Complexity of Path Semantics

by Sven Nilsen, 2022

In this paper I show that Path Semantics is extremely more complex than classical logic, by calculating the complexity of binary functions in Path Semantical Quantum Propositional Logic.

In normal Boolean Algebra[1], there are 16 binary functions[2]:

$$|\text{bool} \times \text{bool} \to \text{bool}| = |\text{bool}|^{|\text{bool} \times \text{bool}|} = 2^4 = 16$$

Each of the 16 binary functions have a name:

0000 false_2
0001 and
0010 nimply/exc
0011 fstb
0100 nrimply/rexc
0101 sndb
0110 neqb/xor
0111 or
1000 nor
1001 eqb/nxor
1010 nsndb
1011 rimply/nrexc
1100 nfstb
1101 imply/nexc
1110 nand
1111 true_2

For each symmetric normal path[3] by `not`, one gets a pair of binary functions, for example:

and[not] <=> or   or[not] <=> and

These two normal paths are known as "De Morgan's laws"[4].
This means that there are 8 functions pairs that are central to how we think about Boolean algebra.

However, normal Boolean algebra can be extended in different ways. For example, one way is Answered Modal Logic[5] or Uberwrong Logic[6], which are equivalent. Another way is Homotopy Level Two Computing[7]. There exists other four-value logics as well[8].

Since four-value logic extends normal Boolean algebra by replacing a single bit with two bits, it follows that all extensions to four-value logic are in some sense isomorphic. Yet, the number of binary functions in four-value logic is so vast, that treating these four-value logics as the same language is impractical:

$$|\text{bool}^2 \times \text{bool}^2 \to \text{bool}^2| = |\text{bool}^2|^{|\text{bool}\hat{}2 \times \text{bool}\hat{}2|} = 4^{16} = 4294967296$$

It means, most of these functions are never given any name in practice. This is why for example Uberwrong Logic can have 16 "authentic" functions and 16 "inauthentic" functions, although any of these functions are just one among 4294967296 others. The bias of language is a perspective.

The number of binary functions in an extended logic of N-bit values is given by the formula:

(2^n)^((2^n)^2)

Here is a table of this sequence up to 5 bits:

N	Number of binary functions
0	1
1	16
2	4294967296
3	6277101735386680763835789423207666416102355444464034512896
4	179769313486231590772930519078902473361797697894230657273430081157732675805500963132708477322407536021120113879871393357658789768814416622492847430639474124377767893424865485276302219601246094119453082952085005768838150682342462881473913110540827237163350510684586298239947245938479716304835356329624224137216
5	1877490722242957624872829180435341496724700098100283274460625329492637173681270245761408411049731203702734872618769510830040014131372041737509569386532117887241904300959844699137769324319635466404046613775211702424542813935648836980421603625974932396761795424304082300269676754082443695342254061823340538609531908514107639682502317669663681500314797335324943893622639668297747395498745762177028020499491750441442269164082711285254276222251984105530890643495787038835061974088337280329375413633916444796382640148613966582189470689858262573842718580303528077559712773603632935703500067952561169438356098133486564517039427396159107268796275165897559426150595849536951589067763490785316416993769747839819662724856547324922632131864922254772606755475239323370610204061202509641360345293472994640721638000761877425765953796863438657220422192125386641334314055984766186323786943900169865080654843883682635344894620210914425806918834492585431487638196081082780252276301518494881632302710172093333395720988740976057096835550749863080746440754655249087581510612392073586323748205223023085938674861596998002557757181131629264349612092483946559961088496134888998178721882995203630812827375954695021897215612858898975153639297277445444474175266343835870590702938059969357077134905684379819613000341267568632012618492570395808315383447143245938796881260027803044841450689970286565413242719284402997303612437382766580360521399647072371678262086743847196895014854614590190925113537445109771795598947173720612604679126916219977682688555907263946115046451445 76

An easier formula to use is one that tells the position in binary format, a `1` followed by `0`s:

n*((2^n)^2)

For example, at `N = 18`, the number that counts binary functions take up more than 1 TiB of memory. The complexity of this logic is incomprehensible to humans.

When considering that Homotopy Level Two Computing is just a simplification of Path Semantical Quantum Propositional Logic[9] where `~` can be applied at most once, it follows that Path Semantics[10] is extremely complex. Most of Path Semantics will forever be hidden and unnamed.

References:

[1] "Boolean algebra"
Wikipedia
https://en.wikipedia.org/wiki/Boolean_algebra

[2] "Binary function"
Wikipedia
https://en.wikipedia.org/wiki/Binary_function

[3] "Normal Paths"
Sven Nilsen, 2019
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/normal-paths.pdf

[4] "De Morgan's laws"
Wikipedia
https://en.wikipedia.org/wiki/De_Morgan's_laws

[5] "Answered Modal Logic"
Sven Nilsen, 2020
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/answered-modal-logic.pdf

[6] "Uberwrong Logic"
Sven Nilsen, 2022
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/uberwrong-logic.pdf

[7] "Homotopy Level Two Computing"
Sven Nilsen, 2022
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip2/homotopy-level-two-computing.pdf

[8] "Four-valued logic"
Wikipedia
https://en.wikipedia.org/wiki/Four-valued_logic

[9] "Path Semantical Quantum Propositional Logic"
AdvancedResarch – Summary page on Path Semantical Quality
https://advancedresearch.github.io/quality/summary.html#psq---path-semantical-quantum-propositional-logic

[10] "Path Semantics"
AdvancedResearch
https://github.com/advancedresearch/path_semantics