

# Jeux et Sémantique

Samuel Mimram

Laboratoire PPS, CNRS – Univ. Paris Diderot

Journée Théorie des Jeux et Informatique  
18 février 2009



# Origines de la sémantique des jeux

Précurseurs :

- Lorenzen [Lor61],

*prouvabilité*

$\Leftrightarrow$

*existence d'une stratégie sur le jeu associé*

$\forall x. \exists y. P$

# Origines de la sémantique des jeux

Précurseurs :

- Lorenzen [Lor61],
- Conway, Joyal [Joy77].

Des modèles complets de (fragments de) logique linéaire :

- Abramsky, Jagadeesan [AJ94],
- Hyland, Ong [HO93].

Des modèles pleinement adéquats de PCF :

- Abramsky, Jagadeesan, Malacaria [AJM00],
- Hyland, Ong [HO00, Nic94].

Comment donner une description abstraite de  
*ce que calculent les programmes?*

# Sémantique

sémantique = interprétation mathématique abstraite  
des programmes / des preuves

# Sémantique dénotationnelle

sémantique = interprétation mathématique abstraite  
des programmes / des preuves

sémantique dénotationnelle = invariants du calcul

## Modèle

Modèle d'un langage de programmation :

- un type  $A$  est interprété par un « espace de calcul »  $\llbracket A \rrbracket$ ,
- un programme  $f : A \rightarrow B$  est interprété par une « fonction »  $\llbracket f \rrbracket : \llbracket A \rrbracket \rightarrow \llbracket B \rrbracket$ .

## Modèle dénotationnel

Modèle d'un langage de programmation :

- un type  $A$  est interprété par un « espace de calcul »  $\llbracket A \rrbracket$ ,
- un programme  $f : A \rightarrow B$  est interprété par une « fonction »  $\llbracket f \rrbracket : \llbracket A \rrbracket \rightarrow \llbracket B \rrbracket$ .

$$\begin{array}{ccc} f & \rightsquigarrow & \llbracket f \rrbracket \\ \beta \downarrow & & \parallel \\ f' & \rightsquigarrow & \llbracket f' \rrbracket \end{array}$$



## Modèle dénotationnel

Modèle d'un langage de programmation :

- un type  $A$  est interprété par un « espace de calcul »  $\llbracket A \rrbracket$ ,
- un programme  $f : A \rightarrow B$  est interprété par une « fonction »  $\llbracket f \rrbracket : \llbracket A \rrbracket \rightarrow \llbracket B \rrbracket$ .

$$\begin{array}{ccc} f & \rightsquigarrow & \llbracket f \rrbracket \\ \beta \downarrow & & \parallel \\ f' & \rightsquigarrow & \llbracket f' \rrbracket \end{array}$$

Sémantiques de jeux :

- les types sont interprétés par des **jeux**,
- les programmes sont interprétés par des **stratégies**.

## Sémantiques de jeux

Les sémantiques de jeux modélisent *le comportement interactif* des programmes.

```
fun x → not x   :   int ⇒ int
```

```
(fun x → not x)true   ⇔   false  
(fun x → not x>false  ⇔   true
```

# Le langage PCF

PCF est un langage

- fonctionnel,
- avec des valeurs de base : booléens et entiers,
- avec des constructions pour manipuler ces valeurs :  
`if then else`, `+`, etc.
- avec une construction de point fixe,
- simplement typé.

## Un modèle pleinement adéquat de PCF

Deux termes  $M$  et  $N$  sont **extensionnellement équivalents** ( $M \approx N$ ) lorsque pour tout contexte  $C[-]$ ,

$$C[M] \Downarrow v \quad \text{ssi} \quad C[N] \Downarrow v$$

## Un modèle pleinement adéquat de PCF

Deux termes  $M$  et  $N$  sont **extensionnellement équivalents** ( $M \approx N$ ) lorsque pour tout contexte  $C[-]$ ,

$$C[M] \Downarrow v \quad \text{ssi} \quad C[N] \Downarrow v$$

Modèle **pleinement adéquat** de PCF :

$$M \approx N \quad \text{ssi} \quad \llbracket M \rrbracket = \llbracket N \rrbracket$$

