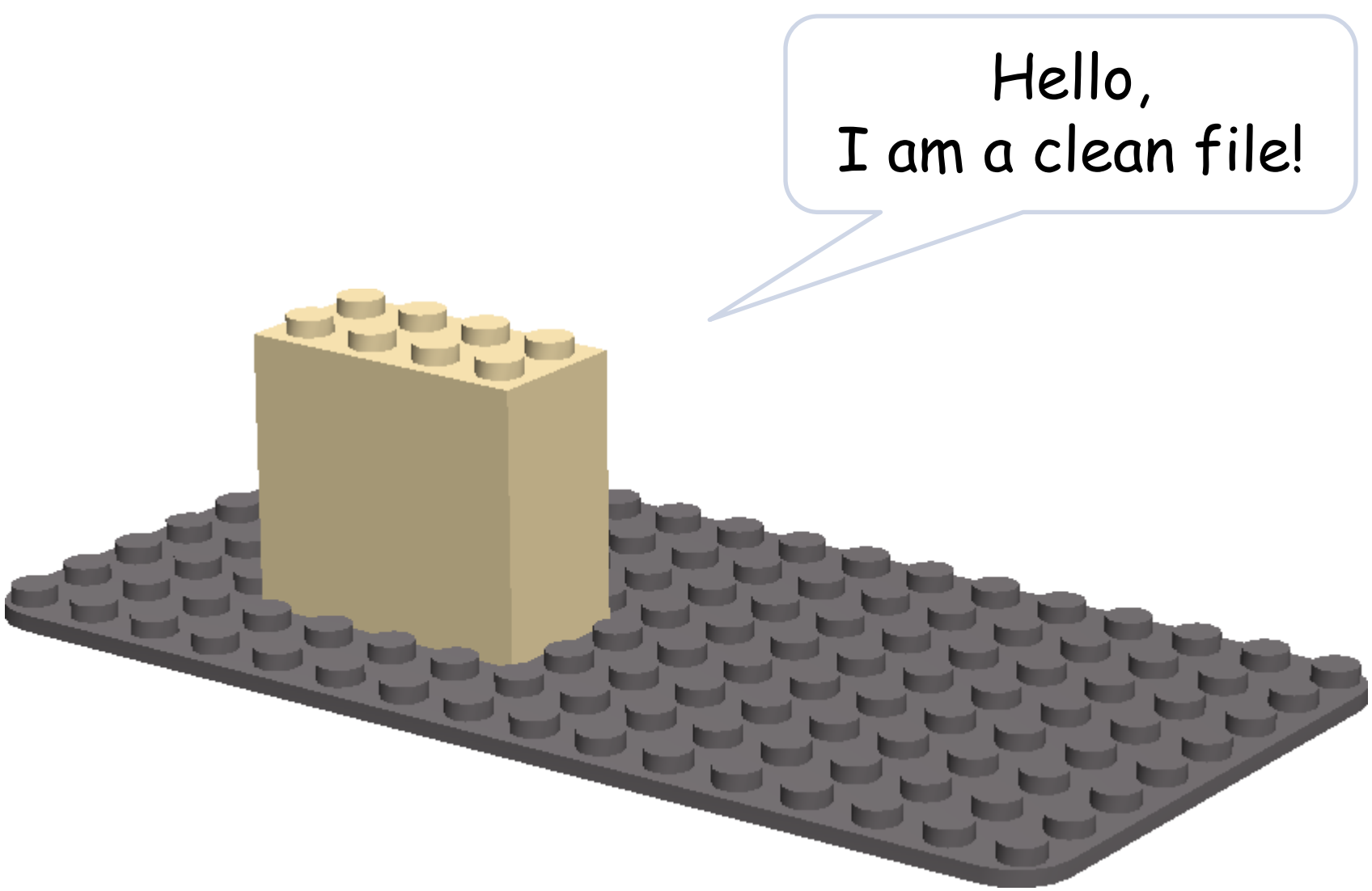
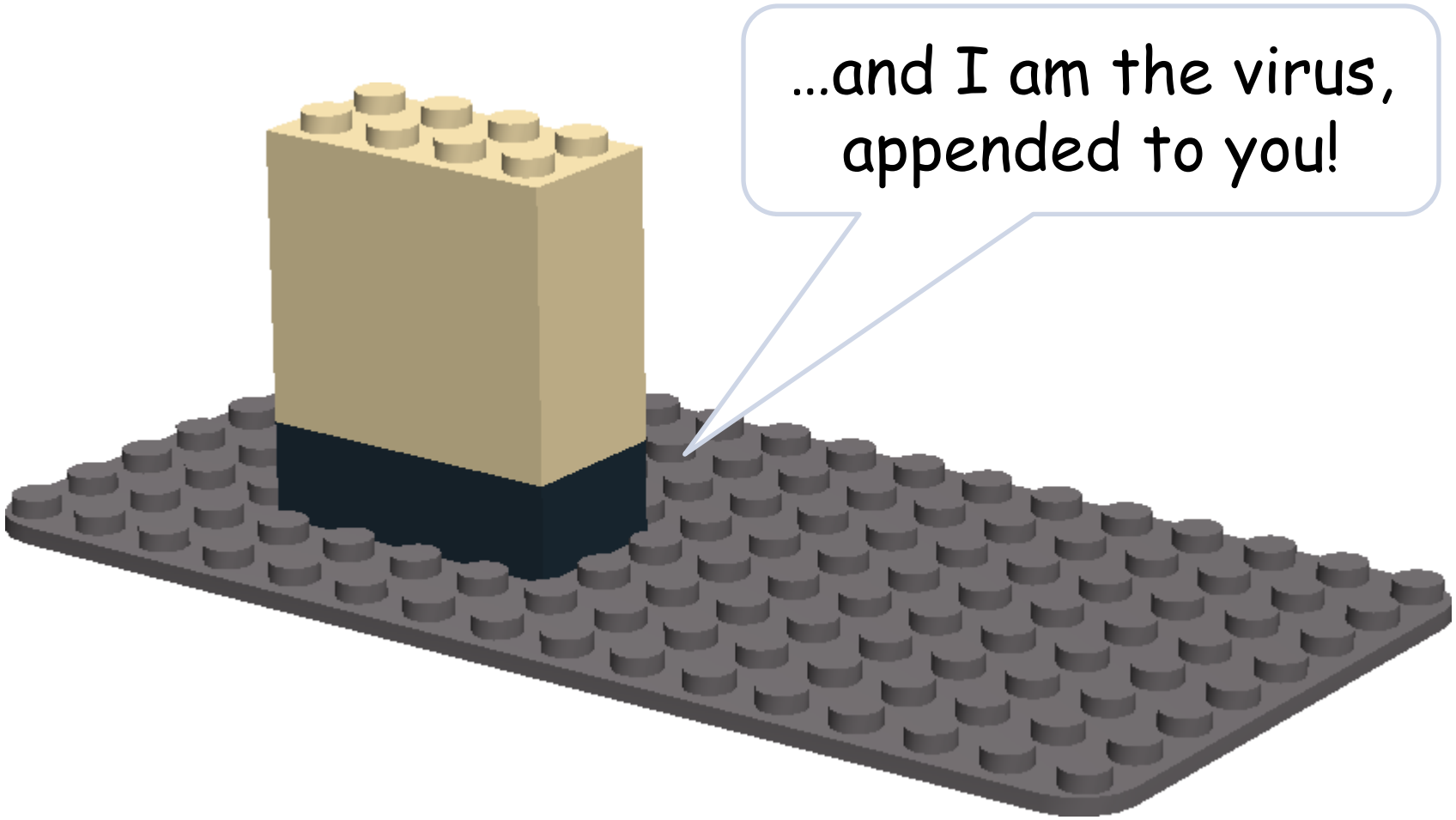
Surf, shop, bank, browse, download music and chat safely online with AVG

