Will this Function Terminate? Termination Analysis for Wikifunctions

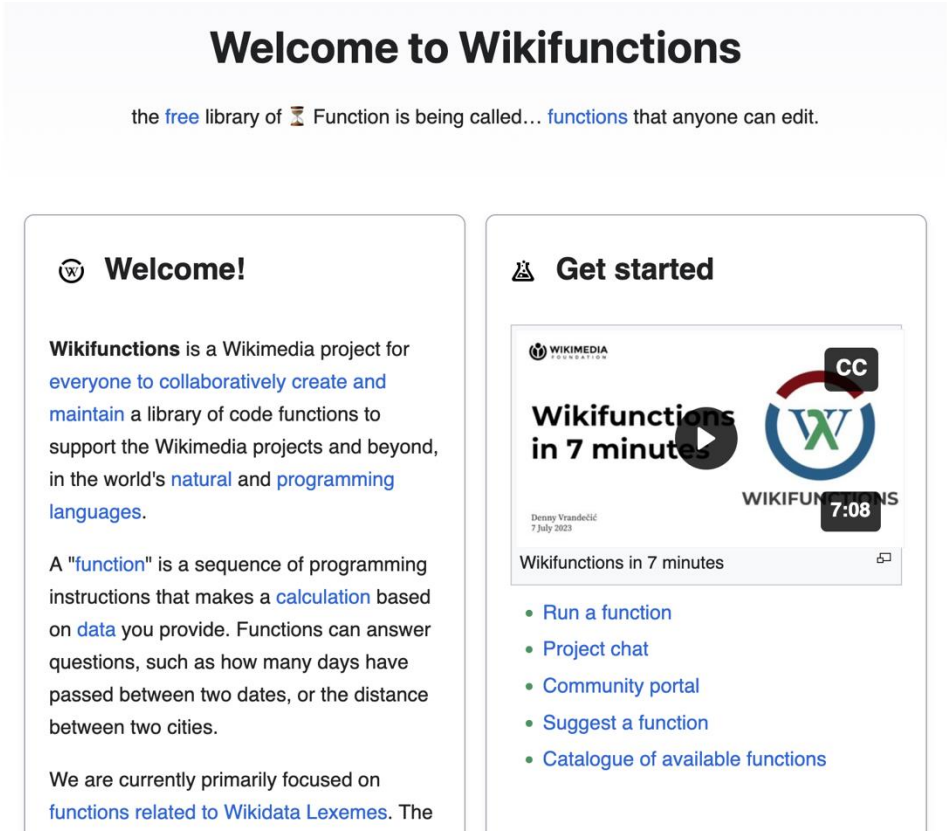
Eddie Begovic, Axel Polleres
WU Wien (Vienna University of Economics and Business)



