Modelling Economic Game Theory with Haskell 'n String Diagrams

Semantics of Economics for Economics

Viktor Winschel
viktor.winschel@gmail.com

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

HaskellerZ, Zürich
25^{th} February, 2016

Semantics of Economics

- **Foundations**
  Social Science / Economics $\simeq$ *Computer Science*
- **Code**
  String Diagrams $\overset{de/compile}{\longleftrightarrow}$ *GameEDSL.hs*
- **StartUps**
  Education, Consulting, Financial, Monetary
  Energy, Social Engine

Please allow me to introduce myself

- **Coder (1980-)**$(x = x + 1)$
  play only selfcoded games, Atari 800XL / C64
  Assembly language, code that implements Basic
- **StartUp, Economics (1990-)(units, household)**
  Movie company, double accounting
- **MA Economics (-2000)(What is money?)**
  self.referential.xls.cell solves Black-Scholes for investm. bank
- **PhD Economics (-2005)(program me, the economist)**
  Optimal currency areas, countries suitable for mon.union
  Computational MacroEconom(etr)ics, Numerics.fortran
- **PostDoc PolyScience (-2010)(meta circular interpreter)**
  AI, linguistics, philosophy, math, logic, computer science
  Symbolics.Mathematica.Prolog.Lisp
- **PostDoc Reflexive Economics (2010-)**$(a = 1 : a, t(f) = f)$
  Logics for Social Behaviour 3, ETH, 2016
  play only selfcoded economies

But what's puzzling you, is the nature of my game

Foundations of Computer Science

- **Reflexivity**

  Code is data, de/compiler

  Universal Turing machine, run machine by machine

  Russell paradox $S = \{S\}$*, type theory*

  λ*-calculus,* $D \cong [D \to D]$

  Coalgebra, $X \to F(X)$

  Polymorphism, category theory, parametricity

- **Compositionality (Modularity, Synthesis)**

  Behaviour of whole from behaviour of parts

  $[\![S_1 \otimes S_2]\!] = [\![S_1]\!] \otimes [\![S_2]\!]$

  Behaviour of code from operators

  Behaviour of group from individuals

Useful for Economics

Foundations of Economics

- **Reflexivity**

  Theory ↔ model, syntax ↔ semantics
  Context ↔ content, modeller in modelled
  Expectation dynamics, self-fulfilling / defeating prophecies
  Egoism (decentralized control), altruism (utility for utility)
  George Soros, act = participate(data), data = observe(act)
  Algebra $F(X) \to X$, coalgebra $X \to F(X)$
  Lucas critique, learning changes the learned
  Networks of networks, endogenous network formation

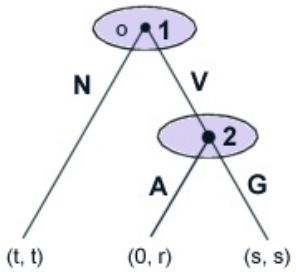
- **Compositionality**

  Behaviour of society from individual (and vice ɐsɹəʌ)
  Modelling in groups, modelling of groups
  Emergent properties, semantics of economics
  Model is code, economic knowledge production industry
  What is Lego of economics, what are the combinators?

Useful for Computer Science

Classical Economic Game Theory

- **Data Types**



$\mathcal{G} = (P = \{1, 2\}, A = \{N, V, A, G\},$
$successor, player, actions, leafs, utility)$
$successor : n \to n, player : n \to P, actions :<> + - ., [] \to A$

