Introduction

Wheeling Systems Checker 5 is a Windows desktop program you can use if you want to check/analyze/improve your coverings.

Starting from version 4.0.0 you need an active Internet connection and a valid subscription to the Covering Repository website to use the program.

Please visit http://www.coveringrepository.com for more information.

System requirements

Operating system: **Windows 7 SP1, Windows 8, Windows 8.1, Windows 10, Windows 11**
Architecture: **64 bits**
RAM: **2GB minimum, 16GB+ recommended**

Important notice: to install the required prerequisites, unzip to a local folder and double-click the **Setup.exe** file.

Acknowledgements

Many thanks to my great friend Giove for his invaluable help.

Deep gratitude to Stefano de Martino for generously sharing his pioneering idea for systems verification.

Special thanks to Alessandro Jurcovich and Terenzio Lazzarotto for their continued support, suggestions and beta testing.

Many thanks to all the Wheels Forum friends and customers from all over the world for their invaluable support.

Contents

Launching the program

Once the program is installed you can launch it by clicking on the black and white gear icon you should find on your desktop.

The first time you run the program (starting from version 4.0.0) a form appears asking you to insert your account information. Please visit http://www.coveringrepository.com if you don't own a subscription yet.
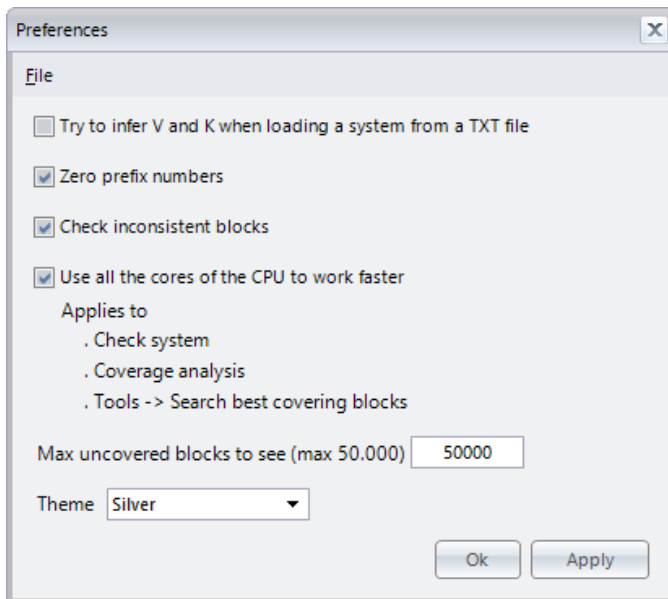

Insert your data and click the Test parameters button. If everything works as expected, a message stating parameters are ok will appear in the status bar. Click the Ok button to close the form.

Preferences

Preferences are the first thing you should check and modify to your liking.

Try to infer V and K when loading a system from a TXT file: if checked, the program will attempt to infer the values of V and K from the contents of the TXT file. They will be estimated based on available data, if possible.

Zero prefix numbers: if checked, all the grids and the reports will add a leading 0 (zero) to numbers less than 10. The suggestion is to keep this option checked to avoid strange behaviors when blocks are sorted alphabetically.

Check inconsistent blocks: if checked, the program will raise an error if a manually inserted block is wrong (block inconsistent with v and/or k parameters). Keep this checked to avoid strange behaviors.

Use all the cores of the CPU to work faster: some features of the program can take full advantage of the power of your CPU. Select this option to check systems faster, for example.

Max uncovered blocks to see (max 50.000): limits the number of uncovered blocks, if any, that you want to see in the relative grid.

Theme: the layout theme. Black, Light blue, Silver options are available. Choose one and click the Apply button to see how the interface colors change.

Loading a system

To load a system, simply click on File -> Open system, or use the Ctrl + O keyboard shortcut, then browse to the folder containing your file.
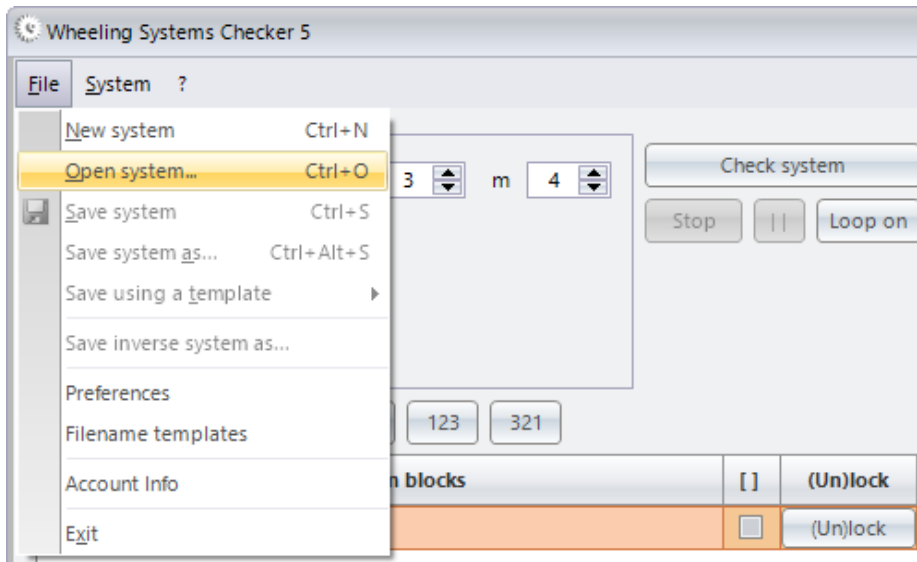
The file should be a normal text file with one block in each line. Numbers can be separated by one or more blank spaces (or tabs). Numbers can also be written with a zero prefix.