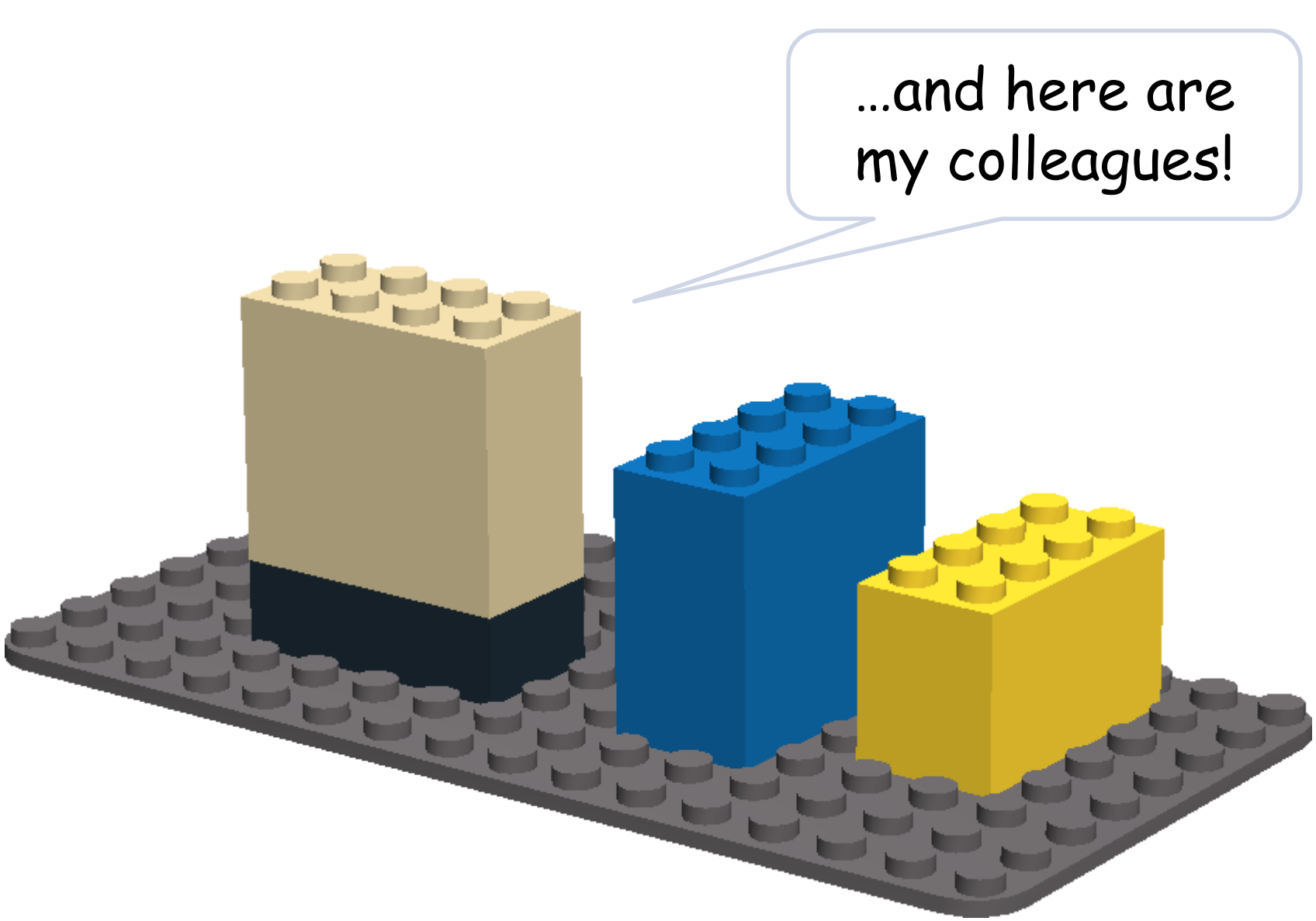
ENTROPY

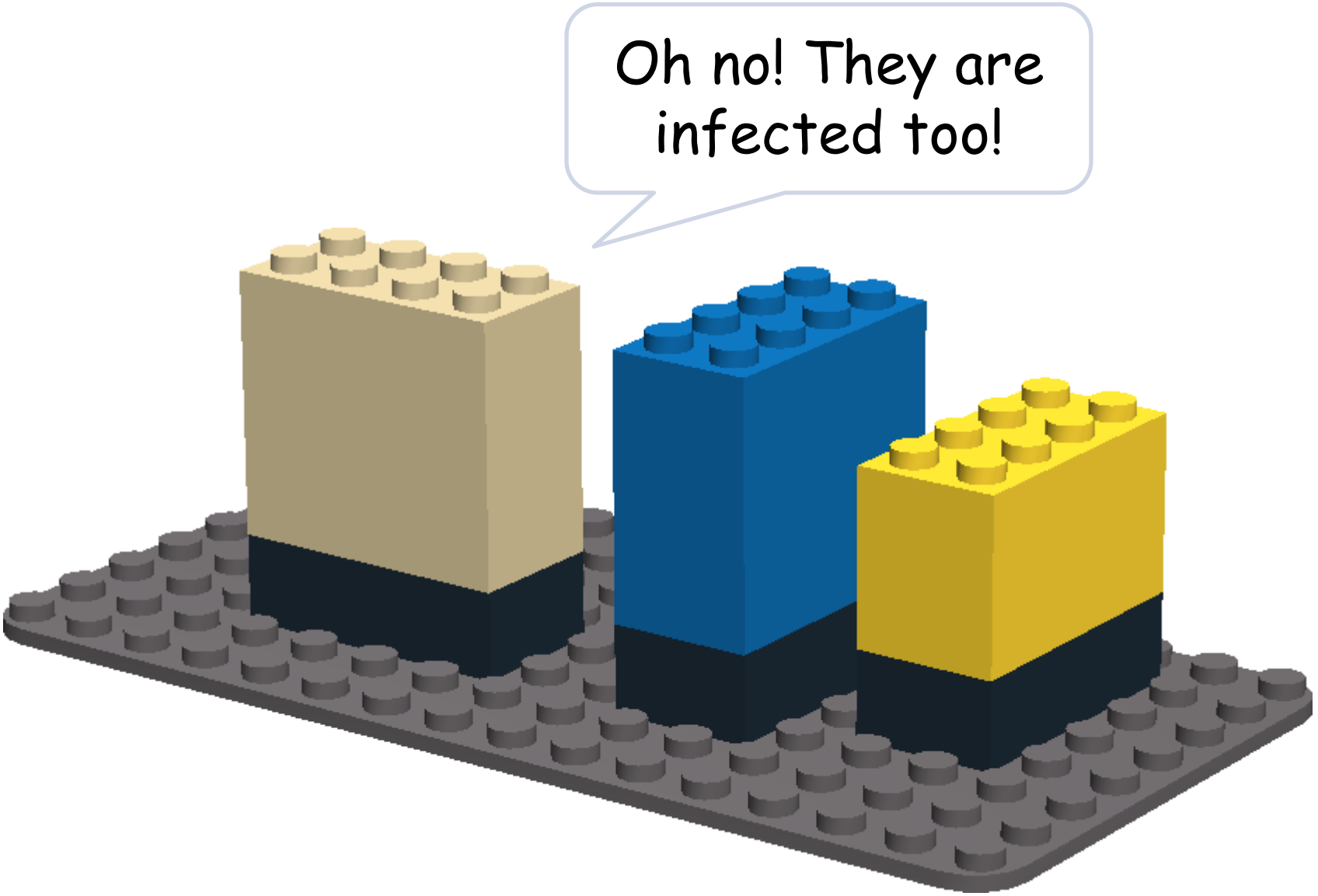
the new vision

Zdeněk Breitenbacher
