12th Transformation Tool Contest
TTC 2019, Eindhoven, July 19th

Intro to TT2BDD

Antonio Garcia-Dominguez, Aston University, UK

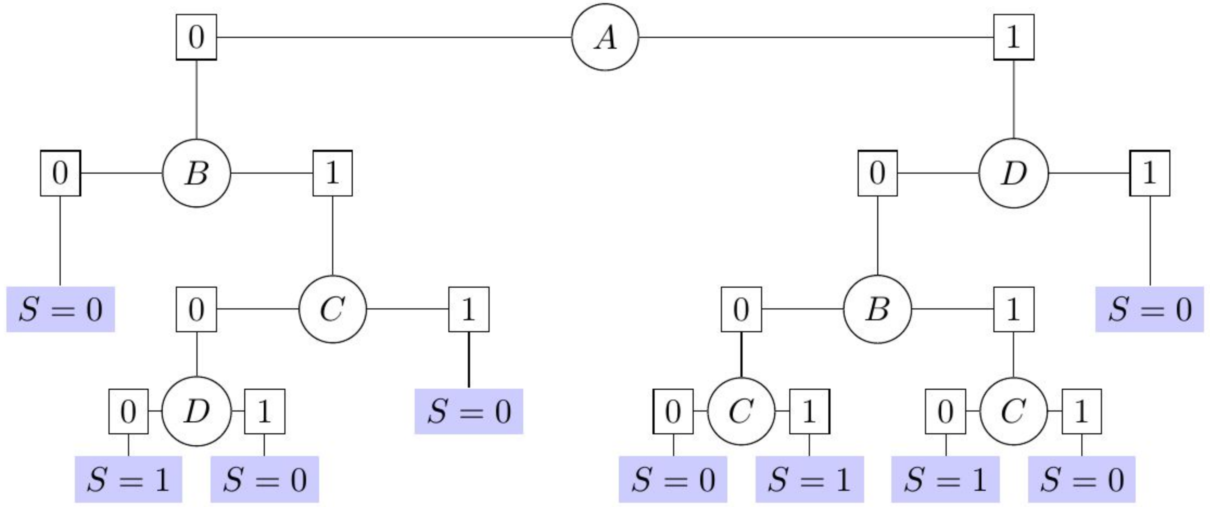
Why this transformation?

- Audience feedback from 2018:
  - Revisit ATL Zoo transformations (need updating!)
  - Research is going beyond "just performance"
  - (But that's still important!)
- We considered two transformations:
  - RSS to ATOM (reasonably complex metamodels)
  - TT to BDD (executable notations)
- TT to BDD seemed more interesting:
  - Both source and input metamodels are executable
  - Amenable for optimisation and verification problems
  - Very graph-oriented

