# Game Comonads

## A new language for complexity theory

**Adam Ó Conghaile, University of Cambridge**

# Talk Outline

- The two most important problems in Complexity Theory
- Game Comonads: a new framework for thinking about these problems
- Oracles, Quantifiers and how my work improves game comonads

# A quick introduction to my field

# The Two Most Important Problems

| | Constraint Satisfaction Problem | Graph Isomorphism Problem |
|---|---|---|
| | Is there a homomorphism? | Is there an isomorphism? |
| | $\mathscr{X} \to \mathscr{D}\,?$ | $\mathscr{G} \cong \mathscr{H}\,?$ |
| Uses | SAT-solvers, Sudoku querying databases | Verification, code optimisation, pattern recognition |
| Complexity | Either **P** or **NP-Complete** (Bulatov & Zhuk, 2018) | Not known to be **P** or **NP-Complete** Suspected "intermediate problem" |
| **P-Time** Approximations | $k$-local consistency algorithm | $k$-Weisfeiler-Lehman algorithm |

My work

# Logic is the key to understanding these algorithms

$k$-local consistency algorithm

$$\mathscr{X} \to_k \mathscr{D} \qquad \Longleftrightarrow \qquad \mathscr{X} \Rrightarrow_{\exists^+\mathscr{L}^k} \mathscr{D}\,?$$

Kolaitis & Vardi, 1992

$k$-Weisfeiler-Lehman algorithm

$$\mathscr{G} \cong_k \mathscr{H} \qquad \Longleftrightarrow \qquad \mathscr{G} \equiv_{\mathscr{L}^k(\sharp)} \mathscr{H}$$

Immerman & Lander, 1990

# Games are the key to understanding logic

$$\mathscr{X} \to_k \mathscr{D} \qquad \Longleftrightarrow \qquad \mathscr{X} \Rrightarrow_{\exists^+\mathscr{L}^k} \mathscr{D}\,?$$

Theorem (Kolaitis & Vardi, 1992)

"Duplicator" has a winning strategy for $\exists\mathrm{Peb}_k(\mathscr{X}, \mathscr{D})$ if and only if $\mathscr{X} \Rrightarrow_{\exists^+\mathscr{L}^k} \mathscr{D}$

$\exists\mathrm{Peb}_k(\mathscr{X}, \mathscr{D})$

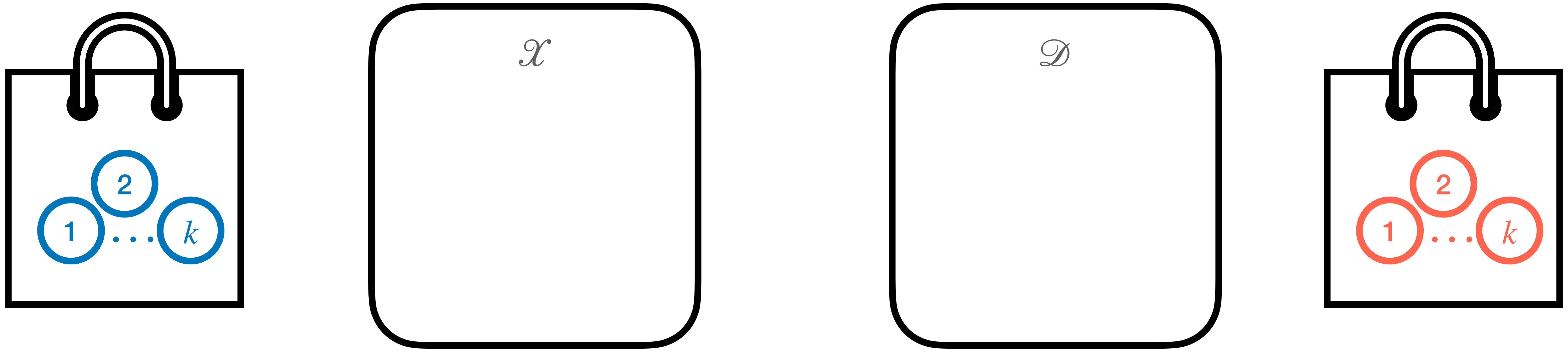
Spoiler

Spoiler wants to convince Duplicator that $\mathscr{X} \nrightarrow \mathscr{D}$

Duplicator wants to convince Spoiler that $\mathscr{X} \to \mathscr{D}$

…but they have limited access to $\mathscr{X}$ and $\mathscr{D}$

Duplicator



# Games are the key to understanding logic

$$\mathscr{G} \cong_k \mathscr{H} \quad \iff \quad \mathscr{G} \equiv_{\mathscr{L}^k(\sharp)} \mathscr{H}$$