zdenek.breitenbacher@avg.com

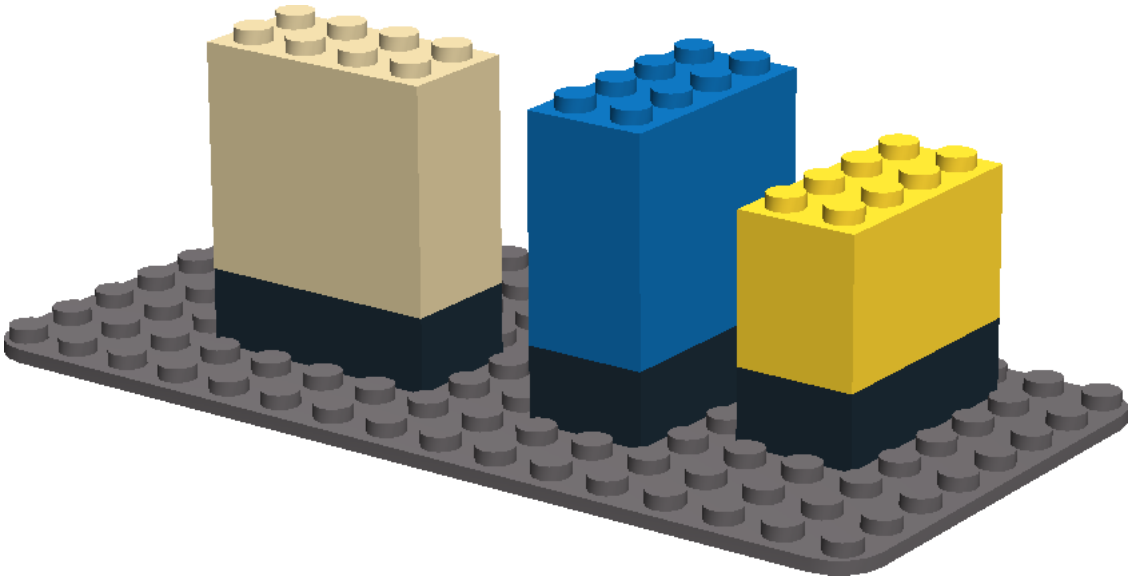

Hello,
I am a clean file!


…and I am the virus,
appended to you!

