Selection Functions, Bar Recursion and Nash Equilibrium

Paulo Oliva

Queen Mary, University of London, UK
(based on joint work with Martín Escardó)

British Logic Colloquium
Swansea, 4 September 2009
Outline

1. Generalised Quantifiers
2. Selection Functions
3. Backward Induction
4. Bar Recursion
Outline

1. Generalised Quantifiers
2. Selection Functions
3. Backward Induction
4. Bar Recursion
Usual quantifiers

$$\exists X, \forall X : (X \rightarrow \mathbb{B}) \rightarrow \mathbb{B}$$
Usual quantifiers ($R = \mathbb{B}$)

$$\exists x, \forall x : (X \rightarrow R) \rightarrow R$$
Usual quantifiers ($R = \mathbb{B}$)

$$\exists_X, \forall_X : (X \to R) \to R$$

Some operations of this type:

<table>
<thead>
<tr>
<th>Operation</th>
<th>$\phi$ : $(X \to R) \to R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantifiers</td>
<td>$\forall_X, \exists_X$ : $(X \to \mathbb{B}) \to \mathbb{B}$</td>
</tr>
<tr>
<td>Integration</td>
<td>$\int_0^1 : ([0, 1] \to \mathbb{R}) \to \mathbb{R}$</td>
</tr>
<tr>
<td>Supremum</td>
<td>$\sup_{[0,1]} : ([0, 1] \to \mathbb{R}) \to \mathbb{R}$</td>
</tr>
<tr>
<td>Limit</td>
<td>$\lim : (\mathbb{N} \to R) \to R$</td>
</tr>
<tr>
<td>Fixed point operator</td>
<td>$\text{fix}_X : (X \to X) \to X$</td>
</tr>
</tbody>
</table>
Definition (Generalised Quantifiers)

Let us call operations $\phi$ of type

$$(X \to R) \to R$$

generalised quantifiers. Write $K_R X \equiv (X \to R) \to R$. 
Definition (Generalised Quantifiers)

Let us call operations $\phi$ of type

$$(X \to R) \to R$$

generalised quantifiers. Write $K_{RX} \equiv (X \to R) \to R$.

Definition (Product of Generalised Quantifiers)

Given quantifiers $\phi : K_{RX}$ and $\psi : K_{RY}$ define the product quantifier $\phi \otimes \psi : K_{R(X \times Y)}$ as

$$(\phi \otimes \psi)(p) : \equiv \phi(\lambda x^X. \psi(\lambda y^Y. p(x, y)))$$

where $p : X \times Y \to R$. 
Generalised Quantifiers

What does

$$(\phi \otimes \psi)(p) \overset{R}{:=} \phi(\lambda x^X. \psi(\lambda y^Y. p(x, y)))$$

mean?
Generalised Quantifiers

What does

\[(\phi \otimes \psi)(p) \equiv \phi(\lambda x^X. \psi(\lambda y^Y. p(x, y)))\]

mean?

Exactly what you would expect, namely

\[(\exists X \otimes \forall Y)(p^{X \times Y \rightarrow \mathbb{B}}) \equiv \exists x^X \forall y^Y p(x, y)\]

\[(\text{sup} \otimes \int)(p^{[0,1]^2 \rightarrow \mathbb{R}}) \equiv \sup_x \int_0^1 p(x, y) dy\]
Outline

1. Generalised Quantifiers
2. Selection Functions
3. Backward Induction
4. Bar Recursion
Selection Functions, Bar Recursion and Nash Equilibrium

Selection Functions

Theorem (Mean Value Theorem)

For any $p \in C[0, 1]$ there is a point $a \in [0, 1]$ such that

$$\int_0^1 p = p(a)$$
Theorem (Mean Value Theorem)

For any \( p \in C[0, 1] \) there is a point \( a \in [0, 1] \) such that

\[
\int_0^1 p = p(a)
\]

Theorem (Maximum Theorem)

For any \( p \in C[0, 1] \) there is a point \( a \in [0, 1] \) such that

\[
\sup p = p(a)
\]
Theorem (Witness Theorem)

For any $p: X \to \mathbb{B}$ there is a point $a \in X$ such that

$$\exists x \in X \ p(x) \iff p(a)$$

(similar to Hilbert's $\varepsilon$-term).
Theorem (Witness Theorem)

For any \( p : X \rightarrow \mathbb{B} \) there is a point \( a \in X \) such that

\[ \exists x^X p(x) \iff p(a) \]

(similar to Hilbert’s \( \varepsilon \)-term).