Theorem (Hella, 1996)

Duplicator has a winning strategy for $\mathrm{Bij}_k(\mathscr{A}, \mathscr{B})$ if and only if $\mathscr{A} \equiv_{\mathscr{L}^k(\sharp)} \mathscr{B}$

$\mathrm{Bij}_k(\mathscr{A}, \mathscr{B})$

Spoiler

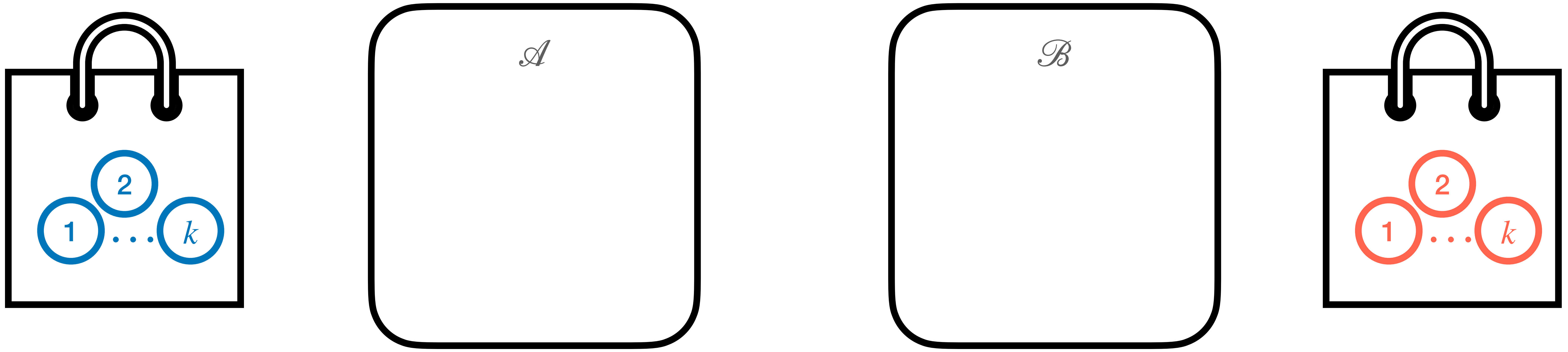
Spoiler wants to convince Duplicator that $\mathscr{A} \ncong \mathscr{B}$

Duplicator wants to convince Spoiler that $\mathscr{A} \ncong \mathscr{B}$

…they have limited access to $\mathscr{X}$ and $\mathscr{D}$

**and** Duplicator has to give a one-to-one map from A to B containing her moves

Duplicator



# Game Comonads are the key to understanding these games

**Research on Constraint Satisfaction**

$\mathscr{X} \to_k \mathscr{D} \iff \mathscr{X} \Rrightarrow_{\exists^+\mathscr{L}^k} \mathscr{D} \iff$ Duplicator wins $\exists\mathrm{Peb}_k(\mathscr{X}, \mathscr{D})$

**Research on Graph Isomorphism**

$\mathscr{G} \cong_k \mathscr{H} \iff \mathscr{G} \equiv_{\mathscr{L}^k(\sharp)} \mathscr{H} \iff$ Duplicator wins $\mathrm{Bij}_k(\mathscr{G}, \mathscr{H})$

**Missing Link?**

**“Pebbling” Comonad $\mathbb{P}_k$ (Abramsky, Dawar & Wang, 2018)**

$\mathbb{P}_k$ is a special type of functor which takes a structure $\mathscr{A}$ and returns a new structure $\mathbb{P}_k\mathscr{A}$ such that:

There is a homomorphism $\mathbb{P}_k\mathscr{A} \to \mathscr{B} \iff$ Duplicator wins $\exists\mathrm{Peb}_k(\mathscr{A}, \mathscr{B})$

There is an isomorphism $\mathbb{P}_k\mathscr{A} \cong \mathbb{P}_k\mathscr{B} \iff$ Duplicator wins $\mathrm{Bij}_k(\mathscr{A}, \mathscr{B})$

There is a “coalgebra” $\mathscr{A} \to \mathbb{P}_k\mathscr{A} \iff \mathscr{A}$ has a "tree decomposition" of width $k$