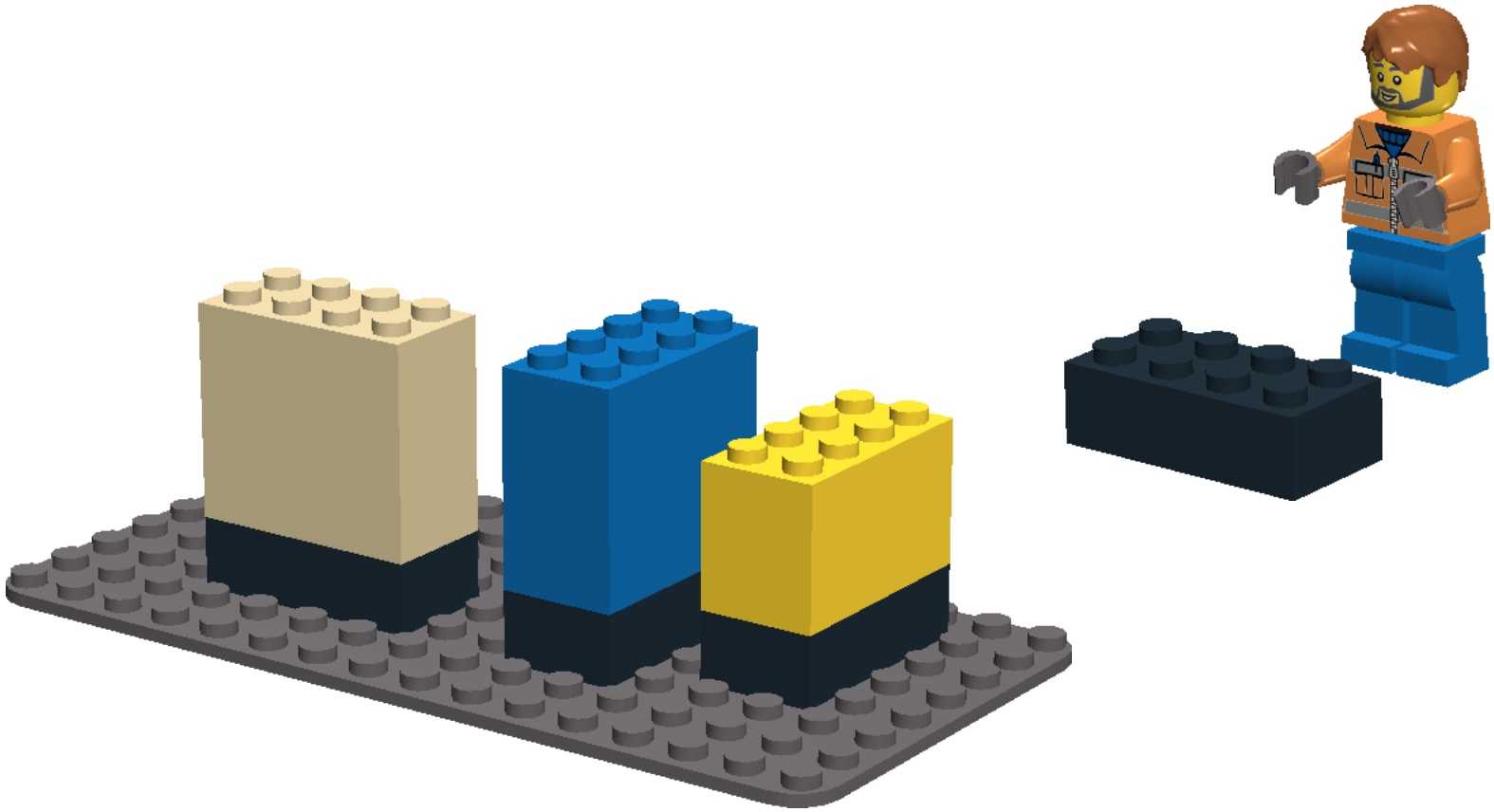

...and here are my colleagues!


Oh no! They are infected too!

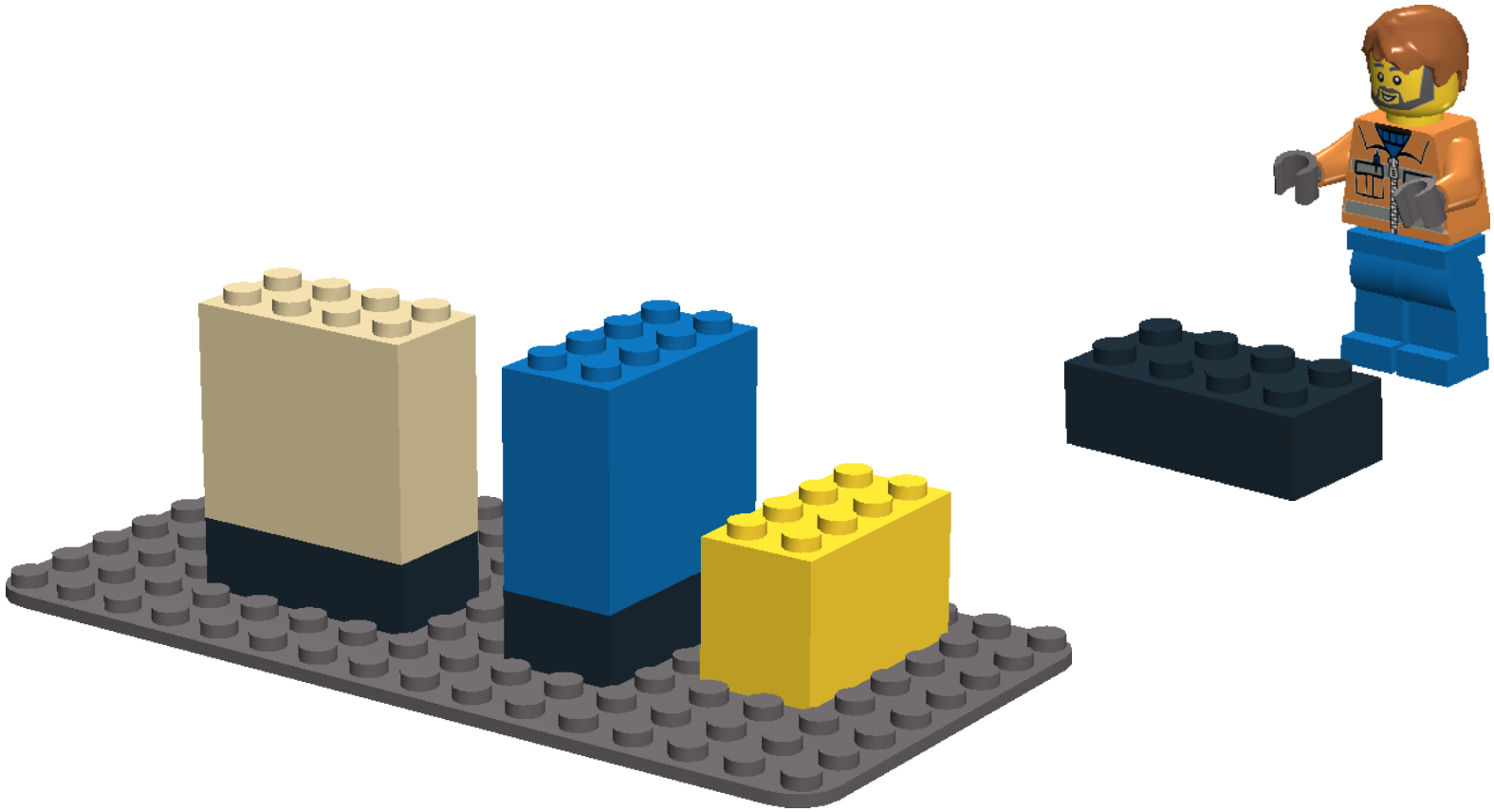
Fortunately, here is Joe, the Virus Fighter…