- **Mathematics**

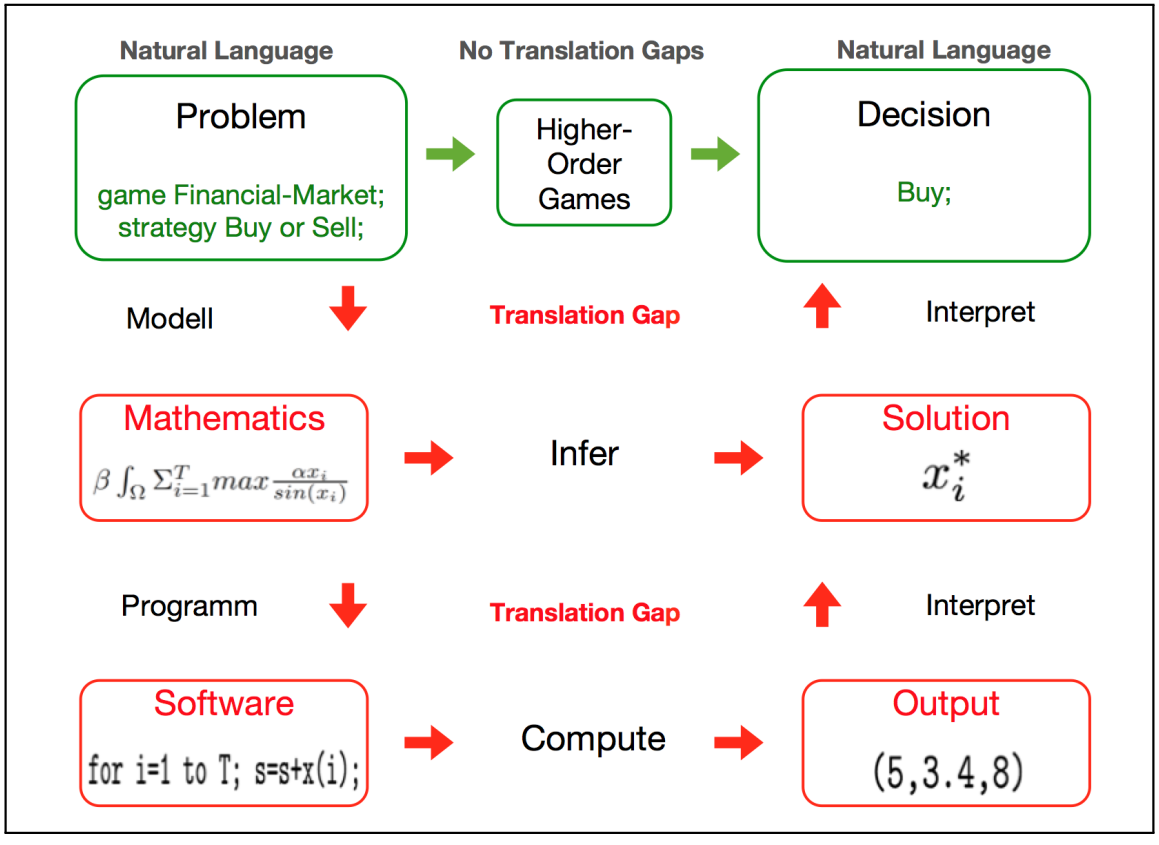
$\sim$ *functional analysis,* $\max_a u(a), du/da = 0$
Utility $\mathbb{R}$*, non computable*
(almost) no automata, ADT, co/algebras, ..., cats

- **Code**

Object oriented Matlab, LOC.m $\preceq$ *LOC / 100.hs*
Linear algebra for co/recursive functions/data types???

Not Reflexive, Compositional, Implementable

Needed: Language for Economics



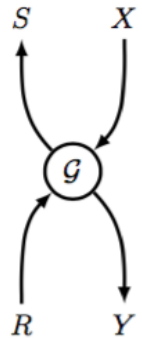
Higher-Order Decision and Game Theory

- **Classical (low level language)**
  Goal: max utility: $max : (X \to R) \to R$
  Act: $argmax : (X \to R) \to P(X)$
- **Higher-Order (higher level language)**
  Context: $p : X \to R$
  Goal: quantifier: $\varphi : (X \to R) \to R$
  Act: selection function: $\varepsilon : (X \to R) \to P(X)$
- **Unifies, Allows**
  Fix point goals: $\varphi_{fix} : (X \to X) \to X$, *Keynes beauty contest*
  Reflexive, coordination, differentiation
  Centralization (egoism), decentralization (altruism)
  Behavioural econ., process orientation, environmental econ, ...
  Model is code, embedded domain specific language (EDSL)

Reflexive, Compositional, Implementable

Category of Pregames

- **Types::objects, pregames::morphisms**



Observation X, *action* Y, *utility* R, *coutility* S
Pregames $\mathcal{G} : X \otimes S^* \to Y \otimes R^*$, *monoidal category*

