**Denial of the alleged Łukasiewicz nightmare in system Ł$_4$**



**Abstract:** The alleged Łukasiewicz nightmare of (◇p&◇q)→◇(p&q) is not tautologous in Prover9; however, the equation recast in one variable as (◇p&◇~p)→◇(p&~p) is tautologous. In Meth8/VŁ4, both propositions are tautologous. This speaks for Meth8/VŁ4, based on the corrected modern Square of Opposition as an exact bivalent system, as opposed to Prover9, based on the uncorrected modern Square of Opposition as an inexact probabilistic vector space.

We assume the method and apparatus of Meth8/VŁ4 with Tautology as the designated *proof* value, **F** as contradiction, N as truthity (non-contingency), and C as falsity (contingency). For results, the 16-valued truth table is row-major and horizontal, or repeating fragments of 128-tables, sometimes with table counts, for more variables. (See ersatz-systems.com.)

LET p, q: Schrödinger's cat is alive; Schrödinger's cat is dead;
~ Not, ¬ ; + Or, ∨, ∪; - Not Or; & And, ∧, ∩; \ Not And;
> Imply, greater than, →, ⊢; < Not Imply, less than, ∈;
= Equivalent, ≡, ⊨; @ Not Equivalent, ≠;
% possibility, for one or some, ∃, ◊, M; # necessity, for every or all, ∀, □, L;
(p=p) T as tautology; (p@p) **F** as contradiction;
(%p<#p) C as contingency, Δ; (%p>#p) N as non-contingency, ∇;
~( y < x) ( x ≤ y), ( x ⊆y).

From: Łukasiewicz, J. (1920). On three-valued logic in L. Borkowski (ed). 1970. 87-88.
Łukasiewicz, J. (1953). A system of modal logic. *Journal of Computing Systems*. 1:111-149.

**Remark:** The term "nightmare" was attributed to J-Y. Béziau in 2011 for the purpose of a four-valued schema of paraconsistent logic to evaluate systems based on numeric values such as -x, +x, -y, +y. The motivation was to discount the fact that the Ł$_4$ logic system was provably bi-valent (James, 2010, Estoril), and hence it was not mappable into a vector space, the continuing definition of paraconsistent logic.

This proposition is supposed to be egregious to logic system Ł$_4$: (◇p&◇q)→◇(p&q). (1.0)

*If possibly the cat is alive and possibly the cat is dead, then possibly both the cat is alive and the cat is dead.* (1.1)

(%p&%q)>%(p&q) ; TTTT TTTT TTTT TTTT (1.2)

Assumptions: ((exists(p) & exists(q))).
Goals (exists(p&q)). Exhausted. (1.3)

Prover9 invalidates Eq. 1.0 to show Ł$_4$ is untenable as an alethic logic.

If we preload p=~q as the antecedent to Eq. 1.0, then: (2.0)

*If possibly the cat is alive is equivalent to Not (the cat is dead), then*
*if possibly the cat is alive and possibly the cat is dead, then*
*possibly both the cat is alive and the cat is dead.* (2.1)

%(p=~q)>(%(p&q)>(%p&%q)) ; `TTTT TTTT TTTT TTTT` (2.2)

Assumptions: (exists(p<->-q)).
Goals: (exists(p)&exists(q))->(exists(p&q)).
Exhausted. (2.3)

Prover9 invalidates Eq. 2.0 to show Ł$_4$ is untenable as an alethic logic.

**Remark 2.3:** Eq. 2.3 shows Prover9 does not distribute the existential quantifier.

We rewrite Eq. 2.1 using one variable and its negation as respectively *alive* and *not alive*:

(◇p&◇~p)→◇(p&~p). (3.0)

*If possibly the cat is alive and possibly the cat is not alive, then*
*possibly both the cat is alive and the cat is not alive.* (3.1)

(%p&%~p)>%(p&~p) ; `TTTT TTTT TTTT TTTT` (3.2)

Assumptions: (exists(p)&-exists(p)).
Goals: (exists(p&-p). Theorem. (3.3)

Prover9 validates Eq. 3.0 to show Ł4 is tenable as an alethic logic.

We explain Eqs. 1.2, 2.2, and 3.2 as rendered as tautologous in Meth8, but 1.3 as exhausted in Prover9 in this way For more than one variable, the vector space for arity with Prover9 diverges from the bivalance inherent inVŁ$_4$, in which modal operators and quantifiers are d distributive. This speaks to Meth8/VŁ4, based on the *corrected* modern Square of Opposition as an exact bivalent system, opposed to Prover9, based on the uncorrected modern Square of Opposition as an inexact probabilistic vector space.

**Remark 3.2:** Meth8/VŁ4 distinguishes between Eqs. 2.0 and 3.0 by protasis and apodosis as:

%p&%q ; `CCCT CCCT CCCT CCCT` (1.2.1.2)
%(p&q)=(p=p) ; `CCCT CCCT CCCT CCCT` (1.2.2.2)

and

%p&%~p ; `CCCC CCCC CCCC CCCC` (3.2.1.2)
%(p&~p)=(p=p) ; `CCCC CCCC CCCC CCCC` (3.2.2.2)