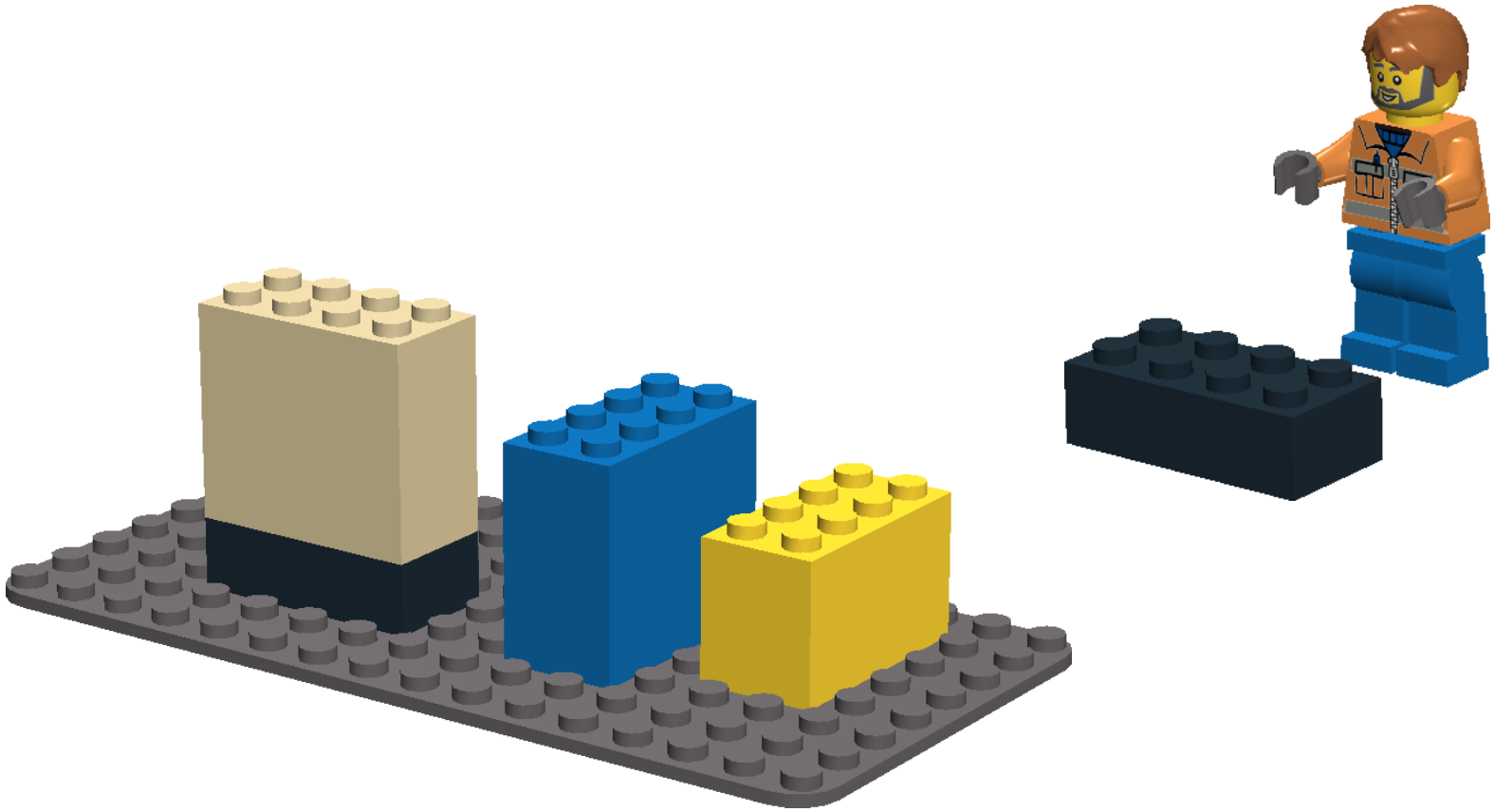
…and he has the Definition!