## Définissabilité

Un élément  $f : \llbracket A \rrbracket$  est **définissable** lorsqu'il existe  $M : A$  tel que

$$f = \llbracket M \rrbracket$$

# Sémantiques de jeux HO

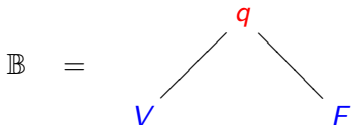
## Les jeux

Une arène

$$(M, \vdash, \lambda, \lambda')$$

est constituée

- d'un ensemble de **coups**  $M$
- d'une relation binaire  $\vdash$  exprimant les **dépendances causales**
- d'une **polarisation** des coups  $\lambda : M \rightarrow \{O, P\}$
- d'une fonction  $\lambda' : M \rightarrow \{Q, A\}$



# Sémantiques de jeux HO

Les jeux

Le jeu  $\mathbb{B} \times \mathbb{B}$  :

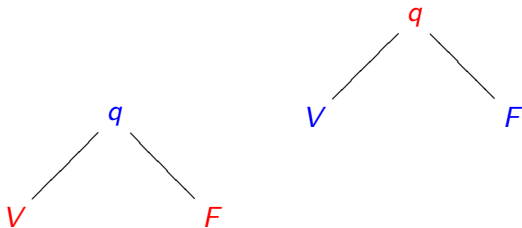




# Sémantiques de jeux HO

## Les jeux

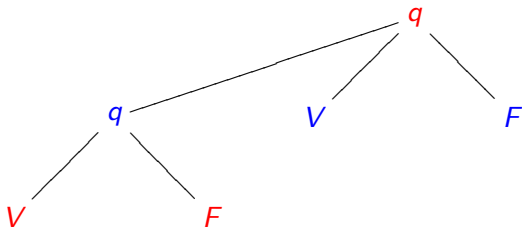
Le jeu  $\mathbb{B} \Rightarrow \mathbb{B}$  :



# Sémantiques de jeux HO

Les jeux

Le jeu  $\mathbb{B} \Rightarrow \mathbb{B}$  :



# Sémantiques de jeux HO

## Les stratégies

Une **partie**  $s$  dans une arène  $A$  est une suite pointée de coups

$$s = m_1 \cdots m_i \cdots m_j \cdots m_{2n}$$

- alternée,
- de longueur paire,
- telle que si  $m_j$  pointe sur  $m_i$  alors  $m_i \vdash_A m_j$ .

# Sémantiques de jeux HO

## Les stratégies

Une **partie**  $s$  dans une arène  $A$  est une suite pointée de coups

$$s = m_1 \cdots m_i \cdots m_j \cdots m_{2n}$$

- alternée,
- de longueur paire,
- telle que si  $m_j$  pointe sur  $m_i$  alors  $m_i \vdash_A m_j$ .

Une **stratégie** est un ensemble non vide de parties, clos par préfixe de longueur paire.

## Sémantiques de jeux HO

La stratégie not :

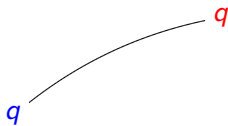
$$\mathbb{B} \Rightarrow \mathbb{B}$$

$$\{ \varepsilon \}$$

# Sémantiques de jeux HO

La stratégie not :

$\mathbb{B} \Rightarrow \mathbb{B}$

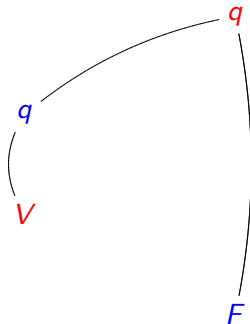


$\{ \varepsilon, \quad q \cdot q \}$

# Sémantiques de jeux HO

La stratégie not :

$\mathbb{B} \Rightarrow \mathbb{B}$

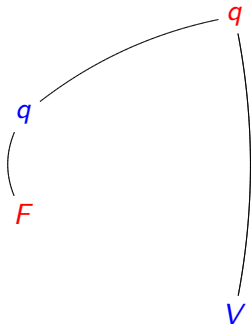


$\{ \varepsilon, \quad q \cdot q, \quad q \cdot q \cdot V \cdot F \}$

# Sémantiques de jeux HO

La stratégie not :