Example 1

01 02 05 07
01 03 05 08
01 05 06 07
02 03 04 06
02 04 06 08
03 04 07 08

Example 2

1 2 5 7
1 3 5 8
1 5 6 7
2 3 4 6
2 4 6 8
3 4 7 8

It is not necessary to write numbers in increasing order. The following blocks are also recognized:

1 7 5 2
8 3 5 1
5 1 7 6
4 3 2 6
6 8 4 2
4 8 3 7

As an alternative, you can copy blocks from your text file (Ctrl + C) and paste them into the program (Ctrl + V).

When the system is loaded, the blocks are visible in the left grid.

You can manually insert/edit/delete blocks. You can also lock/unlock them (more about this on the Tools page).

To edit a block, double click on it, modify it and then click Enter to confirm. To exit the edit mode without modifying anything, press Esc on the keyboard.

More options are available in the System menu (image below).
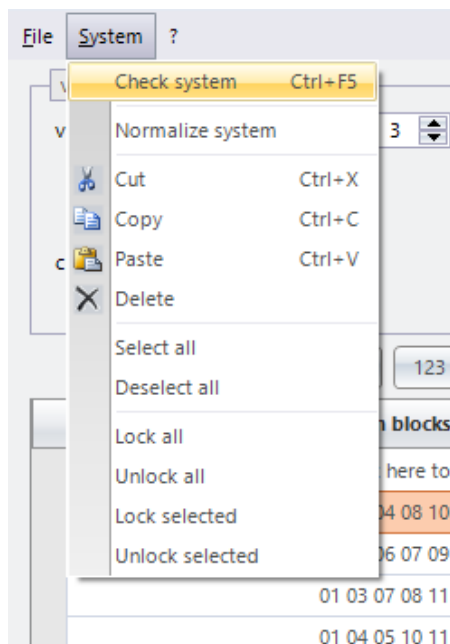
Normalize system changes the first block to 01 02 03 04 05… and all the others accordingly.

Filename templates

Filename templates are used to quickly save a system using a name generated from a predefined pattern. Templates can be modified, deleted, or newly created as needed.

