Won't you bar my neighbor[hood]?

brouwer's realizability and the bar principle

Jon Sterling

SlamData, Inc.

March 23, 2016

Spreads as Spaces and Trees

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

1. If $\vec{u} \in \mathfrak{S}$, then there exists an $x \in \mathbb{N}$ such that $\vec{u} \frown x \in \mathfrak{S}$.

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

1. If $\vec{u} \in \mathfrak{S}$, then there exists an $x \in \mathbb{N}$ such that $\vec{u} \frown x \in \mathfrak{S}$.
2. If $\vec{u} \frown x \in \mathfrak{S}$, then also $\vec{u} \in \mathfrak{S}$.

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

1. If $\vec{u} \in \mathfrak{S}$, then there exists an $x \in \mathbb{N}$ such that $\vec{u} \frown x \in \mathfrak{S}$.
2. If $\vec{u} \frown x \in \mathfrak{S}$, then also $\vec{u} \in \mathfrak{S}$.
3. Finally, $[] \in \mathfrak{S}$.

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

1. If $\vec{u} \in \mathfrak{S}$, then there exists an $x \in \mathbb{N}$ such that $\vec{u} \frown x \in \mathfrak{S}$.
2. If $\vec{u} \frown x \in \mathfrak{S}$, then also $\vec{u} \in \mathfrak{S}$.
3. Finally, $[] \in \mathfrak{S}$.

The predicate $\mathfrak{S}$ either defines the finite prefixes of paths down an infinite tree, or it defines the lattice of open sets of a topological space.

Spreads as Spaces and Trees

A *spread* is at the same time a topological space and a non-wellfounded tree.

A spread is defined by a predicate $\mathfrak{S}$ on lists of natural numbers subject to some laws:

1. If $\vec{u} \in \mathfrak{S}$, then there exists an $x \in \mathbb{N}$ such that $\vec{u} \frown x \in \mathfrak{S}$.
2. If $\vec{u} \frown x \in \mathfrak{S}$, then also $\vec{u} \in \mathfrak{S}$.
3. Finally, $[] \in \mathfrak{S}$.

The predicate $\mathfrak{S}$ either defines the finite prefixes of paths down an infinite tree, or it defines the lattice of open sets of a topological space.

Usually, we will work implicitly with the *universal spread*, which always says "yes".

Neighborhoods and Points, Prefixes and Paths

A list of naturals $\vec{u} \in \mathbb{N}*$ can be thought of as a neighborhood around a point, or as a prefix of a path through an infinite tree.

Neighborhoods and Points, Prefixes and Paths

A list of naturals $\vec{u} \in \mathbb{N}*$ can be thought of as a neighborhood around a point, or as a prefix of a path through an infinite tree.

A stream of naturals $\alpha \in \mathbb{N}^{\mathbb{N}}$ can be thought of as an ideal point in the spread (space), or as a path through the spread's infinite tree.

Neighborhoods and Points, Prefixes and Paths

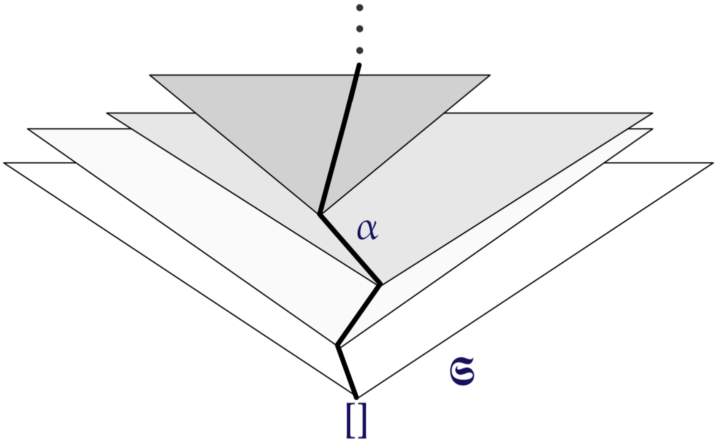
A list of naturals $\vec{u} \in \mathbb{N}*$ can be thought of as a neighborhood around a point, or as a prefix of a path through an infinite tree.