2 Wikis, with different purposes…

https://www.wikifunctions.org/

https://termination-portal.org/





Wikifunctions…

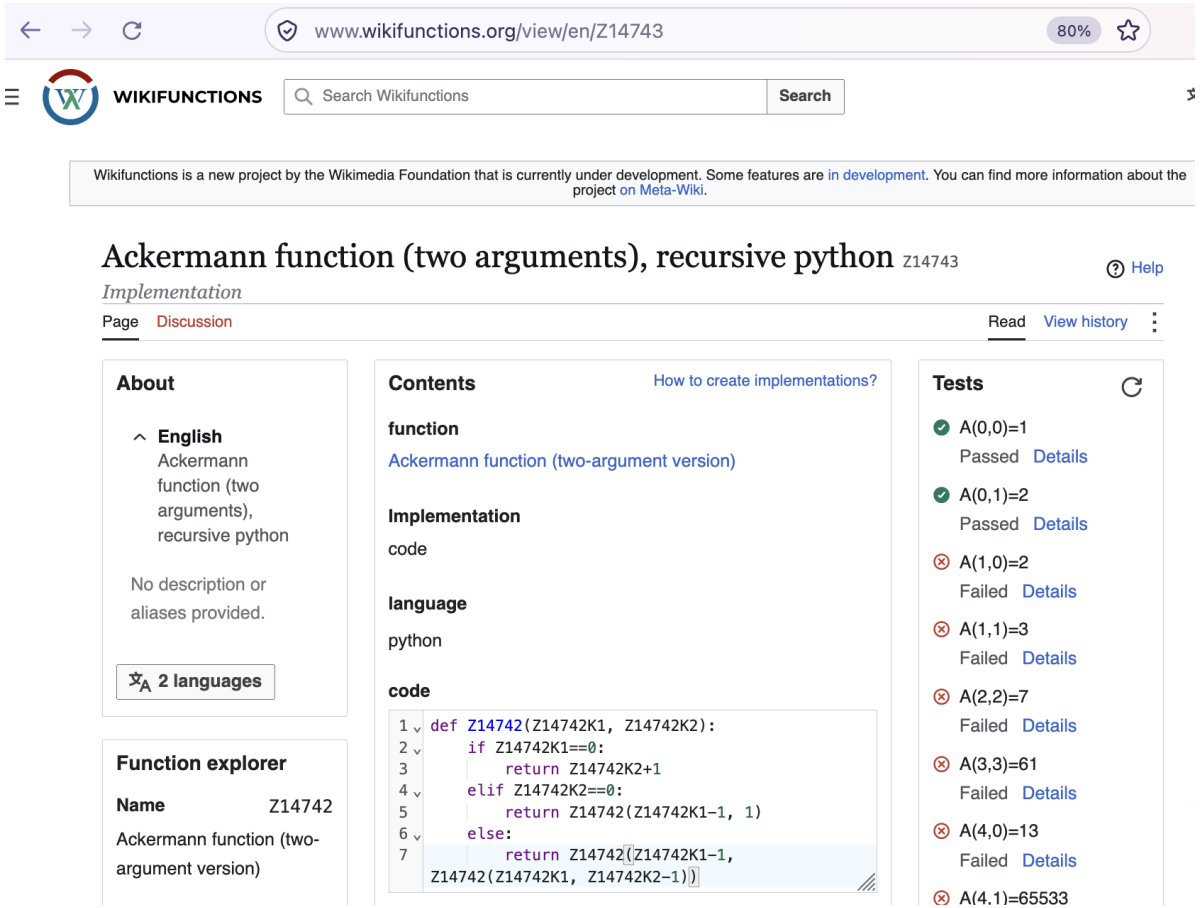
Functions to try out

reverse string (Z10012): Inverts the order of the characters in a String (see Z10548 for one with Unicode grapheme support)

Get day of week from date (Z13163): returns the day of week when given a date as year, month, and day of month

age (Z20756): returns the difference between two dates on the Gregorian calendar in years

how many days between two days in the year (Z20733): number of days (non-inclusive) between two days of the year



Anyone can add code, but is it guaranteed to terminate?

- Wikifunctions is Wikimedia's open library of functions and complements Wikidata
- Like items on Wikidata, every function and implementation has its own ID, here Z13667 and Z13668
- Anyone can add implementations in **Python** or **JavaScript**, or compose them from other functions (Python de facto standard)
- Automated agents calling functions could benefit from documented termination guarantees

```
def Z13667(Z13667K1):
    k=1
    for i in range(1,Z13667K1+1):
        k*=i
    return k
```

Implementation Z13668 of factorial (Z13667), from the dump of 1 May 2026

Termination… no chance in general..



But…

- Active community around automated analysis of function classes that terminate.
- E.g. using *ranking functions* and *loop invariants* on loops:

```
def Z13612(Z13612K1, Z13612K2):
    if (Z13612K1 == 0):
        return Z13612K2
    x = Z13612K1
    y = Z13612K2
    while(y):
        x, y = y, x % y
    return x
```

Loop guard
y != 0

$$V = \{x, y\}$$

$$\Theta : x > 0 \wedge y \geq 0$$

$$\tau : y \neq 0 \wedge x' = y \wedge y' = x \bmod y$$

y steadily *decreases* in a *strictly monotic* manner, with *lower bound 0* in each transition

Our question: How far do existing termination tools get on real Wikifunctions code?