Theorem (Counter-example Theorem)

For any \( p : X \rightarrow \mathbb{B} \) there is a point \( a \in X \) such that

\[ \forall x^X p(x) \iff p(a) \]

(aka “Drinker’s paradox”).
Let $J_R X :\equiv (X \to R) \to X$. 
Let $J_R X \equiv (X \to R) \to X$.

**Definition (Selection Functions)**

$\epsilon : J_R X$ is called a *selection function* for $\phi : K_R X$ if

$$\phi(p) = p(\epsilon p)$$

holds for all $p : X \to R$. 

**Definition (Attainable Quantifiers)**

A generalised quantifier $\phi : K_R X$ is called *attainable* if it has a selection function $\epsilon : J_R X$. 

Let $J_R X \equiv (X \rightarrow R) \rightarrow X$.

**Definition (Selection Functions)**

$\varepsilon : J_R X$ is called a **selection function** for $\phi : K_R X$ if

$$\phi(p) = p(\varepsilon p)$$

holds for all $p : X \rightarrow R$.

**Definition (Attainable Quantifiers)**

A generalised quantifier $\phi : K_R X$ is called **attainable** if it has a selection function $\varepsilon : J_R X$. 
For Instance

Any fixed point operator

\[ \text{fix} : (X \rightarrow X) \rightarrow X \]

is an attainable quantifier, and a selection function.

In fact, the fixed point equation

\[ \text{fix } p = p(\text{fix } p) \]

says that fix is its own selection function.
A Mapping $J_R \mapsto K_R$

Not all quantifiers are attainable, but every element

$$\varepsilon : J_R X$$

is a selection function for some attainable quantifier, namely

$$\bar{\varepsilon} : K_R X$$

defined as

$$\bar{\varepsilon} p := p(\varepsilon p).$$

So, we call all elements $\varepsilon : JX$ "selection functions".
Questions

Is “being attainable” closed under finite product?
What about countable product?
Questions

Is “being attainable” closed under finite product?

What about countable product?

Yes! We define a product of selection functions such that

$$\bar{\varepsilon} \otimes \bar{\delta} = \bar{\varepsilon} \otimes \bar{\delta}$$
Definition (Product of Selection Functions)

Given selection functions $\varepsilon: J_R X$ and $\delta: J_R Y$ define a product selection function

$$\varepsilon \otimes \delta : J_R (X \times Y)$$

as

$$(\varepsilon \otimes \delta)(p^{X \times Y \to R})^{X \times Y} := (a, b(a))$$

where

$$a := \varepsilon(\lambda x. p(x, b(x)))$$

$$b(x) := \delta(\lambda y. p(x, y)).$$
Product of Selection Functions

\[ p : X \times Y \rightarrow R \]
Product of Selection Functions

\[ p : X \times Y \rightarrow R \]
Product of Selection Functions

\[ p: X \times Y \rightarrow R \]
Product of Selection Functions

\[ p: X \times Y \rightarrow R \]
Quantifier Elimination

Suppose $\exists n \ p(\vec{v}, n) = p(\vec{v}, \varepsilon(\lambda n. p(\vec{v}, n)))$. 
Quantifier Elimination