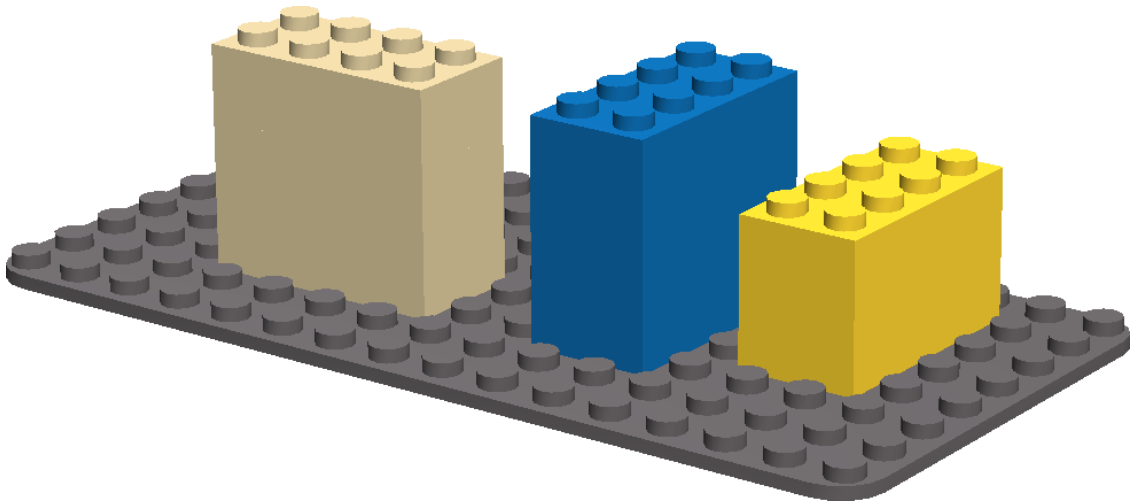
So he has done…



So he has done with the virus…

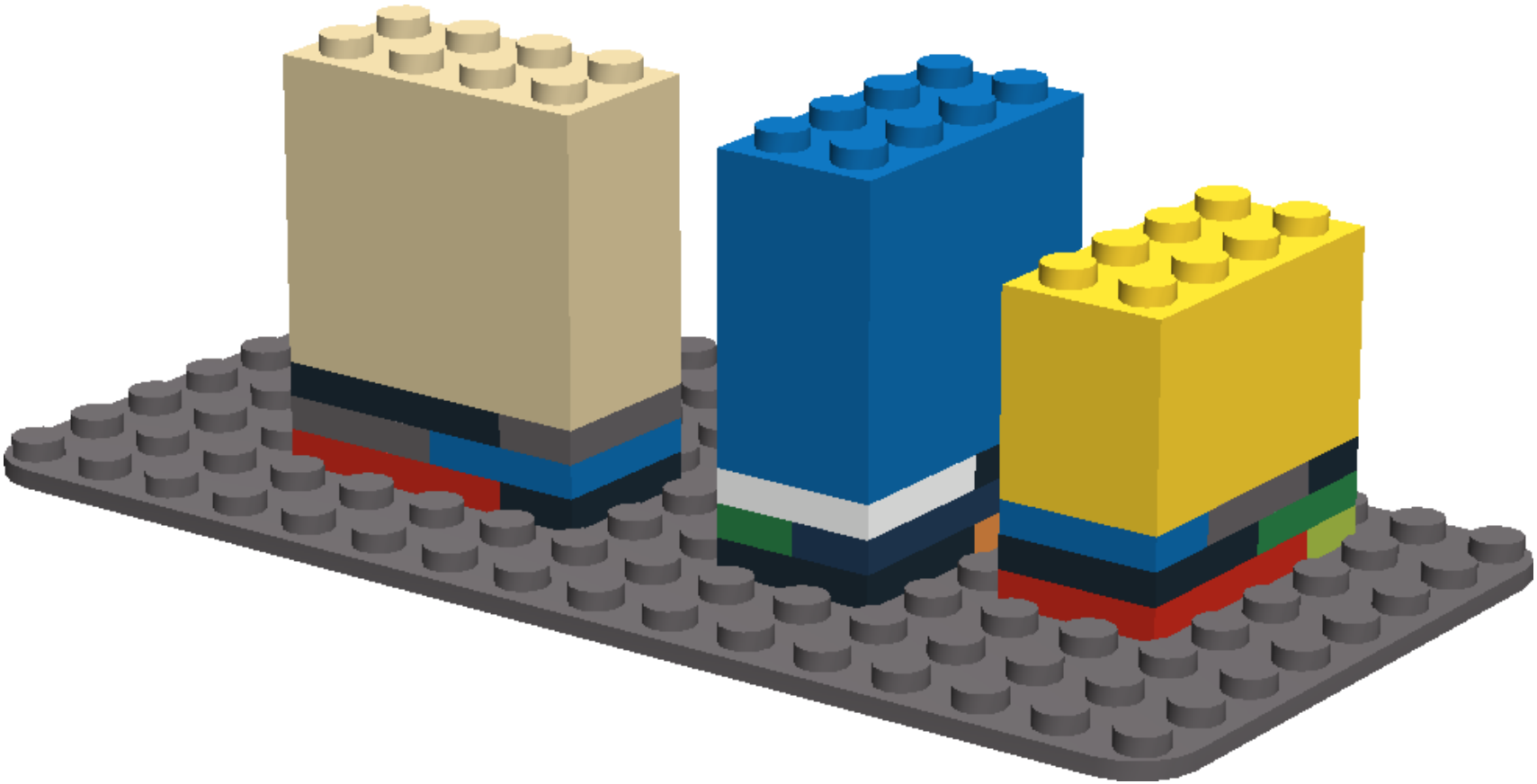


So he has done with the virus easily…

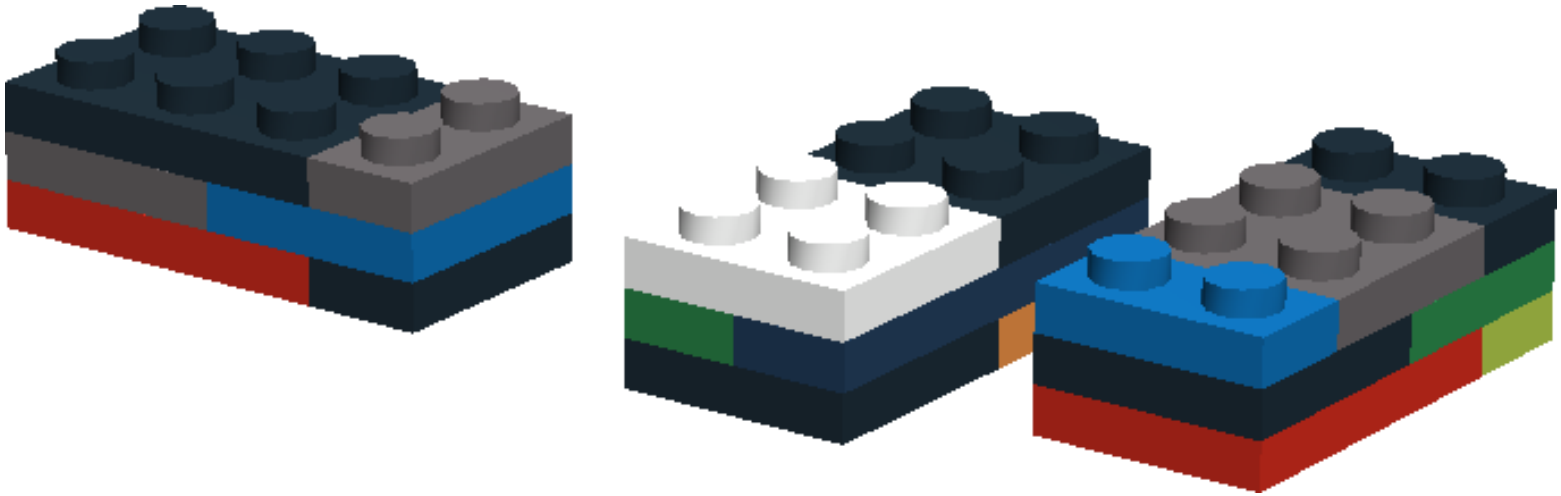