- **Pregame**

$\mathcal{G} = (\Sigma_\mathcal{G}, \mathbb{P}_\mathcal{G}, \mathbb{C}_\mathcal{G}, \mathbb{E}_\mathcal{G})$ *where*
  - $\Sigma_\mathcal{G}$ *is the set of strategy profiles of* $\mathcal{G}$
  - $\mathbb{P}_\mathcal{G} : \Sigma_\mathcal{G} \times X \to Y$ *is the play function of* $\mathcal{G}$*; choose* $\mathbb{P}_\mathcal{G}(\sigma, x) \in Y$
  - $\mathbb{C}_\mathcal{G} : \Sigma_\mathcal{G} \times X \times R \to S$ *is coplay function*
  - $\mathbb{E}_\mathcal{G} : X \times (Y \to R) \to \mathcal{P}(\Sigma_\mathcal{G})$ *is equilibrium function*
    continuation $k : X \to Y$

Semantics for Economics

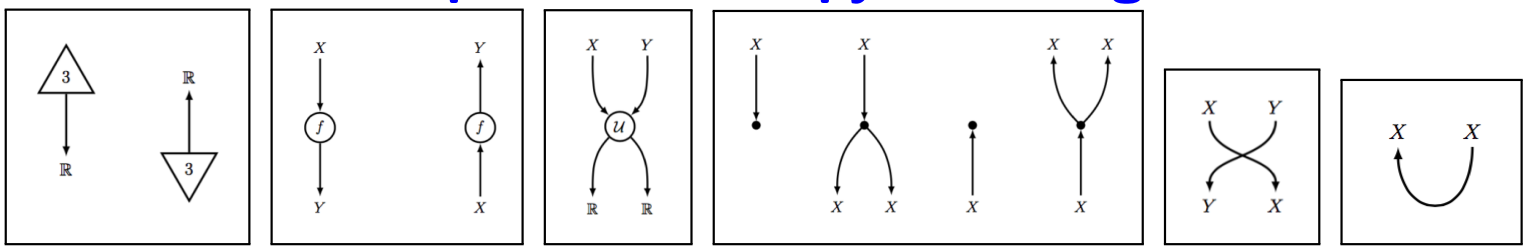
Visual Programming

- **String Diagrams**
  Visualise operations of monoidal categories
- **Simple Strings**
  Lego block = pregame
  Constants, computations, decisions, games, future
- **Complex Games**
  Sequential, parallel, boxing (substitution) combinators
- **Implementation**
  String diagrams have algebraic representation is Haskell

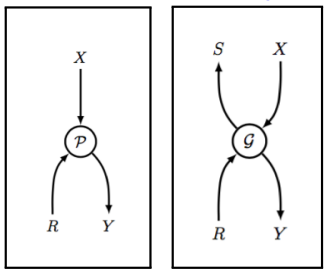
Language for Social Science

Basic Building Blocks

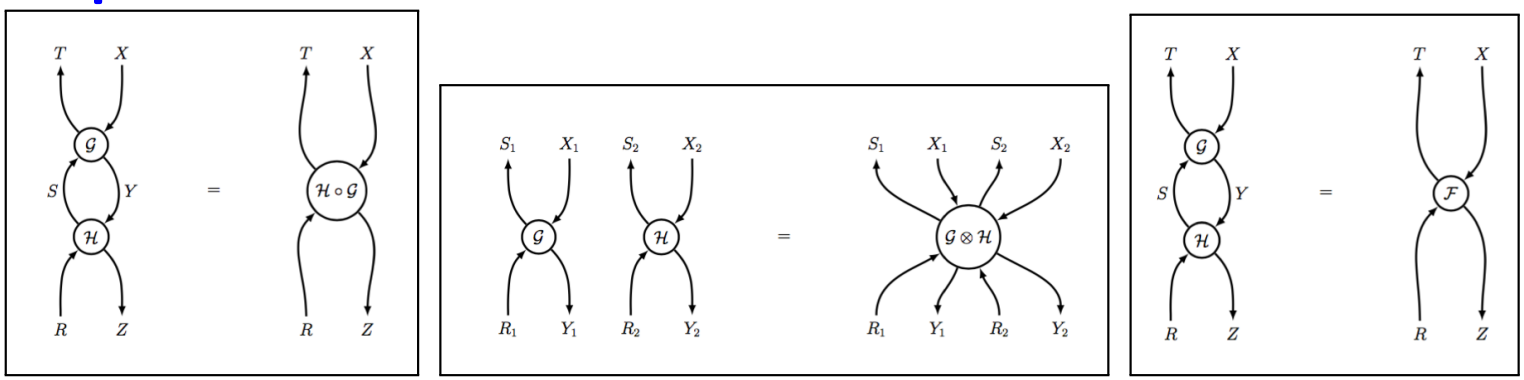
- **Constants, Computations, Copy, Braiding, Future**