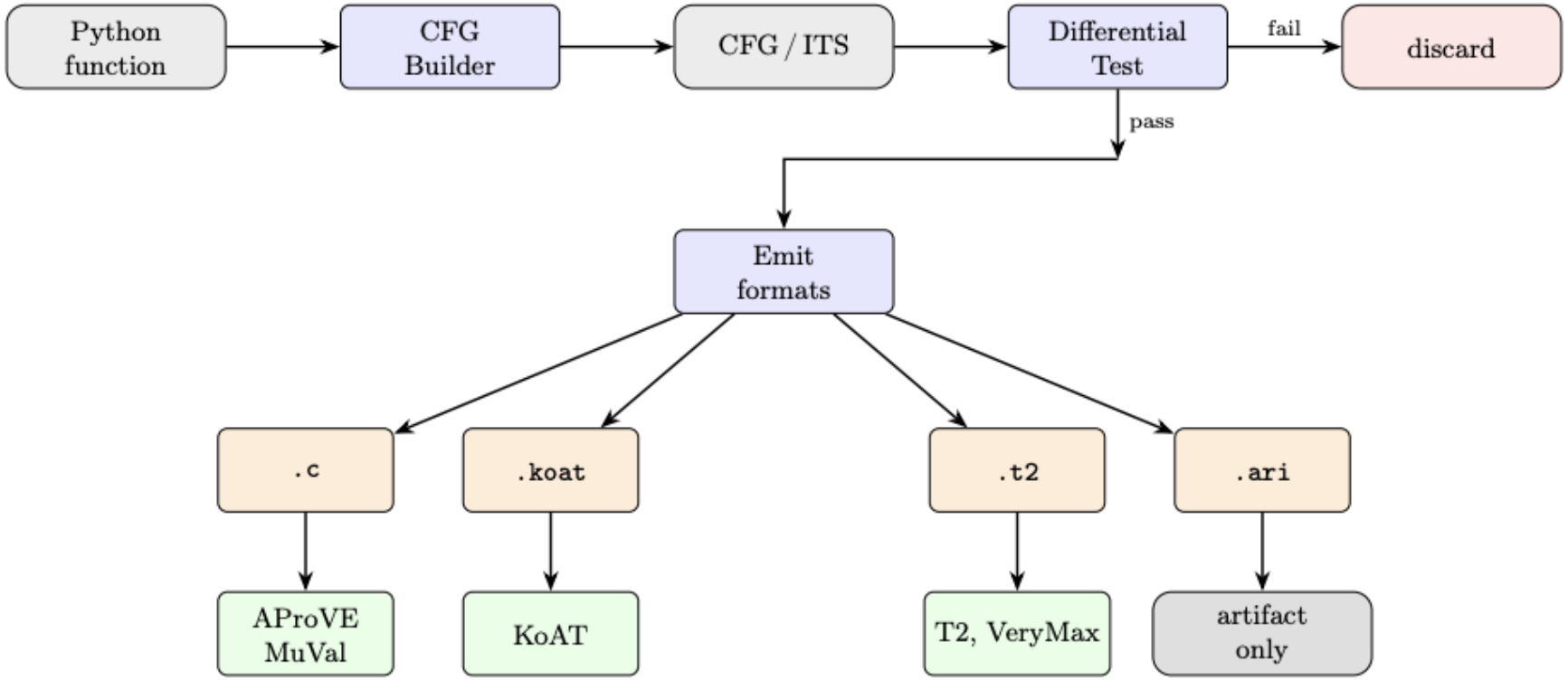
Tools typically don’t accept Python…

Tool	Input formats accepted	Used here
AProVE	Java, C, Haskell, Prolog, TRS [28]	C
MuVal	μCLP, C through a front end [49, 26]	C
T2	native T2 format, C via LLVM [19]	T2
VeryMax	T2, C and C++ via LLVM [16]	T2
KoAT	integer programs in the TermCOMP ITS format, C via llvm2kittel [48]	*.koat*

We restricted to

- integer loop functions, (no recursive functions yet)
- C tools need to be restricted to positive integer input functions only (as **Python and C disagree** on how the modulo operator (%)

Custom Translation pipeline (“Python to Integer transistion systems” and C…):

