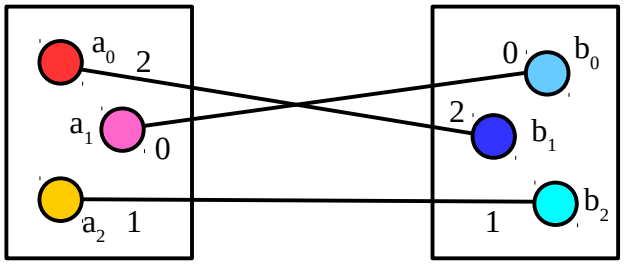
Proving Isomorphism Using Indexing

by Sven Nilsen, 2020

In this paper I present a technique in Path Semantical Logic to prove isomorphism.

An isomorphism[1] is the existence of a map between two collections of objects such that each object in the two collections has a partner object in the other collection. When using indexing to prove an isomorphism, each end point of a pair is assigned a unique number.



In normal propositional logic[2], proving an isomorphism is very difficult.
The reason is that uniqueness requires adding many axioms.

In Path Semantical Logic with level 1 and 0, there are two groups of propositions.
An equality in level 1 propagates into an equality in level 0 between uniquely associated propositions.
Although this is difficult to wrap your head around, one can think of level 0 as types and 1 as members:

Members (Level 1)	**Types (Level 0)**
a_0, a_1, a_2	A
b_0, b_1, b_2	B
i_0, i_1	

The two propositions `$i_0$` and `$i_1$` are used for indexing, encoding the index like a binary number:

$\neg i_0 \wedge \neg i_1$	00	0
$\neg i_0 \wedge i_1$	01	1
$i_0 \wedge \neg i_1$	10	2
$i_0 \wedge i_1$	11	3

Using the Index Theorem[3], one can at the same time set type membership and an index:

$(A \wedge 0)=a_0$	$(B \wedge 0)=b_0$
$(A \wedge 1)=a_1$	$(B \wedge 1)=b_1$
$(A \wedge 2)=a_2$	$(B \wedge 2)=b_2$

That is all! This is sufficient for isomorphism and does not include e.g. `$a_0 \neg= a_1$`.

Q.E.D.

References:

[1] “Isomorphism”
Wikipedia
https://en.wikipedia.org/wiki/Isomorphism

[2] “Propositional calculus”
Wikipedia
https://en.wikipedia.org/wiki/Propositional_calculus

[3] “Index Theorem”
Sven Nilsen, 2020
https://github.com/advancedresearch/path_semantics/blob/master/papers-wip/index-theorem.pdf