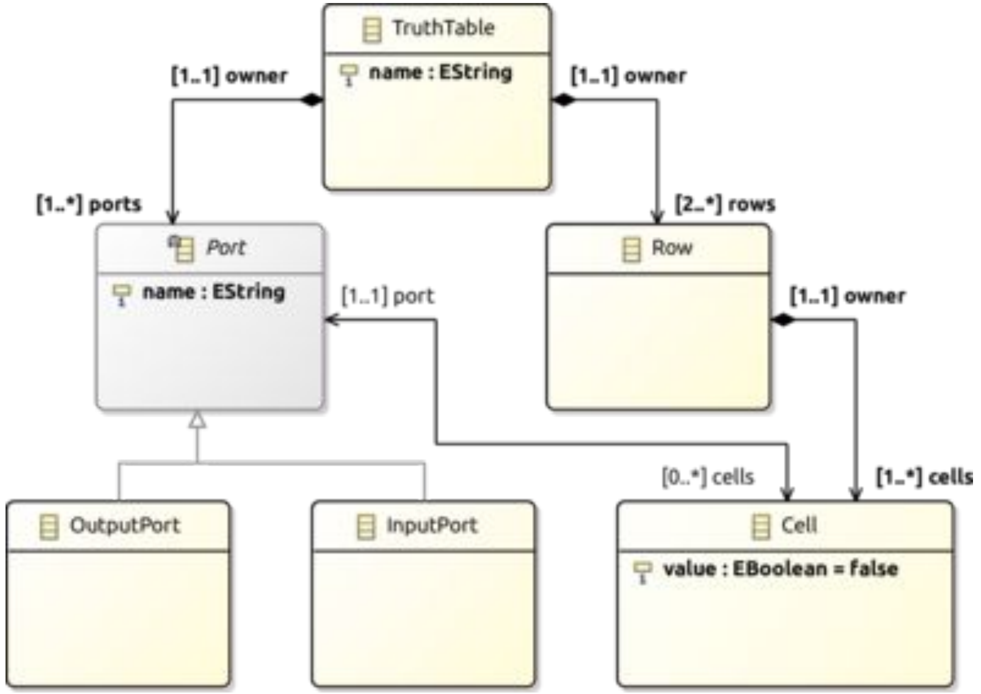
Transformation summary

- We start with a lookup table (with optional inputs)
- We want to produce a decision diagram
- Existing ATL Zoo tx was updated and repackaged

A	B	C	D	S
0	0	-	-	0
0	1	0	0	1
0	1	0	0	1
0	1	0	1	0
0	1	1	-	0
1	0	0	0	0
1	0	1	0	1
1	-	-	1	0
1	1	0	0	1
1	1	1	0	0



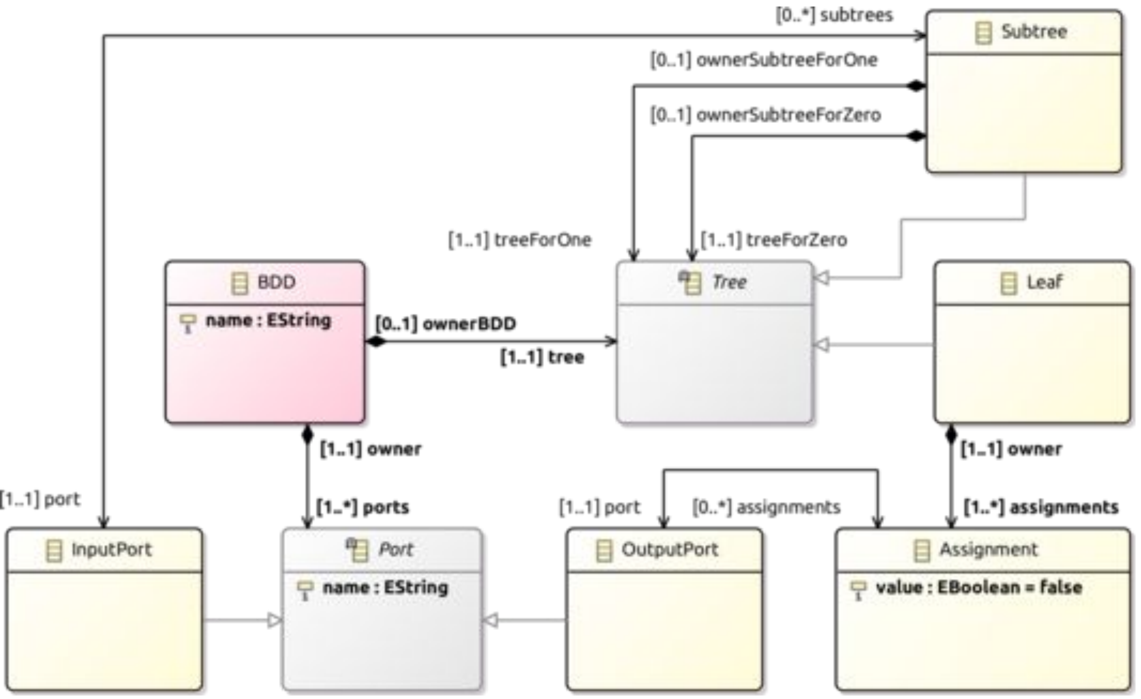
Input metamodel



- TruthTable (root type):
  - Has Rows
  - Has InputPorts
  - Has OutputPorts
- Rows have Cells
- Cell specifies a Boolean value for a certain Row