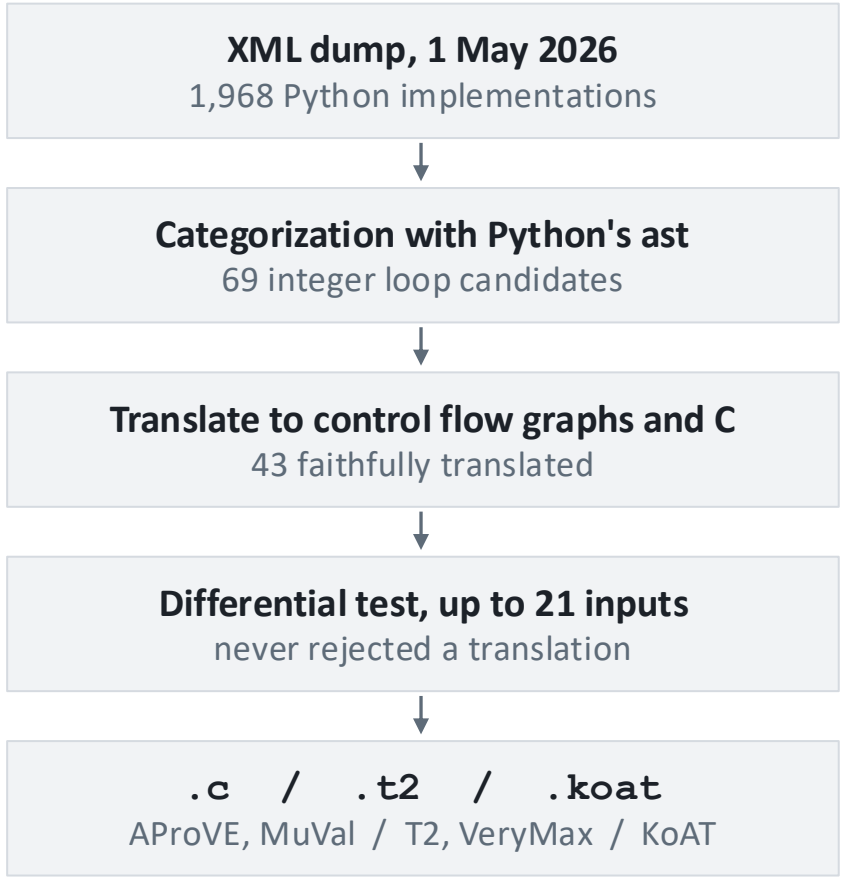


Admittedly coded with Claude’s help… verified with differential C <-> Python result tests.

1,435 terminate trivially, further 43 of 1,968 Python implementations can be analyzed with the tools:

Category	Count	%
No loop, no recursion (trivially terminating)	1,435	72.9
Loops over non-integer data (strings, lists, floats)	288	14.6
Recursion	156	7.9
Integer loops, faithfully translated (analysis set)	**43**	**2.2**
Integer loops not faithfully translatable	26	1.3
Parse errors (Python2 syntax)	20	1.0

Category distribution of the 1,968 Python implementations in the dump of 1 May 2026. Percentages are rounded.



All tools in their Termination Competition Docker images, 60 s per function

The tools decide 33 of 43 and never contradict each other

Tool	Input	Term.	Non-term.	Unknown	Timeout	Error
MuVal	`.c`	**29**	**1**	0	0	13
AProVE	`.c`	**15**	0	7	0	21
T2	`.t2`	**11**	0	30	0	2
VeryMax	`.t2`	**11**	0	2	1	29
KoAT	`.koat`	**6**	0	0	0	37
Combined		**32**	**1**	**10 undecided**		

Verdicts per tool over the 43 translated functions. The combined row counts each function once.

- **33 of 43 decided.** 32 terminating, 1 non-terminating
- **No contradictions.** Whenever two tools decided the same function, they agreed
- **20 of the 33** were decided by exactly one tool, so a portfolio beats any single tool
- **MuVal** is the strongest single tool with 30 conclusive verdicts

Every inconclusive result falls on a function with division or modulo

Functions	Count	Decided by at least one tool	Decided by all five tools
Without integer division or modulo	6	**6**	**6**
With integer division or modulo	37	27	0

Decided means a tool proved the function terminating or non-terminating. “At least one tool” counts the functions that got such a verdict from any of the five tools, “all five tools” the functions that got one from every tool.

What the tools report on the 37 functions with integer division or modulo

Tool	Outcome and message in the tool output
KoAT	Parser error, “Unexpected char: /” or “%” (35) and “Parser error … directly after =” (2)
VeryMax	“Semantic error: Denominator is not a constant integral expression.” (27), sort error (2), unknown (2), timeout (1)
T2	Parses the file, then “Termination/nontermination proof failed” (30), parse error (2)
AProVE	No output (20), “MAYBE” (7, four of them noting a division by zero), “Killed” (1)
MuVal	No verdict, the output stops after compiler warnings (13)