To access the filename templates management window, click on File -> Filename Templates.
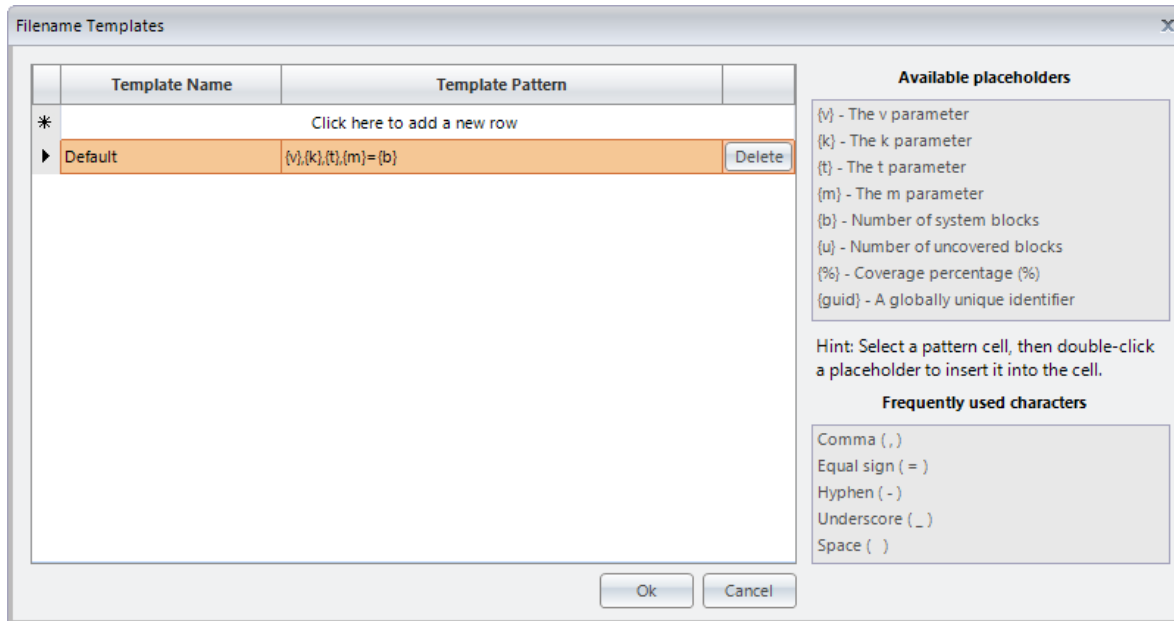
To create a new template, click to the left of the empty row (indicated by the text "Click here to add a new row") to activate the cell where you can enter your template's name.
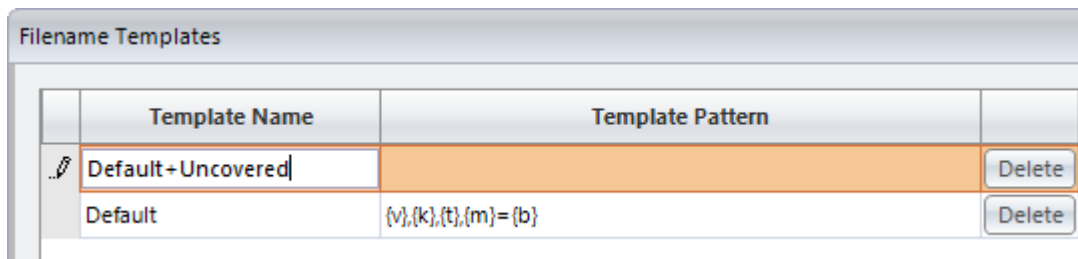
You can now move to the right column (by pressing Tab or clicking with the mouse) and manually enter the pattern, or — recommended — press Enter to add an empty template, select the pattern cell of the newly created template, and insert placeholders or special characters by double-clicking on the lists shown to the right of the grid.

Filename Templates

Template Name	Template Pattern	
✳	Click here to add a new row	
Default	{v},{k},{t},{m}={b}	Delete
▶ Default+Uncovered	{v},{k},{t},{m}={b}-{u}	Delete

Available placeholders

{v} - The v parameter
{k} - The k parameter
{t} - The t parameter
{m} - The m parameter
{b} - Number of system blocks
{u} - Number of uncovered blocks
{%} - Coverage percentage (%)
{guid} - A globally unique identifier

Hint: Select a pattern cell, then double-click a placeholder to insert it into the cell.

Frequently used characters

Comma (,)
Equal sign (=)
Hyphen (-)
Underscore (_)
Space ( )

Ok   Cancel

When you're done, click Ok to save.

Templates can then be used by selecting File -> Save using a template and choosing one of the patterns from the submenu.

File   System   ?

New system	Ctrl+N
Open system...	Ctrl+O
Save system	Ctrl+S
Save system as...	Ctrl+Alt+S
Save using a template	▶
Save inverse system as...	
Preferences	
Filename templates	
Account Info	
Exit	

Default
11,05,03,04=8

Default+Uncovered
11,05,03,04=8-14

3 ⏶ m 4 ⏶

Check system

Stop || Loop on

123   321

n blocks   [ ]   (Un)lock

here to add a new row

01 02 04 08 10   ☐   (Un)lock
01 03 06 07 09   ☐   (Un)lock
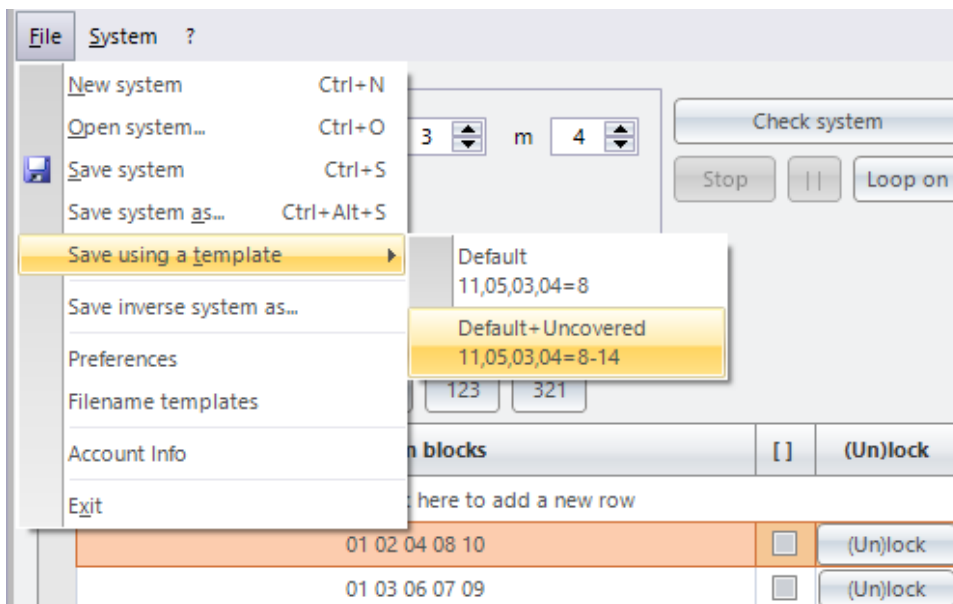
Checking a system