A stream of naturals $\alpha \in \mathbb{N}^{\mathbb{N}}$ can be thought of as an ideal point in the spread (space), or as a path through the spread's infinite tree.

$$\vec{u} \prec \alpha \qquad (\vec{u} \text{ approximates } \alpha)$$

$$\alpha \in \vec{u} \qquad (\vec{u} \text{ is a neighborhood around } \alpha)$$

Spread Visualization



Bars and Securability

A bar $\mathfrak{B}$ is a predicate on neighborhoods such that every point “hits it”. More generally, $\mathfrak{B}$ bars a neighborhood $\vec{u}$ when every path through $\vec{u}$ ends up in $\mathfrak{B}$.

Bars and Securability

A bar $\mathfrak{B}$ is a predicate on neighborhoods such that every point "hits it". More generally, $\mathfrak{B}$ bars a neighborhood $\vec{u}$ when every path through $\vec{u}$ ends up in $\mathfrak{B}$.

$$\forall \alpha \in \vec{u}.\ \exists \vec{v} \in \mathfrak{B}.\ \alpha \in \vec{v}$$

or equivalently

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

Bars and Securability

A bar $\mathfrak{B}$ is a predicate on neighborhoods such that every point "hits it". More generally, $\mathfrak{B}$ bars a neighborhood $\vec{u}$ when every path through $\vec{u}$ ends up in $\mathfrak{B}$.

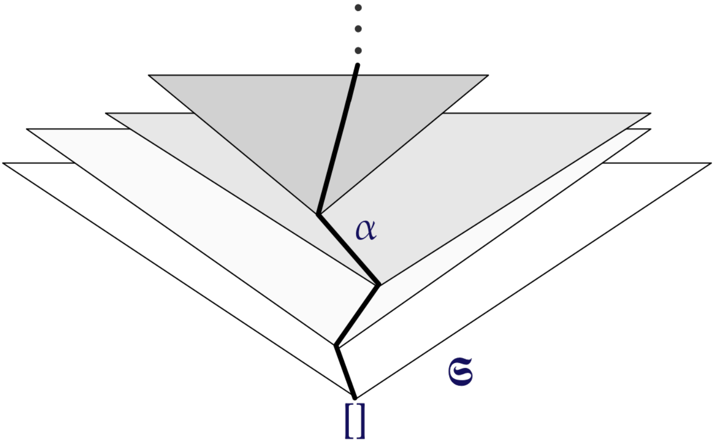
$$\forall \alpha \in \vec{u}.\ \exists \vec{v} \in \mathfrak{B}.\ \alpha \in \vec{v}$$

or equivalently

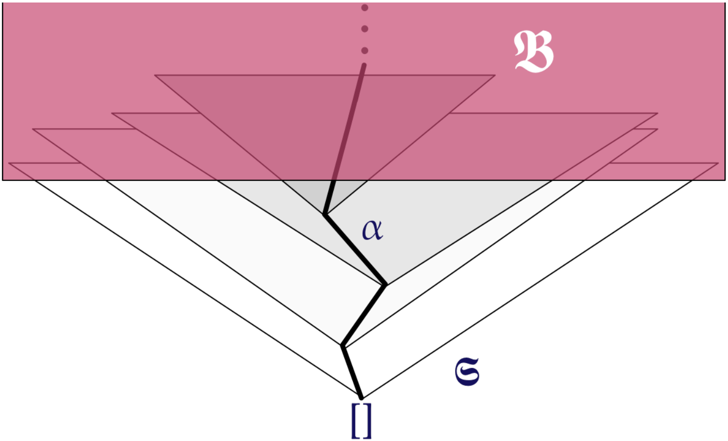
$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

We say that a neighborhood is secured when it is in the bar, and that it is securable when every path out of it eventually hits the bar.