Suppose $\exists n \, p(\vec{v}, n) = p(\vec{v}, \varepsilon(\lambda n. p(\vec{v}, n)))$. Then

$$\exists x \exists y \, p(x, y) = \exists x \, p(x, b(x))$$

where

$$b(x) = \varepsilon(\lambda y. p(x, y))$$
Quantifier Elimination

Suppose $\exists n \ p(\vec{v}, n) = p(\vec{v}, \varepsilon(\lambda n.p(\vec{v}, n))$. Then

$$\exists x \exists y \ p(x, y) = \exists x \ p(x, b(x)) = p(a, b(a))$$

where

$$b(x) = \varepsilon(\lambda y.p(x, y))$$

$$a = \varepsilon(\lambda x.p(x, b(x))).$$
Suppose $\exists n \ p(\vec{v}, n) = p(\vec{v}, \varepsilon(\lambda n. p(\vec{v}, n)))$. Then

$$\exists x \exists y \ p(x, y) = \exists x \ p(x, b(x))$$

$$= p(a, b(a))$$

where

$$b(x) = \varepsilon(\lambda y. p(x, y))$$

$$a = \varepsilon(\lambda x. p(x, b(x)))$$.

In fact, $(\varepsilon \otimes \varepsilon)(p) = (a, b(a))$. 
Bekic’s lemma

**Lemma**

If $X$ and $Y$ have fixed point operators then so does $X \times Y$. 
Bekic’s lemma

\[ p: X \times Y \to X \times Y \]
Bekic’s lemma

\[ p: X \times Y \rightarrow X \times Y \]
Bekic’s lemma

\( p: X \times Y \rightarrow X \times Y \)

\[ \begin{align*}
\text{fix}_X & \quad p(x, b(x)) \\
\text{fix}_Y & \quad pa(y) \\
\end{align*} \]
Outline

1. Generalised Quantifiers
2. Selection Functions
3. Backward Induction
4. Bar Recursion
Nash equilibrium (simultaneous games)

- $n$ players, each with a set of “strategies” $X_i$
- **payoff function** $f : \prod_{i=0}^{n-1} X_i \rightarrow \mathbb{R}^n$
- **strategy profile** $(x_0, \ldots, x_{n-1}) : \prod_{i=0}^{n-1} X_i$
Nash equilibrium (simultaneous games)

- $n$ players, each with a set of “strategies” $X_i$
- **payoff function** $f : \Pi_{i=0}^{n-1} X_i \rightarrow \mathbb{R}^n$
- **strategy profile** $(x_0, \ldots, x_{n-1}) : \Pi_{i=0}^{n-1} X_i$

- **equilibrium strategy profile** if for $i = 0, \ldots, n - 1$
  $$\forall x_i^*(f_i(x_0, \ldots, x_i^*, \ldots, x_{n-1}) \leq f_i(x_0, \ldots, x_i, \ldots, x_{n-1}))$$
Nash equilibrium (simultaneous games)

- $n$ players, each with a set of “strategies” $X_i$
- payoff function $f : \prod_{i=0}^{n-1} X_i \to \mathbb{R}^n$
- strategy profile $(x_0, \ldots, x_{n-1}) : \prod_{i=0}^{n-1} X_i$

- equilibrium strategy profile if for $i = 0, \ldots, n - 1$
  \[ \forall x_i^* (f_i(x_0, \ldots, x_i^*, \ldots, x_{n-1}) \leq f_i(x_0, \ldots, x_i, \ldots, x_{n-1})) \]

- pure equilibria not always exist, but mixed ones do
- consider, however, that the game is played sequentially
Nash equilibrium (for sequential games)

E.g. three players, payoff function $f : X \times Y \times Z \rightarrow \mathbb{R}^3$
Nash equilibrium (for sequential games)

E.g. three players, payoff function $f : X \times Y \times Z \rightarrow \mathbb{R}^3$

- $f(x_0, y_0, z_0) = (0,1,2)$
- $f(x_0, y_0, z_1) = (2,1,1)$
- $f(x_0, y_1, z_0) = (3,0,2)$
- $f(x_0, y_1, z_1) = (1,3,0)$
- $f(x_1, y_0, z_0) = (0,1,0)$
- $f(x_1, y_0, z_1) = (2,1,1)$
- $f(x_1, y_1, z_0) = (2,2,1)$
- $f(x_1, y_1, z_1) = (3,0,2)$
Nash equilibrium (for sequential games)