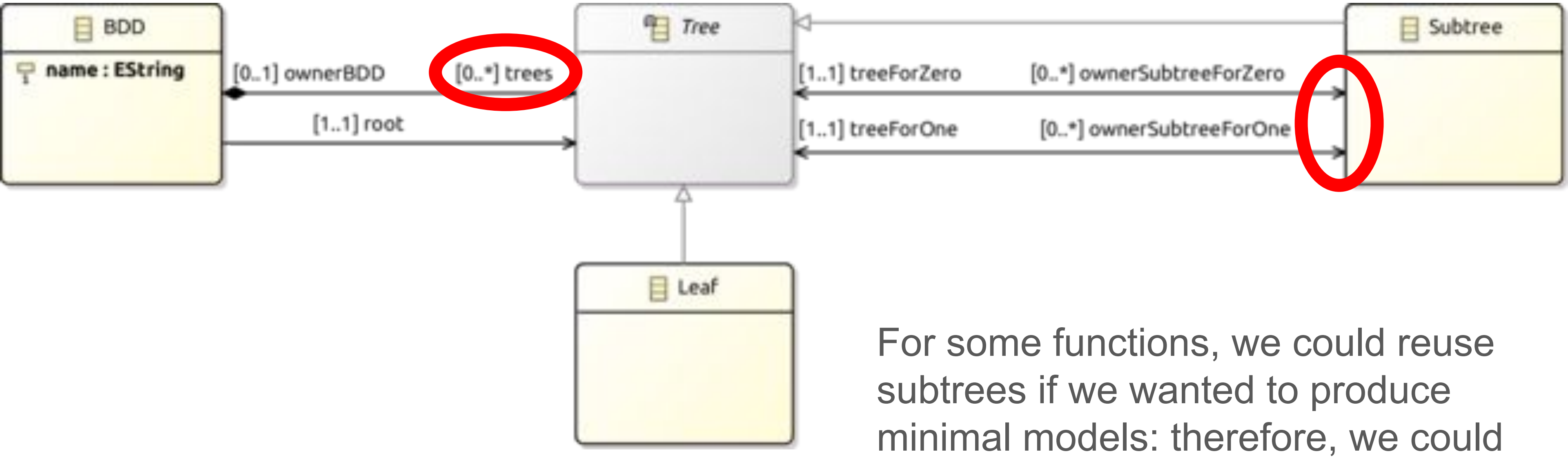
Very simple, executable, but slow to run.

Output metamodel A: Binary Decision Tree



- BDD (root type):
  - Has a Tree
  - Has I/O ports
- Tree structure:
  - Subtree: branch on InputPort value
  - Leaf case: have an Assignment of values for the OutputPort instances

Output metamodel B: Binary Decision Graph



For some functions, we could reuse subtrees if we wanted to produce minimal models: therefore, we could have a binary decision graph!