$\mathbb{B} \Rightarrow \mathbb{B}$

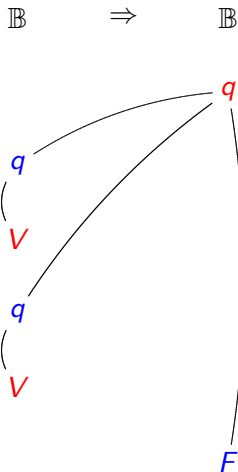


$\{ \varepsilon, \quad q \cdot q, \quad q \cdot q \cdot V \cdot F, \quad q \cdot q \cdot F \cdot V \}$



## Sémantiques de jeux HO

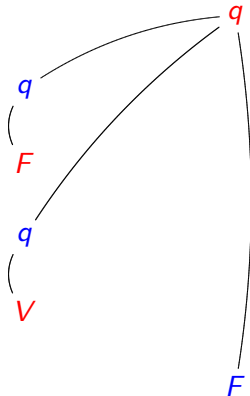
`fun x → if x then not x else not x`



# Sémantiques de jeux HO

fun  $x \rightarrow$  if  $x$  then not  $x$  else not  $x$

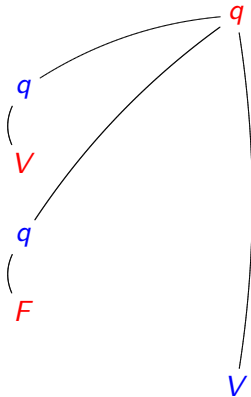
$\mathbb{B} \Rightarrow \mathbb{B}$



## Sémantiques de jeux HO

`fun x → if x then not x else not x`

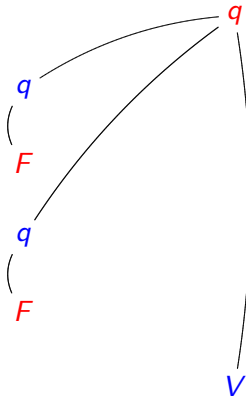
$\mathbb{B} \quad \Rightarrow \quad \mathbb{B}$



## Sémantiques de jeux HO

`fun x → if x then not x else not x`

$\mathbb{B} \quad \Rightarrow \quad \mathbb{B}$



## Catégories de jeux

Les stratégies ne cherchent pas à gagner.

En revanche, on veut avoir une notion de *composition* qui reflète celle du langage modélisé.

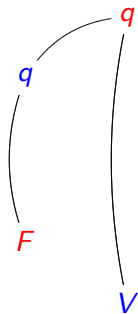
# Composition

## Composition parallèle

Composition de not avec not :

$\mathbb{B} \longrightarrow \mathbb{B}$

$\mathbb{B} \longrightarrow \mathbb{B}$



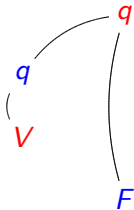
# Composition

## Composition parallèle

Composition de not avec not :

$\mathbb{B} \longrightarrow \mathbb{B}$

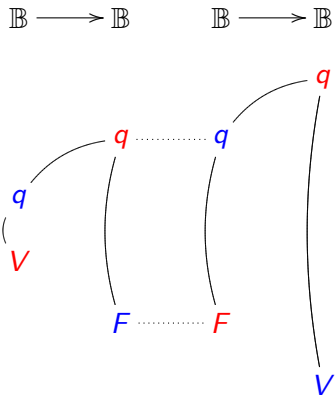
$\mathbb{B} \longrightarrow \mathbb{B}$



# Composition

## Composition parallèle

Composition de not avec not :

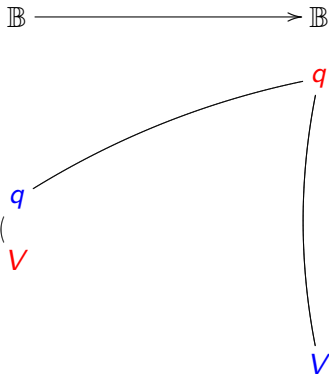




# Composition

Composition parallèle + masquage.