Visualizing Bars



Visualizing Bars



- bars represent inevitability

- bars represent inevitability
- bars represent termination

- bars represent inevitability
- bars represent termination
- bars let us reason by induction on infinite trees

- bars represent inevitability
- bars represent termination
- bars let us reason by induction on infinite trees

(intuitionistic and classical, not constructive)

The Bar Induction Principle

Let $\mathfrak{B}$ be a decidable bar; let $\mathfrak{A}$ be another predicate on neighborhoods (the motive of induction). Suppose:

The Bar Induction Principle

Let $\mathfrak{B}$ be a decidable bar; let $\mathfrak{A}$ be another predicate on neighborhoods (the motive of induction). Suppose:

1. Every $\mathfrak{B}$-secured neighborhood is also in $\mathfrak{A}$.

$$\forall \vec{u} \in \mathfrak{B}.\ \vec{u} \in \mathfrak{A}$$

The Bar Induction Principle

Let $\mathfrak{B}$ be a decidable bar; let $\mathfrak{A}$ be another predicate on neighborhoods (the motive of induction). Suppose:

1. Every $\mathfrak{B}$-secured neighborhood is also in $\mathfrak{A}$.

$$\forall \vec{u} \in \mathfrak{B}.\ \vec{u} \in \mathfrak{A}$$

2. If every immediate refinement of a neighborhood is in $\mathfrak{A}$, then that neighborhood is also in $\mathfrak{A}$.

$$\forall \vec{u} \in \mathfrak{A}.\ \left(\forall x. \vec{u} \frown x \in \mathfrak{A}\right) \Rightarrow \vec{u} \in \mathfrak{A}$$

The Bar Induction Principle

Let $\mathfrak{B}$ be a decidable bar; let $\mathfrak{A}$ be another predicate on neighborhoods (the motive of induction). Suppose:

1. Every $\mathfrak{B}$-secured neighborhood is also in $\mathfrak{A}$.

$$\forall \vec{u} \in \mathfrak{B}.\ \vec{u} \in \mathfrak{A}$$

2. If every immediate refinement of a neighborhood is in $\mathfrak{A}$, then that neighborhood is also in $\mathfrak{A}$.

$$\forall \vec{u} \in \mathfrak{A}.\ \left(\forall x. \vec{u} \frown x \in \mathfrak{A}\right) \Rightarrow \vec{u} \in \mathfrak{A}$$

Then, every $\mathfrak{B}$-securable neighborhood is also in $\mathfrak{A}$. Or, equivalently:

$$[] \in \mathfrak{A}$$

What is the constructive meaning of the bar principle?

It asserts that the evidence that $\vec{u}$ is $\mathfrak{B}$-securable is a wellfounded/inductive tree.

What is the constructive meaning of the bar principle?

It asserts that the evidence that $\vec{u}$ is $\mathfrak{B}$-securable is a wellfounded/inductive tree.

Then, clause (1) gives us the base case, and clause (2) gives us the inductive step for concluding $\vec{u} \in \mathfrak{A}$.

What is the constructive meaning of the bar principle?

It asserts that the evidence that $\vec{u}$ is $\mathfrak{B}$-securable is a wellfounded/inductive tree.

Then, clause (1) gives us the base case, and clause (2) gives us the inductive step for concluding $\vec{u} \in \mathfrak{A}$.