When the system is loaded, manually adjust the v, k, t and m parameters, if needed.

In the v, k, t, m settings panel you can see some information describing the current system.

k combinations: the number of combinations of v numbers taken k at a time.

m combinations: the number of combinations of v numbers taken m at a time.

covered by 1 block: the number of m-blocks covered by a single k-block (depends on t).

inverse: the corresponding inverse parameters. Black numbers mean you can save the current system as inverse (File -> Save inverse system as...). Grey numbers mean the inverse parameters are equal to the selected v, k, t, m parameters. Red "not available" means the selected parameters do not have a corresponding inverse set of parameters.

Finally, when you press the Check system button, the program starts.
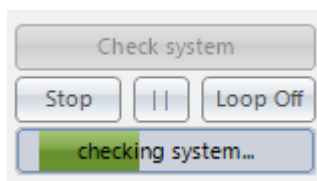
A green progress bar appears informing you the program is checking the system.

Press the Stop button to terminate, press the Pause button || to suspend.

When the task is finished, the Progress report panel shows the result.

100% covered

Progress panel
System blocks: 9
Uncovered: 0
Coverage %: 100,000000000000000%
Elapsed time: --

95,75% covered

Progress panel
System blocks: 8
Uncovered: 14
Coverage %: 95,757575757575752%
Elapsed time: --
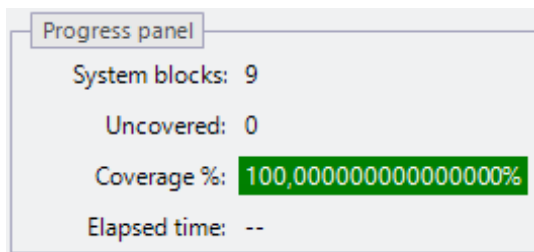
In the second case 14 blocks are not covered.

Uncovered blocks

When a system is not 100% covered, you can see which m-blocks are not covered by inspecting the Uncovered blocks grid. You can also open uncovered blocks in Notepad.

Progress panel

System blocks: 8	To process: 36
Uncovered: 14	In memory: 36
Coverage %: 95,757575757575752%	Processed: 36
Elapsed time: --	

Uncovered blocks   Coverage analysis   Tools   Additional tools   Covering tool

Open uncovered blocks in Notepad

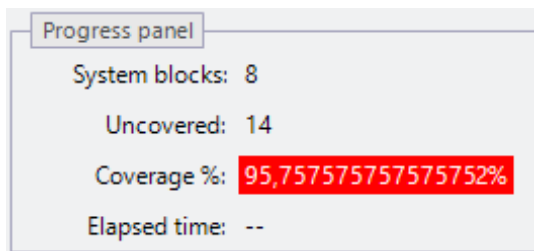| Uncovered blocks	
01 05 06 08	Add block
01 05 08 09	Add block
02 06 08 09	Add block
04 05 06 09	Add block
04 05 07 08	Add block
04 05 08 09	Add block
04 07 08 09	Add block
05 06 08 09	Add block
05 06 09 10	Add block
05 07 08 10	Add block
05 08 09 10	Add block
05 08 09 11	Add block
06 08 09 11	Add block
07 08 09 10	Add block

If k-blocks are the same length as m-blocks, the Add block buttons are enabled and you can add one or more of the uncovered blocks to the system blocks. In this case they are disabled because the current system parameters are 11, 05, 03, 04 and k (5) is not equal to m (4).