- **Decisions, Games**

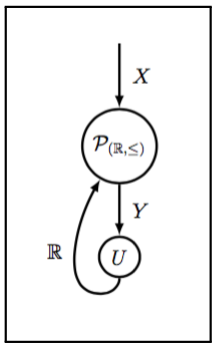


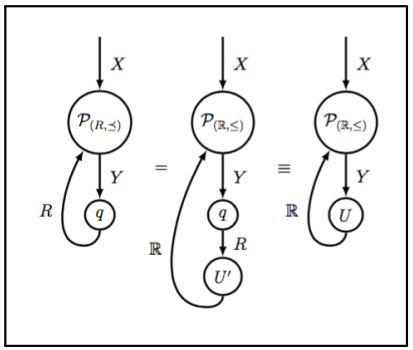
- **Sequential and Parallel Combinator, Substitution**

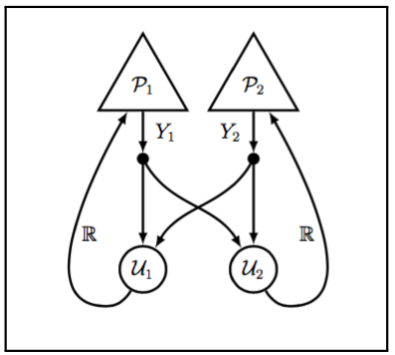


Games with Strings Attached

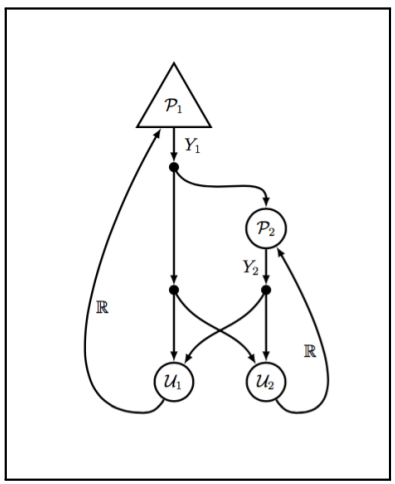
- **Decisions, Basic Game**

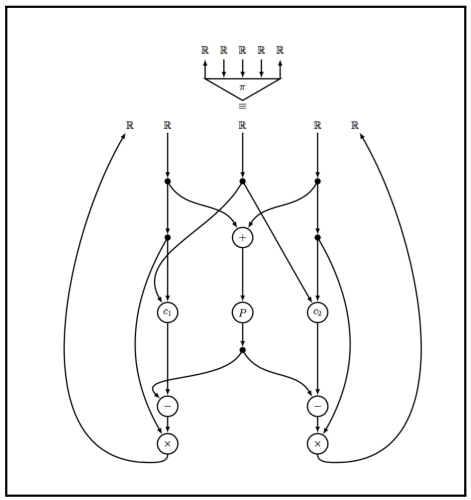


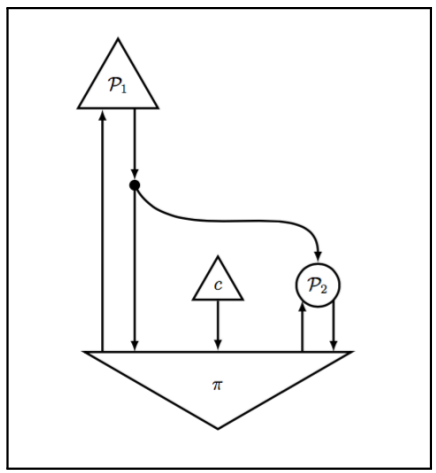




- **Complex Games**

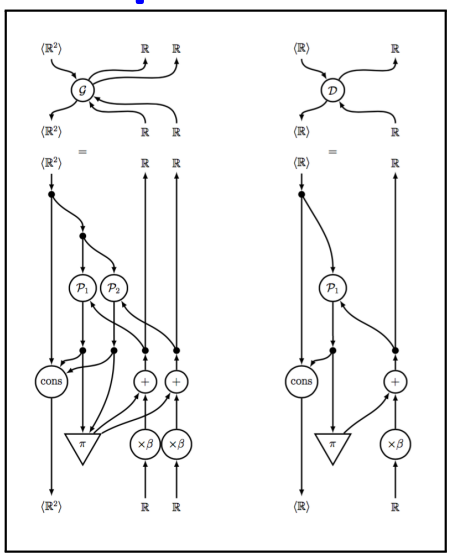


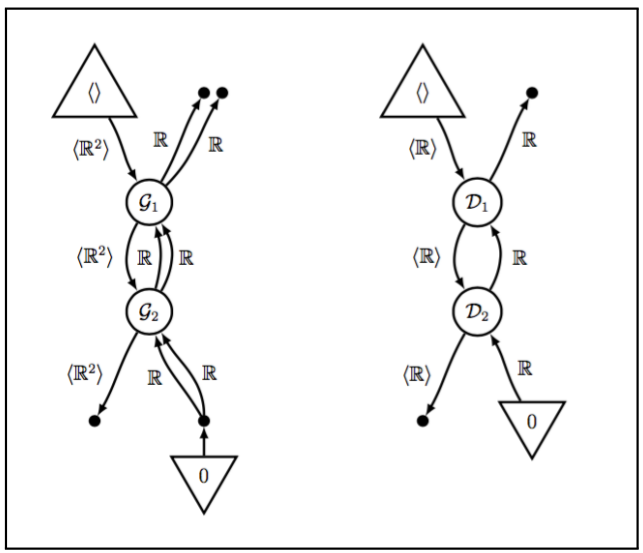


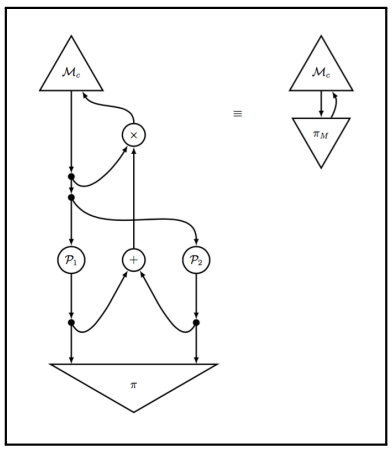