Some error messages quoted from the raw tool outputs

Most “interesting” finding:
MuVal finds a function that loops forever on input 0 !!!

```
def Z14961(Z14961K1):
    i = 1
    n = Z14961K1
    while True:
        if n % i == 0:
            n //= i
        else:
            break
        i += 1

    return n == 1
```

Implementation Z14962 of “is factorial” (Z14961)

Input 0 is valid, the argument is declared as a natural number

Trace for n = 0

`i = 1`	`0 % 1 = 0`	`n = 0`
`i = 2`	`0 % 2 = 0`	`n = 0`
`i = 3`	`0 % 3 = 0`	`n = 0`
…	`always 0`	**`never stops`**

Tool outcomes

Tool	Outcome on Z14962
MuVal	non-terminating
AProVE	no output
KoAT	parse error
T2	parse error
VeryMax	tool error

Checking 10 undecided manually… found another bug!

Implementation	Function	Manual result
Z13736	largest prime divisor	terminates
Z13748	n/(largest prime dividing n)	terminates
Z13768	number of unique prime divisors	terminates
Z13777	sum of unique prime divisors	terminates
Z14350	number of digits	terminates
Z14951	is k-almost prime	terminates
Z25892	is prime	terminates
Z14065	total stopping time (Collatz function)	**open problem**
Z14069	stopping time (Collatz function)	**open problem**
Z26991	N has number of factors of specified M	**does not terminate**

Function labels from Wikifunctions, the one for Z14350 shortened. Collatz termination for all inputs is a long-standing open problem (Tao 2022)

Sources: Tao 2022, Almost all orbits of the Collatz map attain almost bounded values, Forum of Mathematics Pi 10

```
def Z26988(K1, K2):
    i=0
    while (K1%K2==0):
        K1 //= K2
        i=i+1
    return i
```

Z26991, argument names shortened

- Terminates when K1 ≥ 1 and K2 ≥ 2
- Loops forever when K2 = 1, or K1 = 0 and K2 ≥ 1
- Division by zero when K2 = 0
- All of these are valid natural-number inputs

Limitations

- **Differential test** gives practical confidence of the adhoc translation tool, but it is **not a proof** of equivalence.
- **Different input domains for C and Python.** The C translations are restricted to non-negative integers, the ITS are not. All arguments are declared as natural numbers on Wikifunctions, and Python and C agree on division and modulo only for non-negative values, since **Python rounds down and C rounds towards zero**. The restriction is what keeps the translation faithful.
- **“Only” one non-termination found.** The tools proved non-termination once, for Z14962… Is this a limitation? …
- … yet, in an Open function repo like Wikidata, nothing speaks against adding non-terminating examples (future proof?)

Discussion

- **Where the tools apply, they are trustworthy.** No contradictions, best used as a portfolio.
- **Proposed: a dedicated metadata property** on Wikifunctions could record the such verdicts & integrate the analysis? (similar to constraint checks in Wikidata)
- **Editors** could see issues like Z14962 before **users** run into them, **agents** could retrieve machine-readable hints before a call
- Wikifunctions is a potentially interesting benchmark **for the termination community**. The Termination Portal is itself a MediaWiki wiki … could it be linked with Wikifunctions?

Implementation	Termination	Evidence
Z13668 factorial	**proven**	5 of 5 tools
Z14962 is factorial	**not on input 0**	MuVal
Z14065 Collatz	**undecided**	no tool verdict

Sketch of such a property

Sources: Termination Portal (termination-portal.org)

Future work

Code the pipeline does not cover yet

Direction	Size in the May 2026 dump	Possible route
Integer division & modulo	Cf. errors …	Naïve fallback loop implementation?
Recursion	156 Python implementations	Term rewriting systems or tail recursion rewritten into loops
Loops over strings, lists, floats	288 Python implementations	Ranking structural measures, such as the length of a list
JavaScript	374 functions with JavaScript but no Python implementation	Adapted translation pipeline, same analysis
Calls across functions	393 functions in a call cycle of compositions	Analysis beyond a single definition

Code, translations, tool outputs and results: github.com/16x29/Master-Thesis---Certainly-Terminating-Wikifunctions

Further directions

- **C only pipeline.** C is far more expressive than ITS and could cover more functions, but the cross-format comparison would be lost
- **Non-termination examples.** Documented non-terminating functions added to Wikifunctions on purpose would let future evaluations test non-termination detection
- **BTW:** Why is there no SPARQL endpoint for Wikifunctions?

Thank you!