Coverage analysis

Coverage analysis tells you how many blocks are exclusively covered by each system block. From here you can delete one or more blocks. You can also lock/unlock blocks (more about this in the Tools page).

| Uncovered blocks | Coverage analysis | Tools | Additional tools | Covering tools | Shrinking tools |

| Coverage analysis | Open analysis in Notepad |

System blocks	Blocks covered		
05 06 07 08 09	14	Delete	(Un)lock
02 03 05 07 09	14	Delete	(Un)lock
01 03 07 08 11	14	Delete	(Un)lock
01 03 06 07 09	14	Delete	(Un)lock
02 05 06 07 11	14	Delete	(Un)lock
01 02 04 08 10	17	Delete	(Un)lock
02 04 09 10 11	17	Delete	(Un)lock
03 04 06 08 10	18	Delete	(Un)lock
01 04 05 10 11	18	Delete	(Un)lock

When you delete a block, the whole situation changes so you should perform a new coverage analysis whenever you delete a block. In the sample above, if you delete the first two blocks at the same time, the final uncovered blocks will not be 28 but 34. In fact, if you delete the first one (05 06 07 08 09) and do a new coverage analysis, the second one (02 03 05 07 09) will exclusively cover 20 blocks instead of 14.

You can open the report in Notepad by clicking the Open analysis in Notepad button.

Tools

This function can help you reduce the number of uncovered blocks. Eventually **locked blocks** are not taken into consideration and will not change throughout the process.

Open coverage improvements report in Notepad: a text file, reporting all the possible improvements, opens. Activating this option automatically sets Loop Off.

Max number of improvements: to speed up the reduction process you can set a limit of improvements to find after which the procedure starts again. To use in conjunction with Loop On.

Sort blocks after each best block replacement: you can sort system blocks in 4 ways after each replacement – alphabetical, reverse alphabetical, numerical, reverse numerical.

Numbers shared with each analyzed block: if k = 5 and 4 is selected, the program will search for improvements between the blocks which have 4 numbers in common with the current analyzed block.

Use parallel to initialize: the initialization phase can be speeded up by selecting this option which will use all the cores of your CPU (overrides the global settings you find in preferences).

When finished and changed: when you click stop or the search task ends (and only if the system has changed) you can decide to open the final blocks in Notepad or to replace the system blocks in the main grid.

Search best covering blocks

Remaining numbers in uncovered blocks: 10/11 - C(10,5)=252

Best so far: 14

[Search blocks]   [Open best blocks in Notepad]

Best blocks	Covered blocks	Weight	
05 06 07 08 09	14	2.376	Add block
04 05 06 08 09	12	1.224	Add block
01 05 06 08 09	10	1.224	Add block
02 05 06 08 09	10	1.800	Add block
01 05 07 08 09	10	2.448	Add block
02 05 07 08 09	10	7.632	Add block
01 04 05 08 09	9	1.872	Add block
01 02 05 08 09	7	1.872	Add block
01 02 05 06 09	6	2.448	Add block
01 02 05 06 08	4	1.872	Add block
01 02 04 05 08	4	7.704	Add block
01 02 04 08 09	4	7.704	Add block
01 02 04 05 06	2	2.520	Add block

With this function you can search for the best covering blocks. If the remaining numbers in the uncovered blocks are less than the total numbers (v parameter), the search is restricted to the k-combinations generated by the remaining numbers; in this case, 10 out of 11 numbers remain in the uncovered blocks, resulting in a search for only 252 k-combinations instead of 462.

To add a block to the system blocks simply click the Add block button.

Sometimes multiple blocks cover the same number of blocks. The weight column can help you decide which block is the best choice. Usually the lighter block is the best candidate in order to keep the system balanced for subsequent improvements.

You can open the search results in Notepad by clicking the Open best blocks in Notepad button.

Additional tools

Weights analysis

Block	Weight	
05 06 07 08 09	2.376	Delete
02 03 05 07 09	2.376	Delete
01 03 06 07 09	2.376	Delete
01 02 04 08 10	2.304	Delete
02 05 06 07 11	1.800	Delete
02 04 09 10 11	1.728	Delete
01 04 05 10 11	1.728	Delete
01 03 07 08 11	1.224	Delete
03 04 06 08 10	1.152	Delete

The weight of a block is a value which depends on how many numbers a block has in common with all the other system blocks. The higher the weight, the higher the possibility that the block is a good candidate for deletion. Usually the heaviest blocks are the ones which exclusively cover the lowest number of blocks but sometimes they are not.

To delete one or more blocks, click on the corresponding Delete button.

Multi-covered blocks

Block	Times cover...		
03 07 08 09	4	Del. covering blocks	View covering blocks
02 06 07 09	4	Del. covering blocks	View covering blocks
05 06 07 09	4	Del. covering blocks	View covering blocks
03 06 07 08	4	Del. covering blocks	View covering blocks
03 05 06 07	4	Del. covering blocks	View covering blocks
04 08 10 11	4	Del. covering blocks	View covering blocks
01 02 03 07	3	Del. covering blocks	View covering blocks
01 04 08 10	3	Del. covering blocks	View covering blocks
02 07 09 11	3	Del. covering blocks	View covering blocks
01 05 06 07	3	Del. covering blocks	View covering blocks