But what is ***this***???

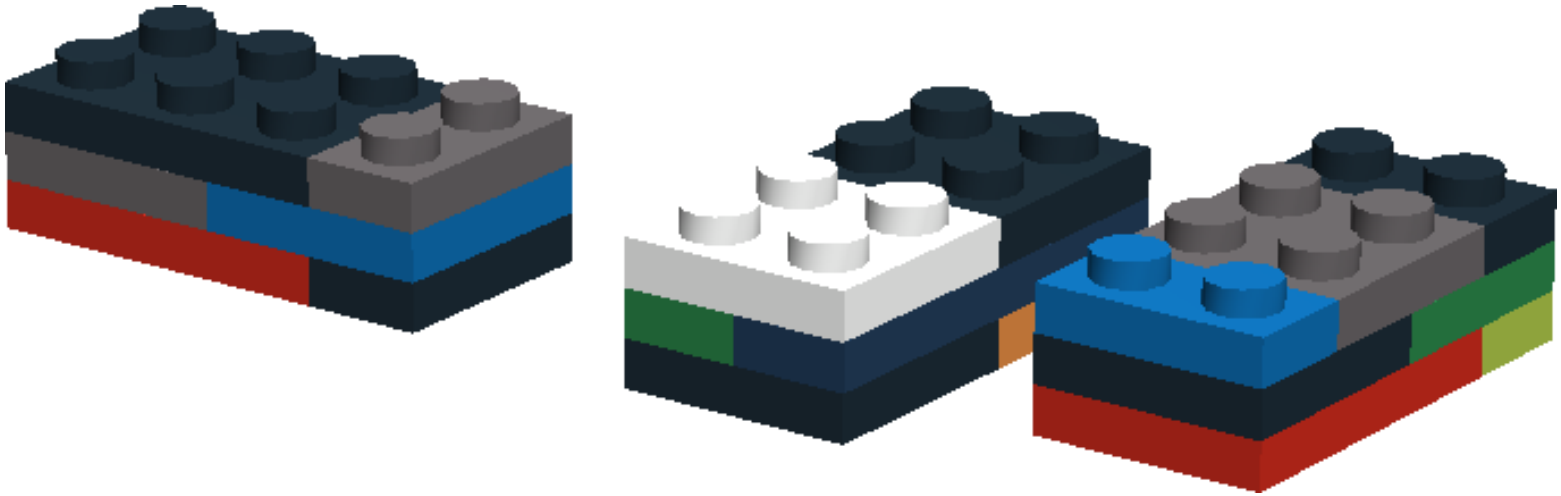
But what is ***this***???



But what is ***this***???

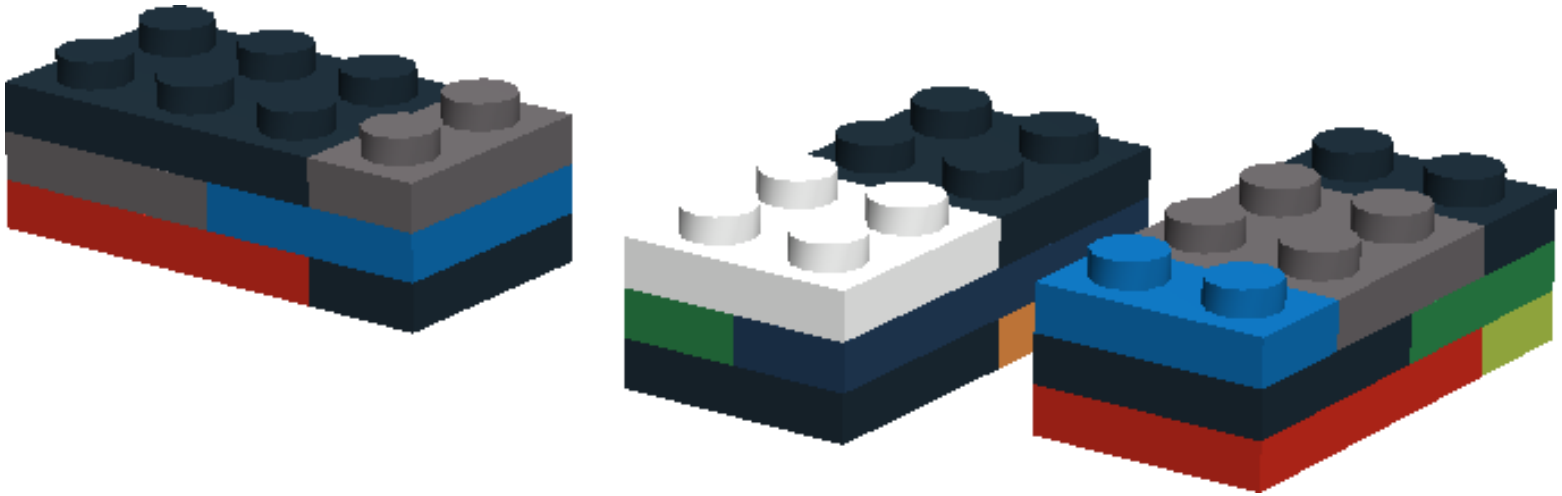


Yes, this is a ***new virus***!