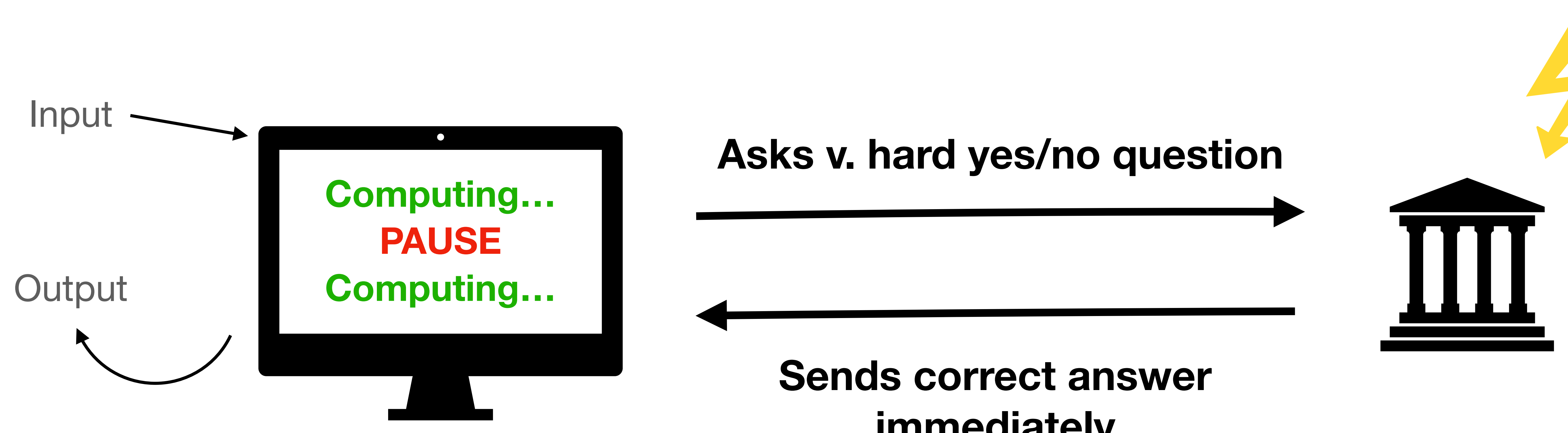
Other comonads have since been discovered for other pairs of logic games modelling other algorithms

# Limitations of the game comonads

- **Lack of computational power:**
  They only capture $\exists^+\mathscr{L}^k$ and $\mathscr{L}^k(\sharp)$, which are not the cutting edge for approximating CSP and GI
- **Only simple "resources":**
  The $k$ in $\mathbb{P}_k$ controls the number of variables (other variants control depth of quantification) but do not control of quantifiers.
- **Ad-hoc constructions:**
  Each new game comonad had to be "engineered" from first principles, no "shortcuts"

# My work on game comonads and quantifiers

# Need more power? Consult an oracle!




Oracle computation exists everywhere in computer science, cryptography and complexity theory (and Ancient Greece!)

In the world of logic, oracles are added using "generalised quantifiers" (due to Per Lindstrom)

Some work had already been done (by Hella) giving a two-way game for logics extended by these oracles.

$$\text{Duplicator wins } \mathrm{Bij}_n^k(\mathscr{A}, \mathscr{B}) \iff \mathscr{A} \equiv_{\mathscr{L}^k(\mathbf{Q}_n)} \mathscr{B}$$

# Improving our understanding of these oracles

$$
\begin{tikzcd}[column sep=small]
 & \mathbf{Bij}^k_n \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
\mathbf{Inj}^k_n \arrow[d, dash] \arrow[dr, dash] & +\mathbf{Bij}^k_n \arrow[dl, dash] \arrow[dr, dash] & \mathbf{Surj}^k_n \arrow[dl, dash] \arrow[d, dash] \\
+\mathbf{Inj}^k_n \arrow[dr, dash] & \mathbf{Fun}^k_n \arrow[d, dash] & +\mathbf{Surj}^k_n \arrow[dl, dash] \\
 & +\mathbf{Fun}^k_n &
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
 & \mathcal{L}^k(\mathcal{Q}^{\mathrm{b}}_n) \arrow[dl, dash] \arrow[d, dash] \arrow[dr, dash] & \\
\mathcal{L}^k(\mathcal{Q}^{\mathrm{i}}_n) \arrow[d, dash] \arrow[dr, dash] & +\mathcal{L}^k(\mathcal{Q}^{\mathrm{b}}_n) \arrow[dl, dash] \arrow[dr, dash] & \mathcal{L}^k(\mathcal{Q}^{\mathrm{s}}_n) \arrow[dl, dash] \arrow[d, dash] \\
+\mathcal{L}^k(\mathcal{Q}^{\mathrm{i}}_n) \arrow[dr, dash] & \mathcal{L}^k(\mathcal{Q}^{\mathrm{h}}_n) \arrow[d, dash] & +\mathcal{L}^k(\mathcal{Q}^{\mathrm{s}}_n) \arrow[dl, dash] \\
 & +\mathcal{L}^k(\mathcal{Q}^{\mathrm{h}}_n) &
\end{tikzcd}
$$

