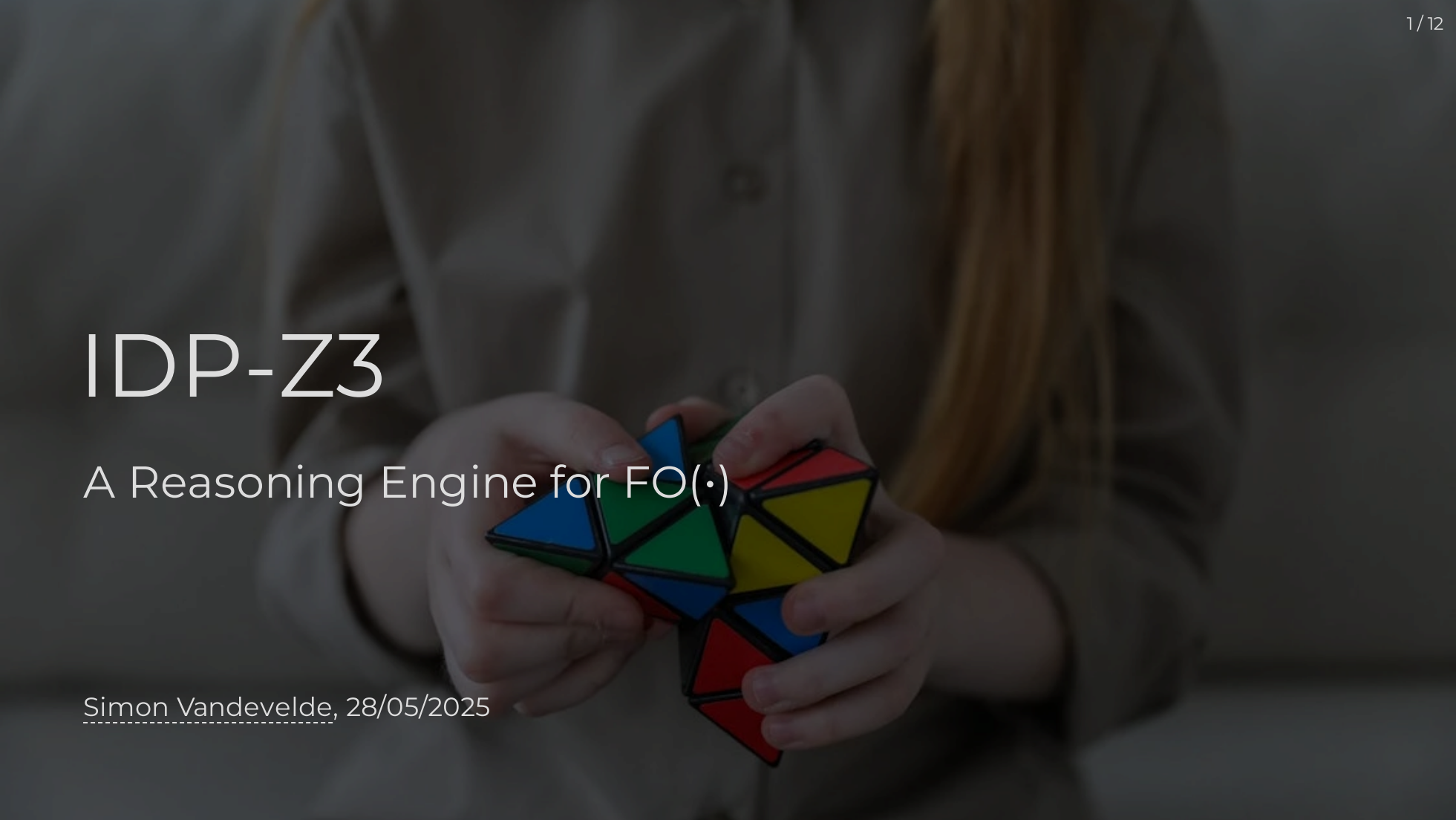


IDP-Z3

A Reasoning Engine for FO(·)