This report tells you which are the system blocks that are covered multiple times and which are the covering blocks. Click Del. covering blocks button to delete all the covering blocks. Click on View covering blocks button to open a report in Notepad. The numbers next to the covering blocks indicate how many numbers they have in common with the covered block and their weight.

```
11,05,03,04=9_tmp598C.tmp - Notepad

File   Edit   Format   View   Help
Covered block
03 07 08 09

Covering blocks
01 03 06 07 09 - 3 - 2.376
01 03 07 08 11 - 3 - 1.224
02 03 05 07 09 - 3 - 2.376
05 06 07 08 09 - 3 - 2.376
```

System analysis

File   Edit   Format   View   Help

```
                        0     1     2     3     4

01 02 04 08 10          4     1     3        -  2.304
01 03 06 07 09          3     2     3        -  2.376
01 03 07 08 11          1     6     1        -  1.224
01 04 05 10 11          3     3     2        -  1.728
02 03 05 07 09          3     2     3        -  2.376
02 04 09 10 11          3     3     2        -  1.728
02 05 06 07 11          2     4     2        -  1.800
03 04 06 08 10          2     5     1        -  1.152
05 06 07 08 09          3     2     3        -  2.376

----------------------------------------------------
Numerical distribution
01 - 4
02 - 4
03 - 4
04 - 4
05 - 4
06 - 4
07 - 5
08 - 4
09 - 4
10 - 4
11 - 4

----------------------------------------------------
Normalized system blocks
01 02 03 04 05
01 03 04 08 11
01 05 07 10 11
01 06 07 09 10
02 03 04 09 11
02 06 07 08 11
02 07 08 09 10
03 04 05 06 10
05 06 07 08 09
```

The first section of the report tells you how many numbers each block has in common with the other blocks and its weight. In the example above, the block 01 02 04 08 10 has 1 number in common with 4 blocks, 2 numbers in common with 1 block and 3 numbers in common with 3 blocks. Its weight is 2.304.

The second section tells you how many times a number is present within the system blocks.

The third section shows a normalized version of the original system.
The first block is changed to be in the form 01 02 03 04 05 and all the others are changed accordingly.

Covering tools

IMPORTANT: always check your system before using the covering tools in order to start working with the right number of uncovered blocks in memory. Not doing so could lead to unexpected results.

The covering tools can be used to try to improve the partial coverage of a not yet complete system, to complete a system or to generate a new one from scratch. In some cases, the program can generate systems from scratch that are remarkably close to existing records, but this is not always the case, it depends on the values of k, t and m parameters.

Basically, the program will try to improve a system by deleting a block (the one considered the best candidate to remove), then adding a new block and finally trying variants of the remaining blocks in search of a better coverage.

From now on, these operations will be referred to as "main cycle".

General options

Try to improve coverage first: when checked, the program will try to improve the existing coverage by working on the current system blocks before starting the main cycle.

Add a new block first: when checked, the program will add a new block before starting the main cycle.

Try to improve coverage first when a new block is added: when checked, the program will try to improve the existing coverage every time a new block is added and then continue with the main cycle.

Coverage % to reach before deep search: in this field you can set a coverage percentage (from 0 to 100) that you want to reach before the program starts the main cycle. In practice, blocks are added to the system until the indicated coverage percentage is reached, after which the main cycle begins. By setting the value to 100, the program will not do anything else once this percentage is reached; the value 100 can still have its use, as you can read below.

Max nr. Of system blocks to reach (0 = no limit): this field allows you to limit the maximum number of blocks you want the program to reach. If left at zero it will not be considered and blocks will be added until full coverage is achieved. For example, if you have a 10 blocks system and set this parameter to 20, the program will add 10 blocks. If you want to reach a certain number of blocks, instead of a percentage, before the program starts the main cycle, you can set this parameter and set the value Coverage % to reach before deep search to 100. Once the desired number of blocks is reached, the program will stop, opening blocks in a text file or replacing the main system blocks based on your choice in the "Additional options" section; you can then reset the percentage to zero and restart.

Limit deep search to the first "n" blocks (0 = no limit): if the system consists of many blocks, this parameter can be useful. It allows you to limit the number of blocks on which to attempt a replacement (if zero, it will not be taken into consideration). In a 100 blocks system for example, with Search depth = 3 and a limit of 10, the program will try to replace 3 blocks at a time by examining only the 10 heaviest blocks in the system.

Search depth: the search depth is nothing more than the number of blocks the program will try to replace simultaneously. 1 means that it will try to remove one block at a time and replace it with a, hopefully, better one. The higher the value, the longer the program will take to complete the search. The existing blocks are first reordered by "weight" (not by coverage) then the program tries to replace the heavier ones first. The number of steps needed to complete the cycle is indicated next to the Steps label.