Games with Strings Attached

- **Complex Games, ... continued**



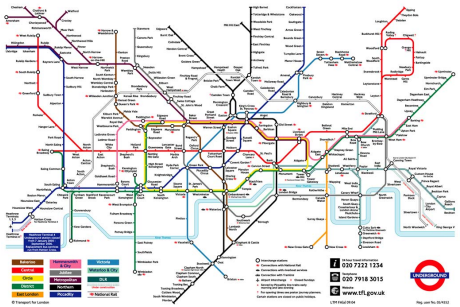




Complex Networks

Any problem can be formulated as a game

- **Social and Technical Networks**



- **Money and Energy Networks**

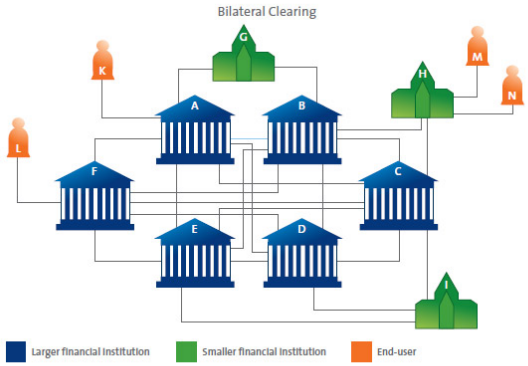
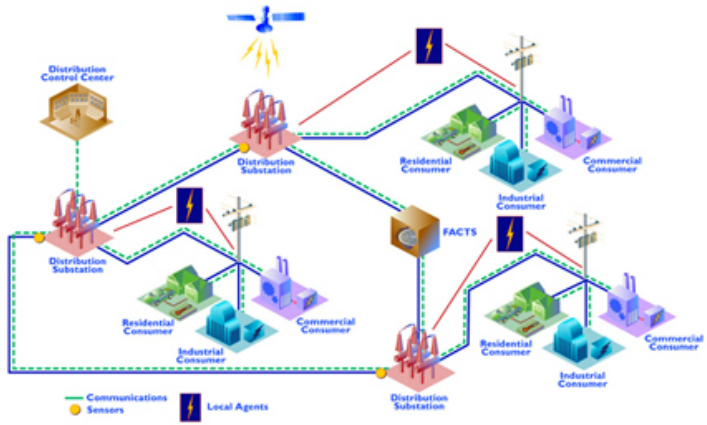
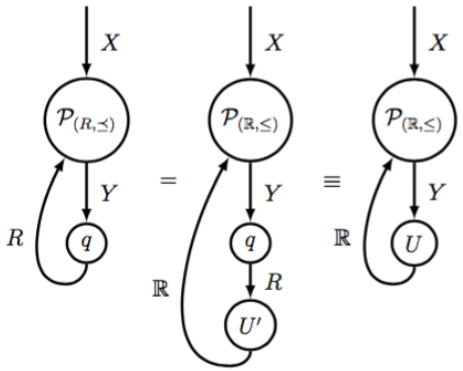


Figure 1: Bilateral Clearing involves numerous one-to-one transactions that appear to be independent but are really interconnected.