Composition de not avec not :

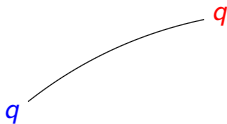


## Stratégie identité (copycat)

$$\mathbb{B} \Rightarrow \mathbb{B}$$

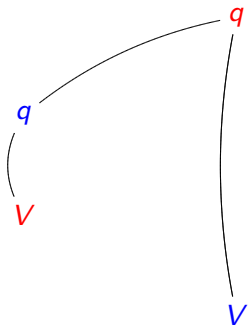
## Stratégie identité (copycat)

$\mathbb{B} \Rightarrow \mathbb{B}$



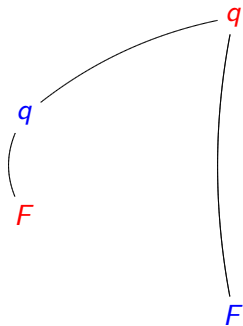
## Stratégie identité (copycat)

$\mathbb{B} \Rightarrow \mathbb{B}$



## Stratégie identité (copycat)

$\mathbb{B} \Rightarrow \mathbb{B}$



Comment caractériser les stratégies définissables  
par des termes de PCF ?

# Déterminisme

## Définition

Une stratégie  $\sigma : A$  est **déterministe** lorsque

$$s \cdot m \in \sigma \quad \text{et} \quad s \cdot m' \in \sigma \quad \text{implique} \quad m = m'$$

# Déterminisme

## Définition

Une stratégie  $\sigma : A$  est **déterministe** lorsque

$$s \cdot m \in \sigma \quad \text{et} \quad s \cdot m' \in \sigma \quad \text{implique} \quad m = m'$$

Contre-exemple : la stratégie  $\sigma : \mathbb{B}$  définie par

$$\sigma = \{ \varepsilon, q \cdot F, q \cdot V \}$$



## Définition

Une stratégie  $\sigma : A$  est **innocente** lorsqu'elle joue en n'ayant qu'une mémoire partielle de son passé : sa *vue*.

$$s \cdot m \in \sigma \quad \text{et} \quad \ulcorner s \urcorner = \ulcorner t \urcorner \quad \text{implique} \quad t \cdot m \in \sigma$$

# Innocence

1<sup>er</sup> contre-exemple

`fun f → ... f ...`

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$

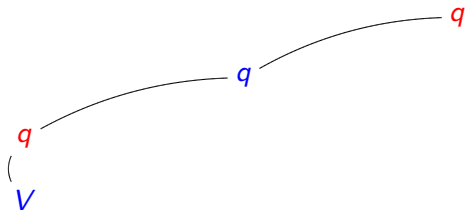


# Innocence

1<sup>er</sup> contre-exemple

`fun f → ... f true ...`

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$

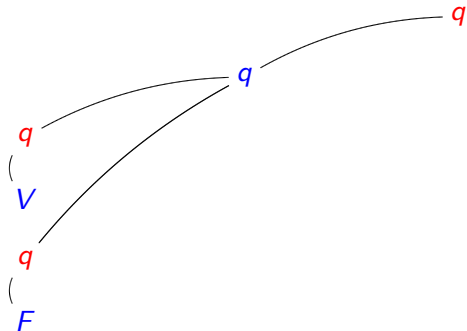


# Innocence

1<sup>er</sup> contre-exemple

`fun f → ... f ? ...`

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$

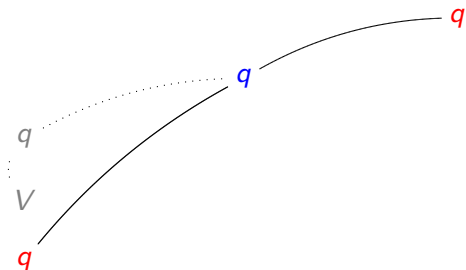


# Innocence

1<sup>er</sup> contre-exemple

`fun f → ... f ? ...`

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Innocence

2<sup>e</sup> contre-exemple

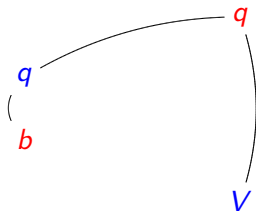
$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Innocence