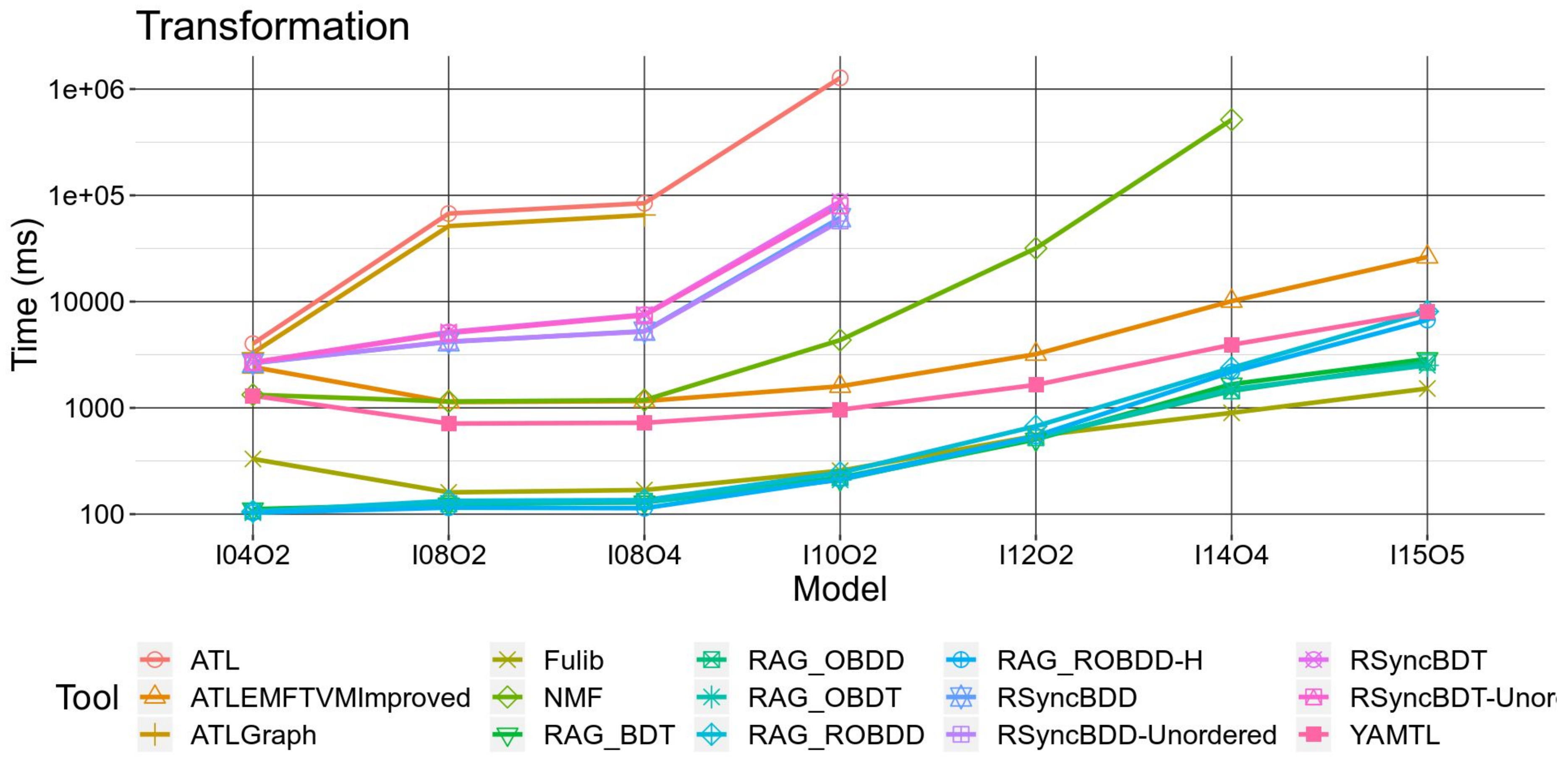
Concerns for this transformation

- Performance
  - A bad implementation could quickly increase in cost as we get larger and larger truth tables
- Optimality
  - We'd like smaller BDDs if possible
  - This is possible by reusing subtrees (graphs)
- Correctness
  - Going beyond manual testing (formal methods)
- Conciseness, usability, understandability...

Tooling for the case

- Generator
  - Can produce random tables of arbitrary size
  - Limitation: no optional inputs
- Validator
  - Runs the truth table and the BDD side-by-side, checking the results
- Dockerfile (new this year!)
  - https://hub.docker.com/r/bluezio/ttc2019-tt2bdd-git
  - Executed on Google Cloud Compute
  - c2-standard-4 (4 cores, 16GiB RAM) with 50GB SSD
  - US Central (Iowa)
  - Excludes MEEDUSE (does not use benchmark framework)

Performance figures



Time for the presentations!

http://bit.ly/ttc19-tt2bdd