Package ‘caverify’

November 2, 2026

Type Package

Title Fast Strength-t Coverage Verification for Covering Arrays

Version 0.3.0

Description Verifies that an array covers every t-way interaction, the certificate check for covering arrays, using compiled C code with optional 'OpenMP' threading. Supports uniform and mixed-level (per-column symbol counts) arrays. Missing values mark flexible (``don't care'') entries, which contribute nothing to coverage, so a verified array remains covering however they are filled. Each set of t-1 columns is checked against all later columns at once, one bit per column, which is many times faster than checking every set of t columns on its own on large arrays. Designed to be easy to embed in other packages: a single C file with a registered .Call entry point and one R wrapper.

License MIT + file LICENSE

URL https://github.com/chrispbsmolen/ca-tools

BugReports https://github.com/chrispbsmolen/ca-tools/issues

Encoding UTF-8

Language en-US

Imports parallel

NeedsCompilation yes

Author Christopher Smolen [aut, cre]

Maintainer Christopher Smolen <chrispbsmolen@gmail.com>

Repository CRAN

Date/Publication 2026-11-02 20:30:08 UTC

Contents

ca_verify 2

Index **4**

`ca_verify` *Verify strength-t coverage of a covering array*

Description

Checks that an $N \times k$ array covers every t-way interaction: for every choice of t columns and every combination of their symbols, at least one row exhibits that combination. This is the certificate check for a covering array, run in compiled C at speeds suitable for large arrays and strengths.

Both uniform arrays (every column has v symbols, v^t combinations per column set) and mixed-level arrays (column i has its own v_i symbols, $\prod v_i$ combinations per column set) are supported; a uniform array is the degenerate mixed-radix case in which all radices coincide.

Usage

```
ca_verify(x, t, v = NULL, threads = NULL, report = 10L)
```

Arguments

`x`	an integer matrix or data frame of integers; rows are runs, columns are factors. `NA` marks a flexible ("don't care") entry: a row contributes nothing to a projection in which it has an `NA`, so a verified array is covering no matter how its `NA` entries are later filled.
`t`	interaction strength to verify (positive integer).
`v`	number of symbols. One of: `NULL` (the default) to infer per-column symbol counts from each column's own maximum (`"auto"` is an explicit alias); a single integer `>= 2` declaring a uniform array; or an integer vector of length `ncol(x)` giving each column its own number of symbols. Versions 0.1.x inferred one uniform value from the global data range instead; changed on a recommendation by Ulrike Groemping, a default should not assume a uniform CA. Declare `v` explicitly when the check should also catch a column that fails to reach its intended number of symbols. Symbols may be coded `0..v-1` or `1..v` (per column for mixed levels); 1-based input is detected and shifted automatically.
`threads`	number of threads. `NULL` (the default) picks automatically: half the machine's logical cores, single-threaded for small jobs, capped during CRAN checks. Override with `options(caverify.threads = n)` or this argument. Long runs are interruptible without aborting R.
`report`	maximum number of missing-tuple examples to collect. The examples are the first column sets with a gap in lexicographic order, each with its smallest missing value combination, whatever the number of threads.

Details

Two exact methods are implemented, and both give identical results.

The `"prefix"` method (since 0.3.0) writes every set of t columns as its $t-1$ smallest columns P plus its largest column d. For a fixed P the rows are grouped by their values on P, and one pass

over each group, with one bit per column, decides the coverage of $P \cup \{d\}$ for every later column d at once, 64 columns per machine word. On large arrays this is many times faster than checking each column set on its own; on the strength-4 binary arrays used in testing it was about 40 to 50 times faster than version 0.2.0 at a few hundred columns, with the advantage growing with the number of columns.

The `"direct"` method (the method of versions 0.1.x and 0.2.x) checks each column set on its own. It is used when the prefix method would need more than `getOption("caverify.maxmem", 2^30)` bytes, or for more than 64 symbols in a column. `options(caverify.kernel = "direct")` or `"prefix"` forces a method (`"auto"`, the default, chooses); this is meant for testing and comparison.

Value

An object of class `ca_verify`: a list with `covered` (logical), `colsets`, `gaps`, `missing_tuples`, `examples` (matrix; first `t` entries per row are 1-based column indices, the rest the missing value combination in the input coding), and `t`, `v` (a single integer for uniform arrays, an integer vector of per-column symbol counts for mixed-level arrays), `N`, `k`, and `kernel`, the method used (`"prefix"` or `"direct"`, see Details).

Examples

```
ca <- rbind(c(0,0,0), c(0,1,1), c(1,0,1), c(1,1,0))
ca_verify(ca, t = 2)          # a covering array of strength 2
ca_verify(ca[-1, ], t = 2)    # removing a run breaks coverage

# a mixed-level array: one 3-level and two 2-level columns
mca <- as.matrix(expand.grid(0:2, 0:1, 0:1))
ca_verify(mca, t = 2, v = c(3, 2, 2))
ca_verify(mca, t = 2)   # the default infers each column's count
```

Index

ca_verify, 2

print.ca_verify *(*ca_verify*)*, 2