Modelling cyclic structures in Agda

Evaluating Agda's coinduction through modelling graphs

Author: Faizel Mangroe (5566541)
F.Mangroe@student.tudelft.nl
Professor: Jesper Cockx
Supervisor: Bohdan Liesnikov

1. Motivation

- Graph theory plays an important role in a multitude of different fields such as, but not limited to the field of **chemistry**, **sociology**, **biology**, **operations research** and even **war science** and is widely used in computer science as well.
- Various graph representations exist: a **popular** approach is through an **inductive** way.
- **Cycles** occur in graphs → Inductive approach becomes less intuitive.
- **Solution**: the dual of induction, coinduction, which is not as well researched.

2. Background

Properties of graphs:

- **isPresent**: when provided a node and a graph, it checks if the node is present in the graph.
- **hasPath**: when provided a start node, an end node and a graph, it checks if the end node can be reached from the start node in the graph.
- **hasCycle**: when provided a graph, it checks if the graph contains a cycle.

Agda:

- Functional language
- **Total language**: all functions must terminate.
- **Dependently typed**: types can be indexed by objects of other types.

→ Agda can be used as a **proof assistant**.

- **Coinduction**: Agda has three flavours of coinduction, guarded coinduction, musical coinduction and sized types.

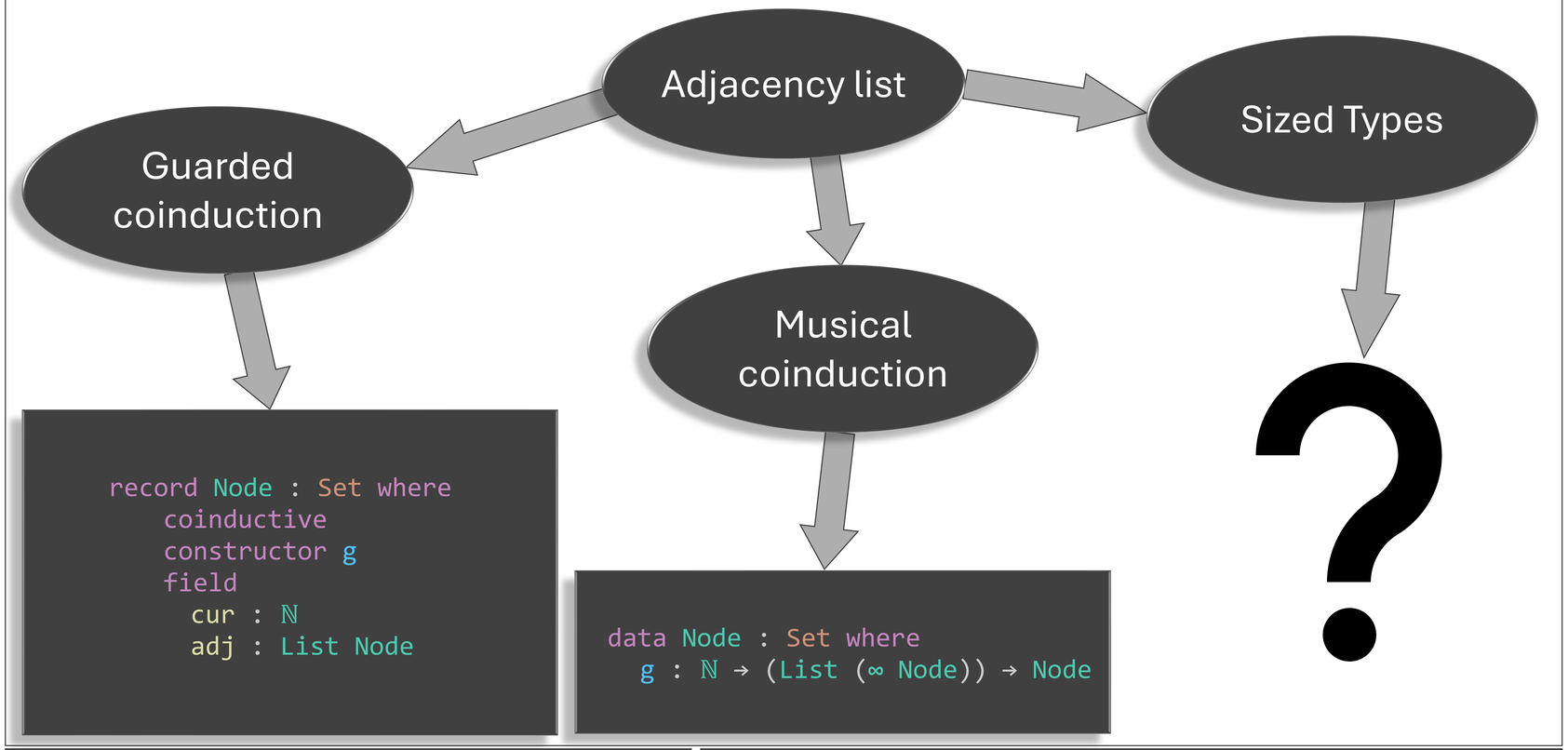
3. Research question

How can graphs be modelled coinductively in Agda such that they are suitable for the proof assistant?

Sub-questions

1. What are different encodings of graphs in Agda and which are suitable for the proof assistant?
2. What properties of graphs can be proven using the various encodings (e.g. has node)?
3. What improvements should be made to Agda in order for it to handle modelling with coinduction more easily?

4. Embeddings




```
record Node : Set where
    coinductive
    constructor g
    field
      cur : ℕ
      adj : List Node
```

```
data Node : Set where
  g : ℕ → (List (∞ Node)) → Node
```

5. Experiments

	Guarded	Musical	Sized Types
isPresent	✓	✓	✘
hasCycle	✓	✓	✘
hasPath	✓	✓	✘

6. Improvements to Agda

- The documentation of Agda does not provide enough information.
  → Difficult to get started with new concepts.
- The error messages in Agda are oftentimes too vague
  → Cumbersome debugging experience.

7. Conclusion and future work

- The encodings using guarded coinduction and musical coinduction were successful.
- The sized type encodings were not successful, due to a lack of understanding and time.
- Even though musical coinduction is deprecated, it seems to give the most power.

Future work:

- The use of sized types for modelling coinductive graphs can be further explored.
- Assessing the capabilities of the musical variant compared to the guarded version.

TU Delft