2<sup>e</sup> contre-exemple

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Innocence

2<sup>e</sup> contre-exemple

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$

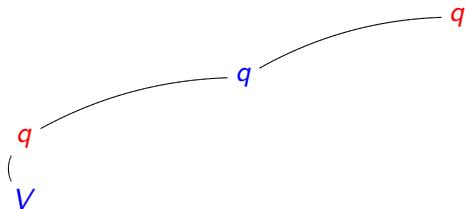




# Innocence

2<sup>e</sup> contre-exemple

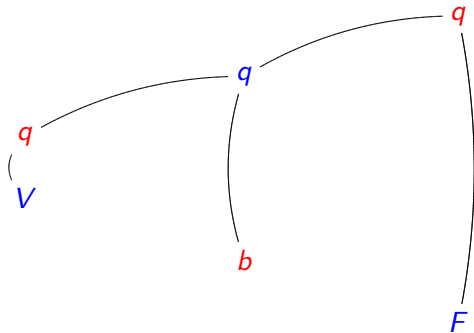
$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Innocence

2<sup>e</sup> contre-exemple

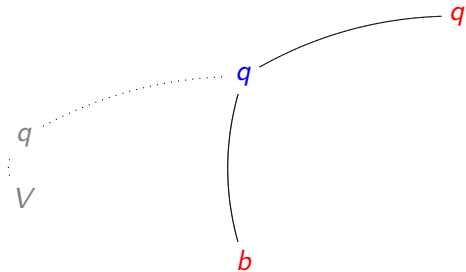
$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Innocence

2<sup>e</sup> contre-exemple

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



## Définition

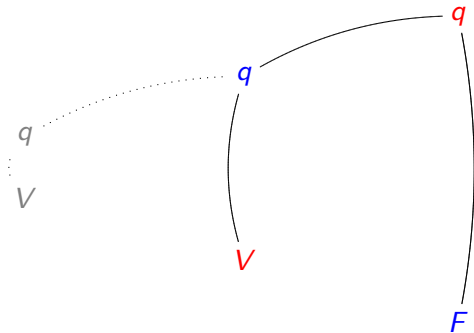
Pour toute partie  $s$ , la **vue**  $\lceil s \rceil$  de  $s$  est définie inductivement par

$$\begin{aligned} \lceil s \cdot m \rceil &= \lceil s \rceil \cdot m && \text{si } m \text{ est un coup Joueur} \\ \lceil s \cdot m \cdot t \cdot n \rceil &= \lceil s \cdot m \rceil \cdot n && \text{si } n \text{ est un coup Opposant pointant sur } m \\ \lceil s \cdot m \rceil &= m && \text{si } m \text{ est un coup Opposant initial} \end{aligned}$$

# Innocence

2<sup>e</sup> contre-exemple

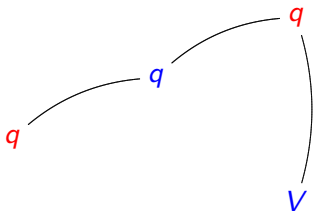
$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



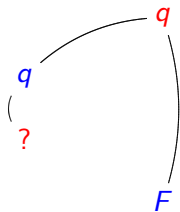
# Catch

La stratégie catch :

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



# Bon parenthésage

## Définition

Une stratégie  $\sigma : A$  est **bien parenthésée** si toute réponse  $m$  du Joueur dans une partie  $s \cdot m$  pointe vers la question pendante de  $\lceil s \rceil$ .

## Théorème (HON 94)

*Le modèle des stratégies déterministes, innocentes et bien parenthésées est pleinement adéquat pour PCF.*



# Parenthésage et innocence

Stratégies

visibles + déterministes + bien parenthésées + innocentes

$\approx$

programme PCF

# Parenthésage et innocence

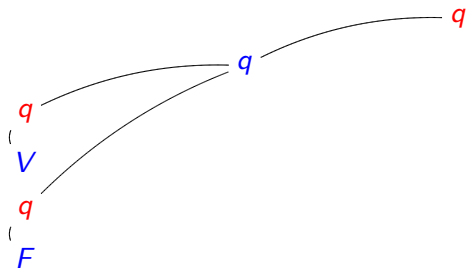
## Stratégies

visibles + déterministes + bien parenthésées

$\approx$

programme PCF + références

$(\mathbb{B} \Rightarrow \mathbb{B}) \Rightarrow \mathbb{B}$



```
fun f → x := 0 ; ... f (incr x ; x == 1) ...
```