Simon Vandevelde, 28/05/2025

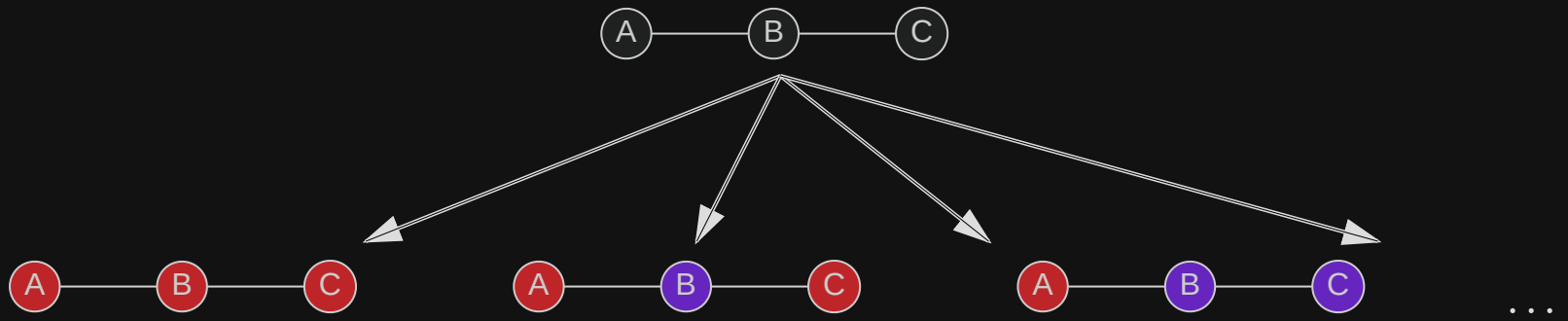
IDP-Z3

- Reasoning engine for FO(·)
- Written in Python
- Open source under LGPLv3
- Uses Z3 SMT solver internally
- Has a rich ecosystem

> 💭 **Main idea**
> "Radically declarative": one knowledge base, multiple uses.

Model-theoretic

Much of what IDP-Z3 does is related to distinguishing **good** from **bad** situations



→ Knowledge is stored in knowledge base

→ Written in FO(·)

Inferences

Given a formal description of what a correct situation should look like, we can:

- **Model-check**: is my situation correct?
- **Model generate**: show me a correct situation
- **Model expand**: given a partial solution, extend it to a full one
- **Propagation**: what do all feasible situations have in common?
- **Explanations**: why is a situation not feasible?
- **Optimization**: find the situation with the most X
- . . .

$\rightarrow$ all with the same knowledge base!

Small demo

$\rightarrow$ allows us to solve complex problems

Example use cases

Component designer:

- International company designs specific component
- Designs are complex
- Lots of engineers, each with own rules-of-thumb
- Lots of costly trial and error

Goal: build a tool to support the engineers in getting more "first-time right" designs

We built a KB capturing the experts' knowledge, which together with IC:

- Succeeds in achieving more first-time right
- Serves as learning tool for newer engineers
- Can challenge assumptions of senior engineers
- Keeps knowledge within the company

Some prior use cases

Intelli-Select:

- Collateral management for loans between large institutions
- Negotiating involves much manual work, takes months
- Using user-friendly interface, companies could enter their own knowledge to KB
- Negotiations in seconds

Notary:

- Registration duties on property purchases in Belgium
- 11 articles of law, quite complex
- Built tool to support notaries in "asking the right questions"
- Relevance inference is crucial!

$\rightarrow$ IDP-Z3 has many application domains!

Using IDP-Z3

Most of our use cases:

- Company has knowledge on a domain (design, financial, legal, ...)
- Wishes to support staff

$\rightarrow$ Using IDP-Z3 through CLI is (obviously) not feasible!

Two routes:

1. use the Interactive Consultant
2. embed IDP-Z3 through its API

Interactive Consultant

Interface for IDP-Z3

- Generic, flexible, interactive
- Main usage loop:
  - Users sets value of their choice
  - IC automatically shows derived knowledge
  - If user is unsure of derivation, they can ask *why* !
- More is possible:
  - Showing a solution
  - Showing an optimal solution w.r.t a term
  - Showing a high-level overview of all remaining solutions

Short demo

API

An ideal IDP-Z3 API should:

- abstract away underlying FO(·) concepts
- make sense from Python's perspective (be Pythonic)
- trivially allow you to re-use results of previous inference in next one
- be well-documented

 After five years of development, how many do you think we meet?

→ none

 Robin's Knowledge Base API (KeBAP) wrapper

Note: the current API definitely works, but it requires significant effort to learn

Development best practices

Competing interests when building a toolbox:

1. Work towards publications
2. Work towards industry adoption

→ Going fast on one typically means going slow on the other

Good common denominator:

- Git
- Issue tracker
- Feature branches (or similar flow)
- Merge reviews
- ...

Adoption by industry

Outreach is important!

- Website (https://IDP-Z3.be)
- Demos
- Online tutorial
- Matrix chat
- Blog posts

→ these have a bigger influence than you'd expect!

Thank you

More info:

- Project: https://idp-z3.be
- Online IDE: https://interactive-consultant.idp-z3.be
- Online tutorial: https://interactive-idp.gitlab.io/
- Slides: https://slides.simonvandevelde.be/DTAI25/slides.pdf

 https://simonvandevel.de

 s.vandevelde@kuleuven.be

 @saltfactory@mastodon.social

 Simon Vandevelde