Compiler

- **String to ~ArrowNotation to Engine.hs**

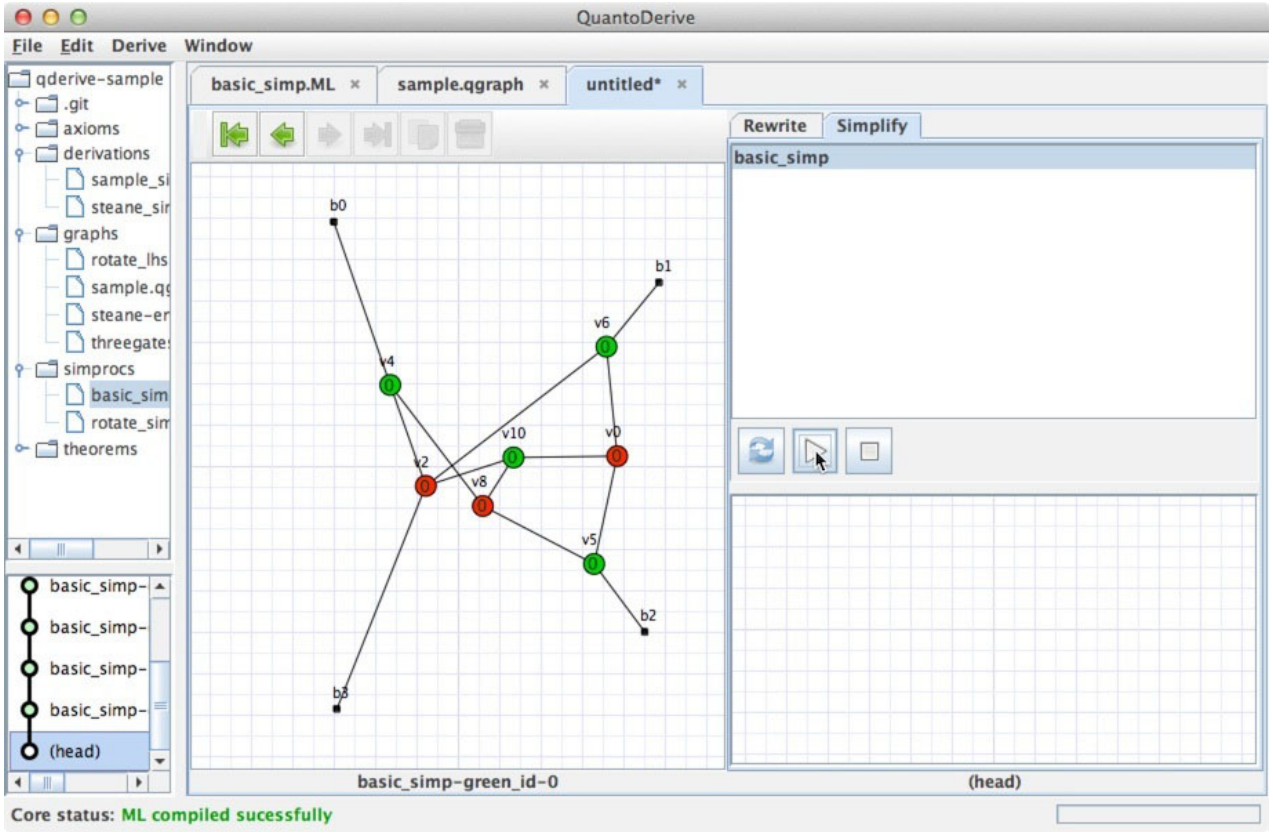


$$\begin{aligned} & \tau_R \circ (R^* \otimes q) \circ \mathcal{P}_R \\ = \; & \tau_{\mathbb{R}} \circ (\mathbb{R}^* \otimes (U' \circ q)) \circ \mathcal{P}_{\mathbb{R}} \\ \equiv \; & \tau_{\mathbb{R}} \circ (\mathbb{R} \otimes U) \circ \mathcal{P}_{\mathbb{R}} \end{aligned}$$

- **Type System**
  Unify, e.g. $(1 \otimes X) \otimes 1) \circ Y = X \circ Y$
  Dependent types, parametricity
  Needed at runtime (for string glueing at GUI)