E.g. three players, payoff function \( f : X \times Y \times Z \rightarrow \mathbb{R}^3 \)

\[
\begin{align*}
  f(x_0, y_0, z_0) &= (0,1,2) \\
  f(x_0, y_0, z_1) &= (2,1,1) \\
  f(x_0, y_1, z_0) &= (3,0,2) \\
  f(x_0, y_1, z_1) &= (1,3,0) \\
  f(x_1, y_0, z_0) &= (0,1,0) \\
  f(x_1, y_0, z_1) &= (2,1,1) \\
  f(x_1, y_1, z_0) &= (2,2,1) \\
  f(x_1, y_1, z_1) &= (3,0,2)
\end{align*}
\]
Nash equilibrium (for sequential games)

E.g. three players, payoff function $f: X \times Y \times Z \rightarrow \mathbb{R}^3$

$\begin{align*}
f(x_0, y_0, z_0) &= (0,1,2) \\
f(x_0, y_0, z_1) &= (2,1,1) \\
f(x_0, y_1, z_0) &= (3,0,2) \\
f(x_0, y_1, z_1) &= (1,3,0) \\
f(x_1, y_0, z_0) &= (0,1,0) \\
f(x_1, y_0, z_1) &= (2,1,1) \\
f(x_1, y_1, z_0) &= (2,2,1) \\
f(x_1, y_1, z_1) &= (3,0,2)
\end{align*}$
Backward Induction

Selection functions in this case are

\[
\text{argmax}_i(p) \{ \quad \text{[argmax}_i : (X_i \to \mathbb{R}^n) \to X_i] \\
\text{for } (x \in X_i) \text{ do} \\
\quad \text{if } p(x) \text{ has maximal } i\text{-coordinate return } x \\
\}
\]
Selection functions in this case are

\[
\text{argmax}_i(p) \{ \\
\text{for } (x \in X_i) \text{ do} \\
\quad \text{if } p(x) \text{ has maximal } i\text{-coordinate return } x \\
\} \\
\text{[argmax}_i : (X_i \rightarrow \mathbb{R}^n) \rightarrow X_i] \\\n\]

Product

\[
\left( \bigotimes_{i=0}^{n-1} \text{argmax}_i \right) (f)
\]

computes “optimal play”, and can be used to calculate strategy profile in Nash equilibrium.
Outline

1. Generalised Quantifiers
2. Selection Functions
3. Backward Induction
4. Bar Recursion
Bar recursion = infinite product

**Bar recursion** is simply the countable iteration of product of selection functions and quantifiers!
Bar recursion = infinite product

**Bar recursion** is simply the countable iteration of product of selection functions and quantifiers!

In other words, define **infinite product** as

\[ \bigotimes_k (\varepsilon) = \varepsilon_k \otimes \bigotimes_{k+1} (\varepsilon). \]

where \( \varepsilon : \prod_{k \in \mathbb{N}} J_R(X_k). \)
Bar recursion = infinite product

**Bar recursion** is simply the countable iteration of product of selection functions and quantifiers!

In other words, define **infinite product** as

$$\bigotimes_{k} (\varepsilon) = \varepsilon_{k} \otimes \left( \bigotimes_{k+1} (\varepsilon) \right).$$

where $\varepsilon : \Pi_{k \in \mathbb{N}} J_{R}(X_k)$.

Then *(intuitively)*

$$\text{BR}(\varepsilon, p, s) = \bigotimes_{|s|} (\varepsilon)(p_s).$$
Two points

**Point 1.** Infinite products not always (uniquely) defined. Recursive equation uniquely defines a function in the model of *continuous functionals*. But it does not on the *full set theoretic model*. 
Two points

**Point 1.** Infinite products not always (uniquely) defined. Recursive equation uniquely defines a function in the model of *continuous functionals*. But it does not on the *full set theoretic model*.