The proof of $\vec{u} \in \mathfrak{A}$ proceeds by considering the possible ways in which $\vec{u}$ could be $\mathfrak{B}$-securable:

- η ▶ $\vec{u}$ is $\mathfrak{B}$-secured.
- ζ ▶ $\vec{u} \equiv \vec{v} \frown x$ such that $\vec{v}$ is $\mathfrak{B}$-securable
- $\digamma$ ▶ For all immediate refinements x, $\vec{u} \frown x$ is $\mathfrak{B}$-securable

Normalizing securability

We have identified three different ways that $\vec{u}$ could be $\mathfrak{B}$-securable:

- η ▶ $\vec{u}$ is $\mathfrak{B}$-secured.
- ζ ▶ $\vec{u} \equiv \vec{v} \frown x$ such that $\vec{v}$ is $\mathfrak{B}$-securable
- $\digamma$ ▶ For all immediate refinements x, $\vec{u} \frown x$ is $\mathfrak{B}$-securable

Normalizing securability

We have identified three different ways that $\vec{u}$ could be $\mathfrak{B}$-securable:

- η $\vec{u}$ is $\mathfrak{B}$-secured.
- ζ $\vec{u} \equiv \vec{v} \frown x$ such that $\vec{v}$ is $\mathfrak{B}$-securable
- $\digamma$ For all immediate refinements x, $\vec{u} \frown x$ is $\mathfrak{B}$-securable

In fact, we can always normalize a proof built from these primitives into one which contains only η and $\digamma$ inferences.

Normalizing securability

We have identified three different ways that $\vec{u}$ could be $\mathfrak{B}$-securable:

- η $\vec{u}$ is $\mathfrak{B}$-secured.
- ζ $\vec{u} \equiv \vec{v} \frown x$ such that $\vec{v}$ is $\mathfrak{B}$-securable
- $\digamma$ For all immediate refinements x, $\vec{u} \frown x$ is $\mathfrak{B}$-securable

In fact, we can always normalize a proof built from these primitives into one which contains only η and $\digamma$ inferences.

Then, every η-inference is replaced with the base case, and every $\digamma$-inference is replaced with the inductive step, obtaining a proof that $\vec{u} \in \mathfrak{A}$.

So what's the problem?

So what's the problem?

The inductive characterization of bar-hood in terms of η, ζ, F is not necessarily the same as the formal/logical definition:

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

So what's the problem?

The inductive characterization of bar-hood in terms of $\eta, \zeta, \digamma$ is not necessarily the same as the formal/logical definition:

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

In fact, if we interpret this statement using propositions-as-types, then it is not the same! There is a procedure to convert a program realizing this statement into a well-founded $\eta, \zeta, \digamma$-tree, but to show that this procedure terminates, we need the bar induction principle already in the metatheory.

So what's the problem?

The inductive characterization of bar-hood in terms of $\eta, \zeta, \digamma$ is not necessarily the same as the formal/logical definition:

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

In fact, if we interpret this statement using propositions-as-types, then it is not the same! There is a procedure to convert a program realizing this statement into a well-founded $\eta, \zeta, \digamma$-tree, but to show that this procedure terminates, we need the bar induction principle already in the metatheory.

So, what would Brouwer say to this?

Brouwerian Realizability: Neighborhood Functions

A point in the spread is an infinite stream of natural numbers. For Brouwer, a function that processes a stream is not a function from streams to results.

$$\mathsf{stream}(\mathbb{N}) \to X \qquad \text{(not this!)}$$

Brouwerian Realizability: Neighborhood Functions

A point in the spread is an infinite stream of natural numbers. For Brouwer, a function that processes a stream is not a function from streams to results.

$$\text{stream}(\mathbb{N}) \rightarrow X \qquad \text{(not this!)}$$

Rather, it is a neighborhood function, a monotonic & continuous function from neighborhoods to partial results:

Brouwerian Realizability: Neighborhood Functions

A point in the spread is an infinite stream of natural numbers. For Brouwer, a function that processes a stream is not a function from streams to results.

$$\mathsf{stream}(\mathbb{N}) \to X \qquad \text{(not this!)}$$

Rather, it is a neighborhood function, a monotonic & continuous function from neighborhoods to partial results:

$$\left\{\phi \in \mathbb{N}^* \to (\mathbb{1} \oplus X) \mid P(\phi)\right\}$$

$$P(\phi) \equiv \forall \alpha \in \mathsf{stream}(\mathbb{N}).\ \exists k \in \mathbb{N}.\ \exists a \in X.\ \forall k' \geq k.\ \phi\,(\bar{\alpha}[k']) \equiv \mathsf{inr}(a)$$