String GUI

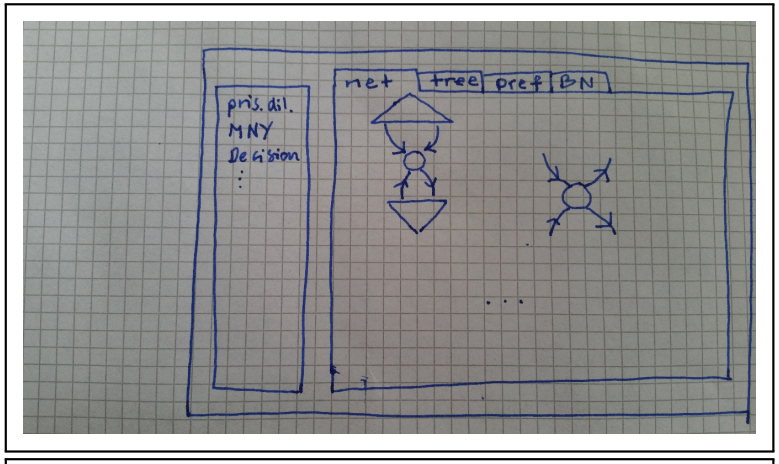
~Quantomatic

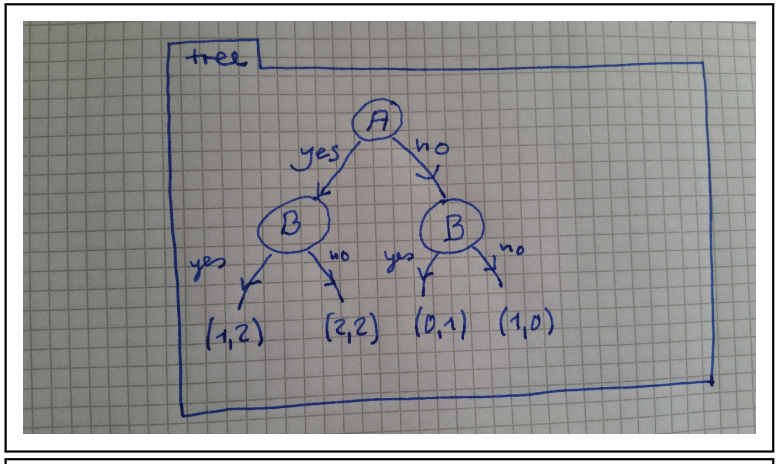


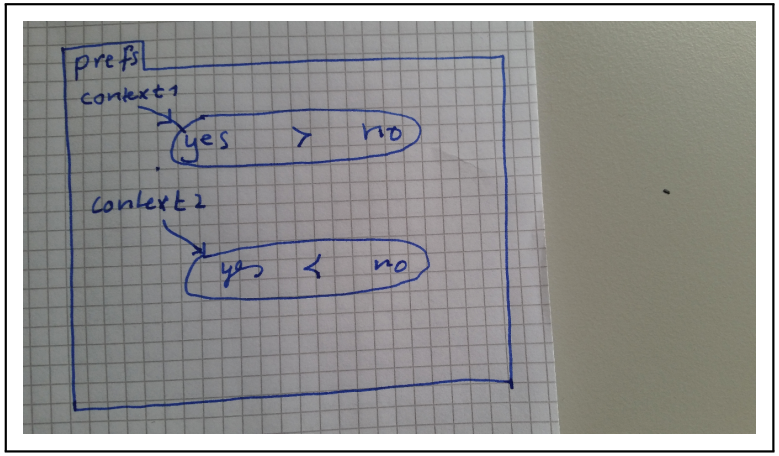
for quantum computing and linguistics

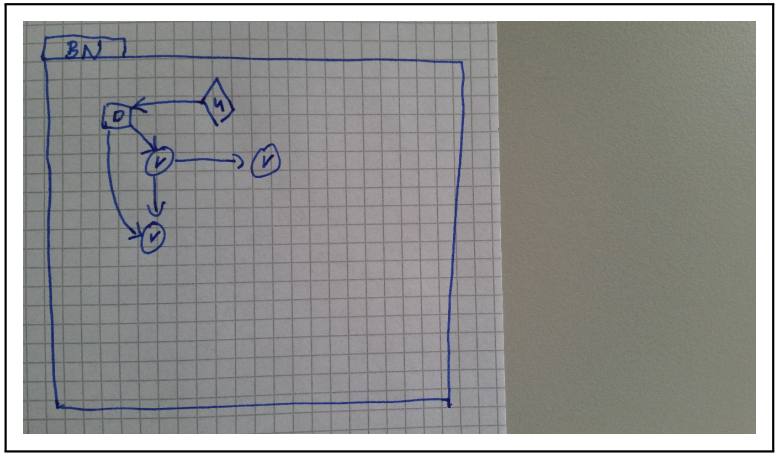
GUI for String Games

Strings net, extended tree, preferences, Bayes network









StartUps

- **Education**
  Let students, manager, economists, politicians play games
  Massive experimental decision data in real time
  Change games fast, get used to string system
- **Smart Energy Grid**
  Decentralised energy demand and!!! supply
  Prosumer (smart), fix point (higher-order) goal
- **Consulting**
  Large models, fast modelling, big data
  Merge and aquisition, auctions, ...
- **Finance, Insurance, Banking, Money**
  Debugging, sensitivity analysis, verification (model generating)
  Invariances in economics, banking and money theory
  Generalized algebraic double accounting (parallel semantics)
- **Gaming**
  Physics engine, AI engine, next: social engine

Work Ahead

- Content: mathematics, computer science, economic theory
  - In/finitely repeated games as coalgebras
  - String diagram compiler, type system, engine
  - Monads: probabilities, IO (human players)
  - Logics, equilibrium checker, debugger
  - Bicategory to quantify over games
  - Social choice, reference points, altruism
  - Self-control, preferences over preferences, ...
  - ... beliefs of belief, coalgebras
  - Incomplete and imperfect information, learning agents
  - Econometrics, Gaussian processes, algorithmic probabilistics
  - Fixed point goal over context, endogenous mechanism design
  - Network topology as strategy in game in network
- Context
  - Prototype, business plan, investors
  - Research proposals

What Do We Gain?

- Modelling in a high level language
- Modelling in the large by compositionality
- Observer isomorph to observed by reflexivity
- Decentralisation of control by fixed point goals
- Not only modelling but real world implementation
- Implementability, economics of economics
- Economic layer on top of engineered systems
- Overall goals: improved group decision making