New block options

An important thing to understand is that a new block does not refer only to the action of adding a block to the system, incrementing its value by 1; it also refers to the blocks added in the main cycle every time a block is replaced. These settings therefore refer to both situations.

Also, a new block is not added based on the number of uncovered blocks it covers but on its weight, which is a calculated value based on how many numbers the block shares with all the blocks composing the system.

Classic mode: in this mode the program will calculate the weight of all the k-combinations and pick the one with the lowest weight, covering some of the uncovered blocks. Suitable for systems with a limited number of k-combinations.

Alternative mode (recommended): this mode allows you to manipulate systems with a high number of k-combinations by limiting a range of numbers to use to compose the blocks to add. Every time the system changes, the program internally updates a ranking of the best candidate numbers to use and searches for the best block within the range you select with min and max pick.

In a system with v=40 and k=15, for example, there are c(40,15) = 40.225.345.056 k combinations.
With a min pick = 15 and a max pick = 21 you limit your search to c(21,15) = 54.264 k-combinations. The remaining 19 numbers (40-21=19) will be tried in the main cycle as variants. If an improvement is found, the ranking changes and the top 21 numbers to use will probably change as well.

There's a difference if you select min pick = 15/max pick = 21 or min pick = 21/max pick = 21.

E.g.: Min pick = 15 – Max pick = 21
The program will add the block with the lowest weight which covers most blocks for each pick value from 15 to 21.
c(15,15)=1 – best weight 940 – covers 800
c(15,14) combined with the 16th best candidate=15 – best weight 950 – covers 780
c(16,14) combined with the 17th best candidate =120 – best weight 930 – covers 820
c(17,14) combined with the 18th best candidate =680 – **best weight 940 – covers 850**
c(18,14) combined with the 19th best candidate =3.060 – best weight 910 – covers 810
c(19,14) combined with the 20th best candidate =11.628 – best weight 930 – covers 830
c(20,14) combined with the 21st best candidate =38.760 – best weight 920 – covers 830
Total attempts 1 + 15 + 120 + 680 + 3.060 + 11.628 + 38.760=54.264
The block with weight 940 which covers 850 blocks will be used.

E.g.: Min pick = 21 – Max pick = 21
c(21,15)=54.264 – best weight 910 – covers 810
The block with weight 910 which covers 810 blocks will be used.

If you check the box "Guarantee some blocks are covered" you force the program to search for a block which covers at least some of the uncovered blocks. This could slow down the process dramatically, probably forcing the program to silently scale up your max pick choice.

If you don't check it **(recommended)** and the system is left with a low number of blocks to cover, it's very likely that a block covering 0 (zero) blocks will be added. Don't worry, the main cycle will use it anyway and probably find some improvements; if this is not the case, you can remove it manually and add a better one by inspecting the uncovered blocks or by using the "Search best covering blocks" feature, stopping the research after "n" scanned blocks if there are too many k-combinations to analyze.

Please note: when the coverage percentage exceeds 99.99% and the result of multiplying the number of k-combinations — calculated using the remaining numbers in the uncovered blocks — by the number of uncovered blocks is less than 100 million, the program will initiate an exhaustive search through all these k-combinations. It will then select the one that covers the most uncovered blocks. If multiple combinations cover the same number of blocks, the program will prioritize the one with the best weight.

Deep search options

Max number of improvements: to speed up the reduction process you can set a limit of improvements to find after which the procedure starts again. This functionality conflicts with the subsequent options, which will therefore be disabled for any value greater than zero and ignored in the processing phase.

When more blocks with the same coverage are found: in the main cycle, when the program tries to use variants of a block, more variants covering the same number of blocks could be found. You can decide to let the program use the first block it finds or to calculate the weight of these variants and use the one that weighs less.

Variant k-x where x =: A variant is a block taken "k-x" numbers at a time and completed with all the other numbers (depends on v parameter) "x" at a time. You specify this "x" value. Minimum value is 1 and maximum value is 5.

E.g.: In a 10,05,03,05 system, with "x=1", the block 01 02 03 04 05 would produce these variants:
01 02 03 04 06
01 02 03 04 07
01 02 03 04 08
01 02 03 04 09
01 02 03 04 10
01 02 03 05 06
01 02 03 05 07

and so on up to

02 03 04 05 06
02 03 04 05 07
02 03 04 05 08
02 03 04 05 09
02 03 04 05 10

The higher the "x", the slower the program will run because it will have to try many more variants. Furthermore, when you select 2, variants with k-1 and k-2 will be tried; if you select 3, variants with k-1, k-2 and k-3 will be tried and so on.

Play sound on improvement: select to hear a sound whenever an improvement is found.

Play sound when system is 100% covered: select to hear a sound when processing ends and your system is 100% covered.

Use parallel to initialize: the initialization phase can be speeded up by selecting this option which will use all the cores of your CPU (overrides the global settings you find in preferences).