**Point 2.** There are several variants of bar recursion, but only two binary products have been defined?

- **Product of quant.** $\rightarrow$ Spector BR [Spector’62]
- **Product of s.f.** $\rightarrow$ Course-of-value BR [Escardo/O.’09]
- **Skewed product** $\rightarrow$ Modified BR [Berger/O.’06]
- **Symmetric product** $\rightarrow$ BBC [Berardi et al’98]
Spector Bar recursion

Iterated product of quantifiers

\[ \bigotimes_k (\phi) = \phi_k \otimes (\bigotimes_{k+1} (\phi)) \]

in general fails to exist (even assuming continuity).

Ps.: Actually, Spector uses dependent products – c.f. paper.
Spector Bar recursion

Iterated product of quantifiers

$$\bigotimes_k (\phi) = \phi_k \otimes (\bigotimes_{k+1} (\phi))$$

in general fails to exist (even assuming continuity).

Spector’s original bar recursion corresponds to a “conditional” iterated product

$$\bigotimes_k (\phi)(p) = \begin{cases} p(0) & \text{if } p(0) < k \\
(\phi_k \otimes (\bigotimes_{k+1} (\phi)))(p) & \text{otherwise.} \end{cases}$$

Ps.: Actually, Spector uses dependent products – c.f. paper.
Double negation shift

The double negation shift $\text{DNS}$

$$\forall n \neg \neg A(n) \to \neg \neg \forall n A(n)$$

corresponds to the type

$$\Pi_n ((A_n \to \bot) \to \bot) \to (\Pi_n A_n \to \bot) \to \bot.$$
Double negation shift

The double negation shift DNS

$$\forall n \neg \neg A(n) \rightarrow \neg \neg \forall n A(n)$$

corresponds to the type

$$\Pi_n \left( (A_n \rightarrow \bot) \rightarrow \bot \right) \rightarrow (\Pi_n A_n \rightarrow \bot) \rightarrow \bot.$$  

If $\bot \rightarrow A_n$, this is equivalent to

$$\Pi_n \left( (A_n \rightarrow \bot) \rightarrow A_n \right) \rightarrow (\Pi_n A_n \rightarrow \bot) \rightarrow \Pi_n A_n$$

i.e. $\Pi_n J(A_n) \rightarrow J(\Pi_n A_n)$. 

Double negation shift

The double negation shift $\text{DNS}$

$$\forall n \neg \neg A(n) \rightarrow \neg \neg \forall n A(n)$$

corresponds to the type

$$\Pi_n \left( \left( A_n \rightarrow \bot \right) \rightarrow \bot \right) \rightarrow \left( \Pi_n A_n \rightarrow \bot \right) \rightarrow \bot.$$ 

If $\bot \rightarrow A_n$, this is equivalent to

$$\Pi_n \left( \left( A_n \rightarrow \bot \right) \rightarrow A_n \right) \rightarrow \left( \Pi_n A_n \rightarrow \bot \right) \rightarrow \Pi_n A_n$$
i.e. $\Pi_n J(A_n) \rightarrow J(\Pi_n A_n)$.

The type of the \textbf{countable product} of selection functions!
Not Mentioned but Very Interesting

- Connection to **classical logic**
  Finite product of quantifiers witnesses dialectica interpretation of IPHP

- General notion of **game**
  Optimal strategies as products of selection functions
  History dependent games, dependent products

- Relation to **monads**
  $K, J$ are strong monads, $\varepsilon \mapsto \bar{\varepsilon}$ a monad morphism

- **Interdefinability** between bar recursions
  E.g. “normal” product $=$ “skewed” product
For more information see:

*Selection functions, bar recursion and backward induction*
M. Escardo and P. Oliva, Submitted, July 2009
Preprint available from my webpage.

*Instances of bar recursion as iterated products of selection functions and quantifiers*
M. Escardo and P. Oliva, In preparation.