamma}{\Gamma \vdash A} \; (\textsc{Hyp})$$

**Linear Logic**

$$\frac{}{A \vdash A} \; (\textsc{Hyp})$$

# Identity

**Constructive Logic**

$$\frac{A \in \Gamma}{\Gamma \vdash A}\ (\textsc{Hyp})$$

**Linear Logic**

$$\frac{}{A \vdash A}\ (\textsc{Hyp})$$

### Intuition

"Given $A$ and nothing else, we can use up $A$"

# Conjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \wedge A_2}\ (\wedge\mathrm{I})$$

$$\frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_1}\ (\wedge\mathrm{E1}) \qquad \frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_2}\ (\wedge\mathrm{E2})$$

# Conjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \wedge A_2} \ (\wedge\mathrm{I})$$

$$\frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_1} \ (\wedge\mathrm{E1}) \qquad\qquad \frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_2} \ (\wedge\mathrm{E2})$$

**Linear Logic**

# Conjunction

## Constructive Logic

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \wedge A_2}\ (\wedge\mathrm{I})$$

$$\frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_1}\ (\wedge\mathrm{E1}) \qquad \frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_2}\ (\wedge\mathrm{E2})$$

## Linear Logic

$$\frac{\Delta_1 \vdash A_1 \qquad \Delta_2 \vdash A_2}{\Delta_1, \Delta_2 \vdash A_1 \otimes A_2}\ (\otimes\mathrm{I})$$

# Conjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \wedge A_2}\ (\wedge\mathrm{I})$$

$$\frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_1}\ (\wedge\mathrm{E1}) \qquad\qquad \frac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_2}\ (\wedge\mathrm{E2})$$

**Linear Logic**

$$\frac{\Delta_1 \vdash A_1 \qquad \Delta_2 \vdash A_2}{\Delta_1, \Delta_2 \vdash A_1 \otimes A_2}\ (\otimes\mathrm{I})$$

$$\frac{\Delta \vdash A_1 \otimes A_2 \qquad \Delta', A_1, A_2 \vdash C}{\Delta, \Delta' \vdash C}\ (\otimes\mathrm{E})$$

# Disjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_1) \qquad \frac{\Gamma \vdash A_2}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_2)$$

$$\frac{\Gamma \vdash A_1 \vee A_2 \qquad \Gamma, A_1 \vdash B \qquad \Gamma, A_2 \vdash B}{\Gamma \vdash B}\ (\vee E)$$

# Disjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_1) \qquad \frac{\Gamma \vdash A_2}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_2)$$

$$\frac{\Gamma \vdash A_1 \vee A_2 \qquad \Gamma, A_1 \vdash B \qquad \Gamma, A_2 \vdash B}{\Gamma \vdash B}\ (\vee E)$$

**Linear Logic**

# Disjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_1) \qquad \frac{\Gamma \vdash A_2}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_2)$$

$$\frac{\Gamma \vdash A_1 \vee A_2 \qquad \Gamma, A_1 \vdash B \qquad \Gamma, A_2 \vdash B}{\Gamma \vdash B}\ (\vee E)$$

**Linear Logic**

$$\frac{\Delta \vdash A_1}{\Delta \vdash A_1 \oplus A_2}\ (\oplus \mathrm{I1})$$

# Disjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_1) \qquad \frac{\Gamma \vdash A_2}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_2)$$

$$\frac{\Gamma \vdash A_1 \vee A_2 \qquad \Gamma, A_1 \vdash B \qquad \Gamma, A_2 \vdash B}{\Gamma \vdash B}\ (\vee E)$$

**Linear Logic**

$$\frac{\Delta \vdash A_1}{\Delta \vdash A_1 \oplus A_2}\ (\oplus \mathrm{I1}) \qquad \frac{\Delta \vdash A_2}{\Delta \vdash A_1 \oplus A_2}\ (\oplus \mathrm{I2})$$

# Disjunction

**Constructive Logic**

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_1) \qquad \frac{\Gamma \vdash A_2}{\Gamma \vdash A_1 \vee A_2}\ (\vee I_2)$$

$$\frac{\Gamma \vdash A_1 \vee A_2 \qquad \Gamma, A_1 \vdash B \qquad \Gamma, A_2 \vdash B}{\Gamma \vdash B}\ (\vee E)$$

**Linear Logic**

$$\frac{\Delta \vdash A_1}{\Delta \vdash A_1 \oplus A_2}\ (\oplus \mathrm{I1}) \qquad \frac{\Delta \vdash A_2}{\Delta \vdash A_1 \oplus A_2}\ (\oplus \mathrm{I2})$$

$$\frac{\Delta \vdash A_1 \oplus A_2 \qquad \Delta', A_1 \vdash B \qquad \Delta', A_2 \vdash B}{\Delta, \Delta' \vdash B}\ (\oplus \mathrm{E})$$

# Implication

**Constructive Logic**

$$\frac{\Gamma, A_1 \vdash A_2}{\Gamma \vdash A_1 \supset A_2}\ (\supset I) \qquad \frac{\Gamma \vdash A_1 \supset A_2 \quad \Gamma \vdash A_1}{\Gamma \vdash A_2}\ (\supset E)$$

# Implication

**Constructive Logic**

$$\frac{\Gamma, A_1 \vdash A_2}{\Gamma \vdash A_1 \supset A_2} \ (\supset I) \qquad \frac{\Gamma \vdash A_1 \supset A_2 \quad \Gamma \vdash A_1}{\Gamma \vdash A_2} \ (\supset E)$$

**Linear Logic**

# Implication

**Constructive Logic**

$$\frac{\Gamma, A_1 \vdash A_2}{\Gamma \vdash A_1 \supset A_2} \ (\supset I) \qquad \frac{\Gamma \vdash A_1 \supset A_2 \quad \Gamma \vdash A_1}{\Gamma \vdash A_2} \ (\supset E)$$

**Linear Logic**

$$\frac{\Delta, A_1 \vdash A_2}{\Delta \vdash A_1 \multimap A_2} \ (\multimap I)$$

# Implication

**Constructive Logic**

$$\frac{\Gamma, A_1 \vdash A_2}{\Gamma \vdash A_1 \supset A_2}\ (\supset I) \qquad \frac{\Gamma \vdash A_1 \supset A_2 \quad \Gamma \vdash A_1}{\Gamma \vdash A_2}\ (\supset E)$$

**Linear Logic**

$$\frac{\Delta, A_1 \vdash A_2}{\Delta \vdash A_1 \multimap A_2}\ (\multimap I) \qquad \frac{\Delta \vdash A_1 \multimap A_2 \quad \Delta' \vdash A_1}{\Delta, \Delta' \vdash A_2}\ (\multimap E)$$

# Model Real Worlds Using Linear Logic

5 dollars can buy one coffee and one donut.

$$\$5 \multimap \texttt{coffee} \otimes \texttt{donut}$$

# Model Real Worlds Using Linear Logic

5 dollars can buy one coffee and one donut.

$$\$5 \multimap \texttt{coffee} \otimes \texttt{donut}$$

Buffet entrance is 10 dollars. Once you enter, you can eat some beef, and with 2 more dollars you can eat some chicken.

$$\$10 \multimap \texttt{beef} \otimes (\$2 \multimap \texttt{chicken})$$