When finished and changed: when you click stop or the main cycle ends (and only if the system has changed) you can decide to open the final blocks in Notepad or to replace the system blocks in the main grid.

Resize button: resize the form to the minimum size for the Covering Tools.

Final note
Last but not least, sometimes, after the improvements research phase ends, one or more blocks covering 0 (zero) blocks may remain; in this case the program will automatically remove these blocks from the system.

Shrinking tools

Shrinking tools are a new feature introduced in version 4.0.0 to help you reduce the number of blocks of your systems. Speed and result depend on the v, k, t, m parameters and the number of blocks composing the system. As a rule, the higher the number of blocks covered by a single block, the slower the program will be able to find a way, if any, to shrink.

Please note: the program will not check if your system is 100% covered when you press the Shrink button. This means it will possibly stop at the end of the initialization task if you try to shrink an incomplete system and no explorable path is found.

From version 5.1.0 onwards, when attempting to reduce a system that is not 100% covered, the shrinker displays the number of uncovered blocks and any reductions performed during processing, **subject to prior verification of the system**. Additionally, an option has been introduced to terminate the program if the number of uncovered blocks reaches zero.

The following image shows the Shrinking tools interface.

Progress panel

Blocks: the current number of blocks. When you start shrinking it is equal to the original system blocks.

Uncovered: the current number of uncovered blocks, available only after performing a system check.

Changed: number of changed blocks compared to the original system blocks, reset after each improvement.

Attempts: number of performed attempts (substitutions), reset after a new block is changed.

Steps: in the initialization phase it corresponds to the number of analyzed original system blocks; in the shrinking phase it indicates the number of steps taken within the system blocks while searching for a new attempt. Reset after each improvement.

Seed
As many other programs do, the attempts made while searching for an improvement are based on casualty, so an initial seed is set to try to vary the result. If the initial system is the same, as well as the initial sort (more about this later), using the same seed will lead you to the same result.

Default seed: a fixed internal seed different from zero.
Random seed: a random seed changing every time you click the Shrink button.
User seed: you can use this seed to replicate a previous result or to finally take advantage of your lucky numbers!

Initial sort of system blocks
Here you can decide how to initially sort the system blocks. Whatever type of sort you choose, the program will always remove duplicates first.

No sort: the blocks remain untouched.
Alphabetical: the blocks are sorted as they were strings, which means these 6 blocks

03 04 07 08
01 05 06 07
02 03 04 06
01 02 05 07
01 03 05 08
02 04 06 08

would be sorted this way

01 02 05 07
01 03 05 08
01 05 06 07
02 03 04 06
02 04 06 08
03 04 07 08

Numerical: when you load a system, all the blocks are translated to their correspondent binary representation.

Block 1 2 3 4 corresponds to the 8-bits representation 00001111 which, in turn, is decimal number 15.
$2^0 + 2^1 + 2^2 + 2^3 = 1+2+4+8 = 15$

so the previous 6 blocks would be sorted like this

02 03 04 06 => 46
01 02 05 07 => 83
01 05 06 07 => 113
01 03 05 08 => 149
02 04 06 08 => 170
03 04 07 08 => 204

Shuffle: the blocks are randomly shuffled.

Additional options
Stop if there are no improvements in "n" minutes (0 = no limit): You have the option to specify a time limit, in minutes, to automatically stop the search for improvements. The "Elapsed time" label indicates the amount of time that has elapsed since the shrinking phase commenced or since the most recent improvement was discovered.
Play sound on improvement: select to hear an annoying sound whenever an improvement is found (but very satisfying sometimes).
Stop when the number of uncovered blocks reaches zero: enable this option to automatically stop the process once all

uncovered blocks have been covered, meaning the system has reached 100% coverage.

Use parallel to initialize: the initialization phase can be speeded up by selecting this option which will use all the cores of your CPU (overrides the global settings you find in preferences).

When finished and improved: when you click stop or when no more attempts are available, or a preset time limit has been reached (and only if improvements were found) you can decide to open the final blocks in Notepad or to replace the system blocks in the main grid.

View blocks in Notepad when finished, changed but not improved: when you click stop or when no more attempts are available, or a preset time limit has been reached (and no improvements were found but some blocks were changed) you can decide to open the final blocks in Notepad.

Resize button: resize the form to the minimum size for the Shrinking Tools.

Very often the results tend to converge, regardless of the selected parameters, so the suggestion is to run more than one session of the program each with a unique set of parameters.

The number of sessions that make sense to launch depends on the number of CPU cores and RAM available.

If a session works much better than the other ones, stop all the sessions, copy the blocks from the best session to each session and restart without touching the parameters.

If no attempts are found during the initialization, the program stops immediately and the shrinking phase does not even begins.