Well-Ordering Neighborhood Functions

$$\left\{\phi \in \mathbb{N}^* \rightarrow (\mathbb{1} \oplus X) \mid P(\phi)\right\}$$

$$P(\phi) \equiv \forall \alpha \in \mathsf{stream}(\mathbb{N}).\ \exists k \in \mathbb{N}.\ \exists a \in X.\ \forall k' \geq k.\ \phi\left(\bar{\alpha}[k']\right) \equiv \mathsf{inr}(a)$$

The condition P on a neighborhood function ϕ induces a well-ordering on ϕ's graph. That is, each such function can be identified with some well-founded dialogue tree.

Neighborhood Functions for Securability

"$\vec{u}$ is $\mathfrak{B}$-securable":

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

Neighborhood Functions for Securability

"$\vec{u}$ is $\mathfrak{B}$-securable":

$$\forall \alpha \in \vec{u}.\ \exists n \in \mathbb{N}.\ \bar{\alpha}[n] \in \mathfrak{B}$$

As far as Brouwer is concerned, the evidence for this statement is a dependent neighborhood function:

$$\prod_{\vec{v} \succcurlyeq \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree.

Neighborhood Functions for Securability

$$\prod_{\vec{v} \geqslant \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree, in particular a $\eta, \zeta, \digamma$-tree.

Neighborhood Functions for Securability

$$\prod_{\vec{v} \geqslant \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree, in particular a η, ζ, F-tree.

- inl$(*)$ corresponds to a F-inference

Neighborhood Functions for Securability

$$\prod_{\vec{v} \geqslant \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree, in particular a η, ζ, F-tree.

- $\mathsf{inl}(*)$ corresponds to a F-inference
- $\mathsf{inr}(0, \mathscr{D})$ corresponds to an η-inference

Neighborhood Functions for Securability

$$\prod_{\vec{v} \geqslant \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree, in particular a η, ζ, F-tree.

- $\mathsf{inl}(*)$ corresponds to a F-inference
- $\mathsf{inr}(0, \mathscr{D})$ corresponds to an η-inference
- $\mathsf{inr}(n+1, \mathscr{D})$ corresponds to an ζ-inference

Neighborhood Functions for Securability

$$\prod_{\vec{v} \succcurlyeq \vec{u}} \mathbb{1} \oplus \sum_{n \in \mathbb{N}} \vec{v}[|\vec{u}| + n] \in \mathfrak{B}$$

By the same technique mentioned before, we can identify any such function with a wellfounded tree, in particular a η, ζ, F-tree.

- $\mathsf{inl}(*)$ corresponds to a F-inference
- $\mathsf{inr}(0, \mathscr{D})$ corresponds to an η-inference
- $\mathsf{inr}(n+1, \mathscr{D})$ corresponds to an ζ-inference

Therefore, the Bar Principle is true under a Brouwerian explanation of the logical connectives!

Status of BI in Type Theory

Status of BI in Type Theory

- There are models of type theory that refute the Bar Principle: recursive realizability (via Church's Thesis), etc.

Status of BI in Type Theory

- There are models of type theory that refute the Bar Principle: recursive realizability (via Church's Thesis), etc.
- The Bar Principle relies on an open-ended notion of stream, i.e. one which does not require that all streams be computed by a recursive function.

Status of BI in Type Theory

- There are models of type theory that refute the Bar Principle: recursive realizability (via Church's Thesis), etc.
- The Bar Principle relies on an open-ended notion of stream, i.e. one which does not require that all streams be computed by a recursive function.
- Adding the Bar Principle to Type Theory is harmless, but restricts the possible models.

Concluding Thoughts

To add the Bar Principle as an axiom to Type Theory is to formalize our intention that Type Theory shall be a semi-formal theory of constructions for Brouwer's mathematics.