All “n-ary” quantifiers
(including $\sharp$ for n = 1)

“n-ary” hom-closed quantifiers
(including $\exists$ for n = 1)

Theorem 15 (Ó C. & Dawar, 2021)

For a game $\mathscr{G}$ from the left-hand diagram, Duplicator wins $\mathscr{G}(\mathscr{A}, \mathscr{B})$ if and only if $\mathscr{A} \Rrightarrow_{\mathscr{L}^{\mathscr{G}}} \mathscr{B}$ where $\mathscr{L}^{\mathscr{G}}$ is the corresponding logic from the right-hand diagram

# Constructing a new comonad from an old one

<u>Pebbling Comonad</u>

$\mathbb{P}_k\mathscr{A} \to \mathscr{B} \iff \exists\mathsf{Peb}_k(\mathscr{A},\mathscr{B}) \iff \mathscr{A} \Rrightarrow_{\exists^+\mathscr{L}^k} \mathscr{B}$

$\mathbb{P}_k\mathscr{A} \cong \mathbb{P}_k\mathscr{B} \iff \mathsf{Bij}_k(\mathscr{A},\mathscr{B}) \iff \mathscr{A} \equiv_{\mathscr{L}^k(\sharp)} \mathscr{B}$

<u>New “oracle” Comonad</u>

$\mathbb{G}_{n,k}\mathscr{A} := \mathbb{P}_k\mathscr{A} / \approx_n$

$+\mathsf{Fun}_n^k(\mathscr{A},\mathscr{B}) \iff \mathscr{A} \Rrightarrow_{+\mathscr{L}^k(\mathbf{Q}_n^h)} \mathscr{B} \iff \mathbb{G}_{n,k}\mathscr{A} \to \mathscr{B}$

$\mathsf{Bij}_n^k(\mathscr{A},\mathscr{B}) \iff \mathscr{A} \equiv_{\mathscr{L}^k(\mathbf{Q}_n)} \mathscr{B} \iff \mathbb{G}_{n,k}\mathscr{A} \cong \mathbb{G}_{n,k}\mathscr{B}$

Lemma 20 (Ó C. & Dawar, 2021)

Duplicator has a winning strategy for $+\mathsf{Fun}_n^k(\mathscr{A},\mathscr{B})$ if and only if she has an “$n$-consistent” winning strategy for $\exists\mathsf{Peb}_k(\mathscr{A},\mathscr{B})$

Then defined $\approx_n$ a relation on any $\mathbb{P}_k\mathscr{A}$ such that

$\mathbb{P}_k\mathscr{A} / \approx_n \to \mathscr{B} \iff$ Duplicator wins $\exists\mathsf{Peb}_k(\mathscr{A},\mathscr{B})$ n-consistently

Improvements to

# ~~Limitations of~~ the game comonads

- **Lack of computational power:** $\mathbb{G}_{n,k}$ allows us to reason about oracle power in comonads!
  They only capture $\exists^+\mathscr{L}^k$ and $\mathscr{L}^k(\sharp)$, which are not the cutting edge for approximating CSP and GI
- **Only simple "resources":** $\mathbb{G}_{n,k}$ controls multiple different classes of quantifier as well as variables
  The $k$ in $\mathbb{P}_k$ controls the number of variables (other variants control depth of quantification) but not control of quantifiers.
- **Ad-hoc constructions:** $\mathbb{G}_{n,k}$ was built from $\mathbb{P}_k$ opening up the possibility of new constructions
  Each new game comonad had to be "engineered" from first principles, no "shortcuts"

**Full details in my latest publication:**

Ó Conghaile, Dawar
*Game comonads & generalised quantifiers*
Proceedings of CSL 2021

**Future Work**

- Incorporate new SoA approximations
- Implement game comonads in Haskell
- More connections Logic <-> Algorithms