# Parenthésage et innocence

Stratégies

visibles + déterministes + bien parenthésées + innocentes

$\approx$

programme PCF

# Parenthésage et innocence

## Stratégies

visibles + déterministes + bien parenthésées + innocentes

$\approx$

$\lambda$ -termes simplement typés = preuve intuitionniste

# Parenthésage et innocence

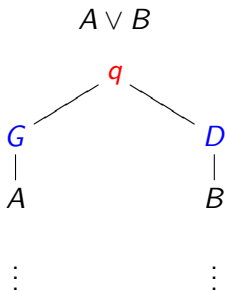
Stratégies

visibles + déterministes

+ innocentes

$\approx$

$\lambda\mu$ -termes simplement typés = preuve classique



# Parenthésage et innocence

Stratégies

visibles + déterministes

+ innocentes

$\approx$

$\lambda\mu$ -termes simplement typés = preuve classique

$\perp$

$q\perp$

# Parenthésage et innocence

Stratégies

visibles + déterministes

+ innocentes

$\approx$

$\lambda\mu$ -termes simplement typés = preuve classique

$$\neg A = A \Rightarrow \perp$$

$q\perp$

|

$A^*$

$\vdots$

# Parenthésage et innocence

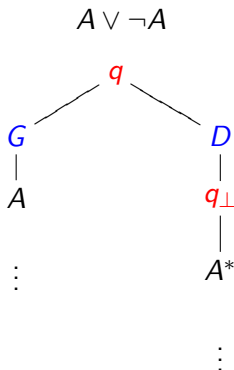
Stratégies

visibles + déterministes

+ innocentes

$\approx$

$\lambda\mu$ -termes simplement typés = preuve classique





# Parenthésage et innocence

Stratégies

visibles + déterministes

+ innocentes

$\approx$

programme PCF + contrôle

# Parenthésage et innocence

Stratégies

visibles + déterministes

$\approx$

programme PCF + contrôle + références

# Applications





- permet de mieux comprendre le comportement interactif des programmes et des preuves,

# Applications

- permet de mieux comprendre le comportement interactif des programmes et des preuves,
- model-checking et interprétation abstraite :  
[GM03] lorsque les types sont au plus du second ordre,
  - les pointeurs ne sont pas nécessaires,
  - les stratégies peuvent être décrites par des langages réguliers,
  - l'équivalence est décidable sur un alphabet fini. . .

Merci !

## Bibliographie I

-  S. Abramsky and R. Jagadeesan.  
Games and Full Completeness for Multiplicative Linear Logic.  
*The Journal of Symbolic Logic*, 59(2) :543–574, 1994.
-  S. Abramsky, R. Jagadeesan, and P. Malacaria.  
Full abstraction for PCF.  
*Information and Computation*, 163(2) :409–470, 2000.
-  D.R. Ghica and G. McCusker.  
The regular-language semantics of second-order idealized  
ALGOL.  
*Theoretical Computer Science*, 309(1-3) :469–502, 2003.
-  J.M.E. Hyland and C.H.L. Ong.  
Fair games and full completeness for multiplicative linear logic  
without the mix-rule.  
*preprint*, 190, 1993.

## Bibliographie II



M. Hyland and L. Ong.

On Full Abstraction for PCF : I, II and III.

*Information and Computation*, 163(2) :285–408, December 2000.



A. Joyal.

Remarques sur la théorie des jeux à deux personnes.

*Gazette des Sciences Mathématiques du Québec*, 1(4) :46–52, 1977.



P. Lorenzen.

Ein dialogisches Konstruktivitätskriterium.

*Infinitistic Methods*, pages 193–200, 1961.

## Bibliographie III



H. Nickau.

Hereditarily sequential functionals.

In A. Nerode and Yu. V. Matiyasevich, editors, *Proceedings of the Symposium on Logical Foundations of Computer Science : Logic at St. Petersburg*, volume 813 of *Lecture Notes in Computer Science*, pages 253–264. Springer Verlag, 1994.