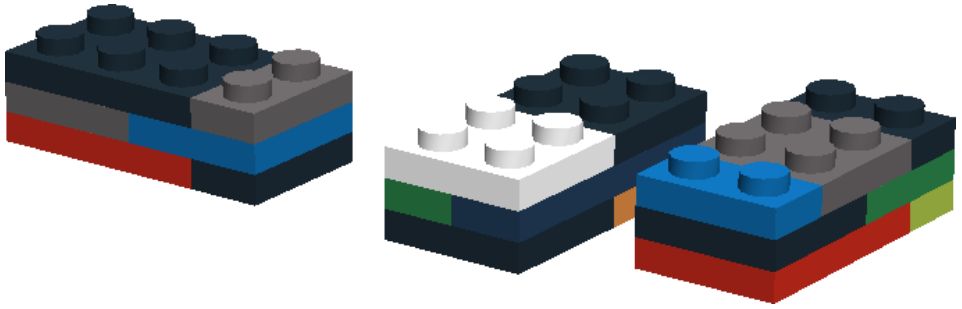
Yes, this is a ***new virus***!

a POLYMORPHIC virus!

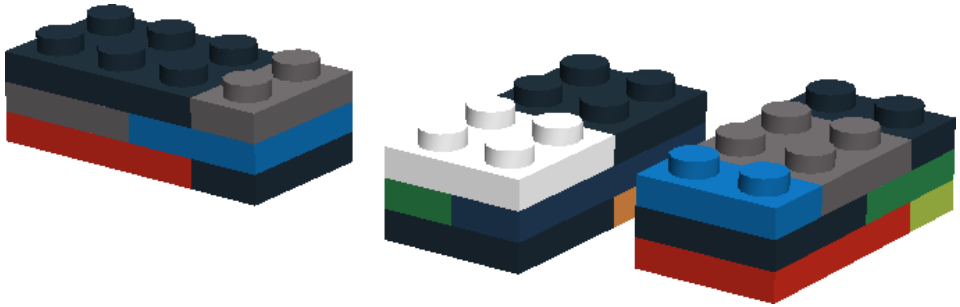


???

Which definition
is the proper one?



None of them!



The last year brought a lot of news in the field of malware evolution.

Polymorphic malware now becomes the standard.

The detection is getting more and more difficult.

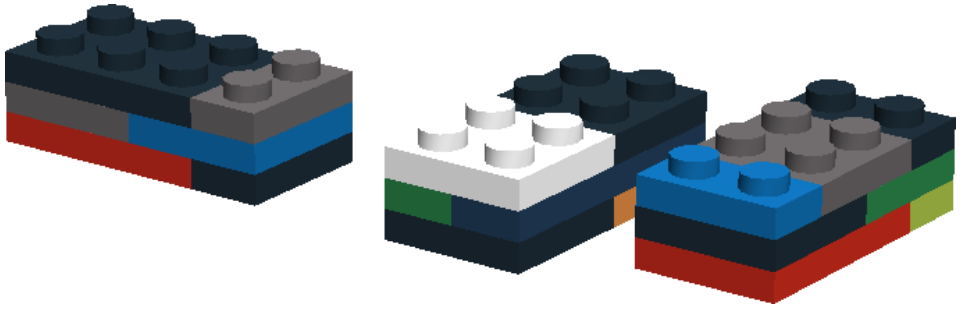
Each copy of polymorphic malware is ***totally different*** in a binary view, but we still can find some characteristics, which remain always the same.

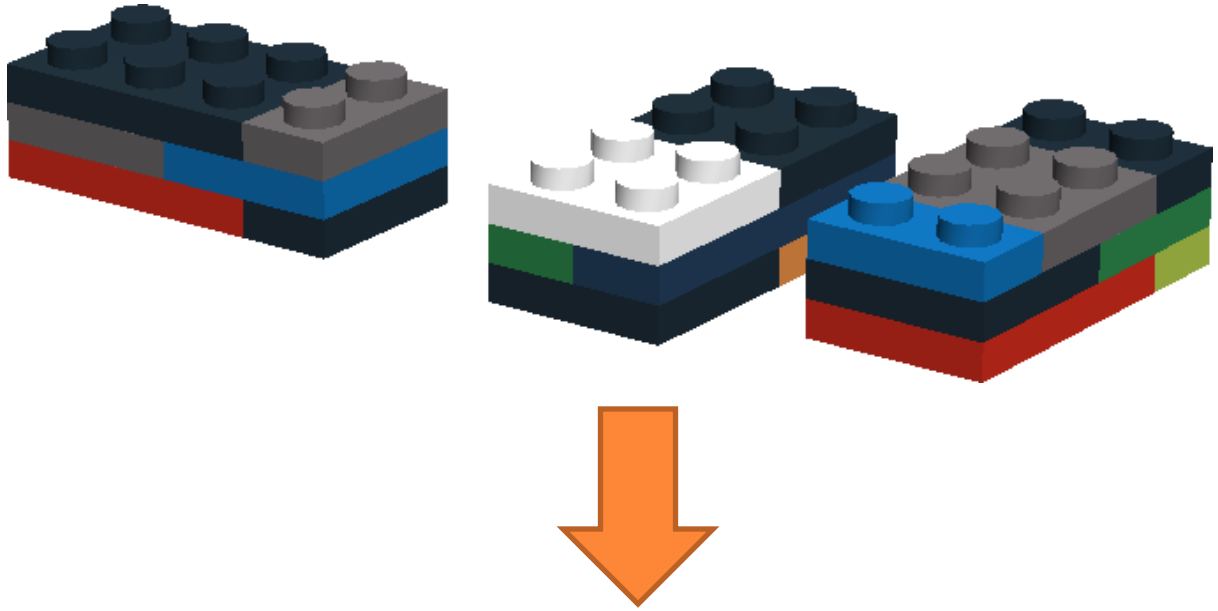
We only have to ***forget all*** previous methods of detection, especially those which were based on searching for some typical signatures.

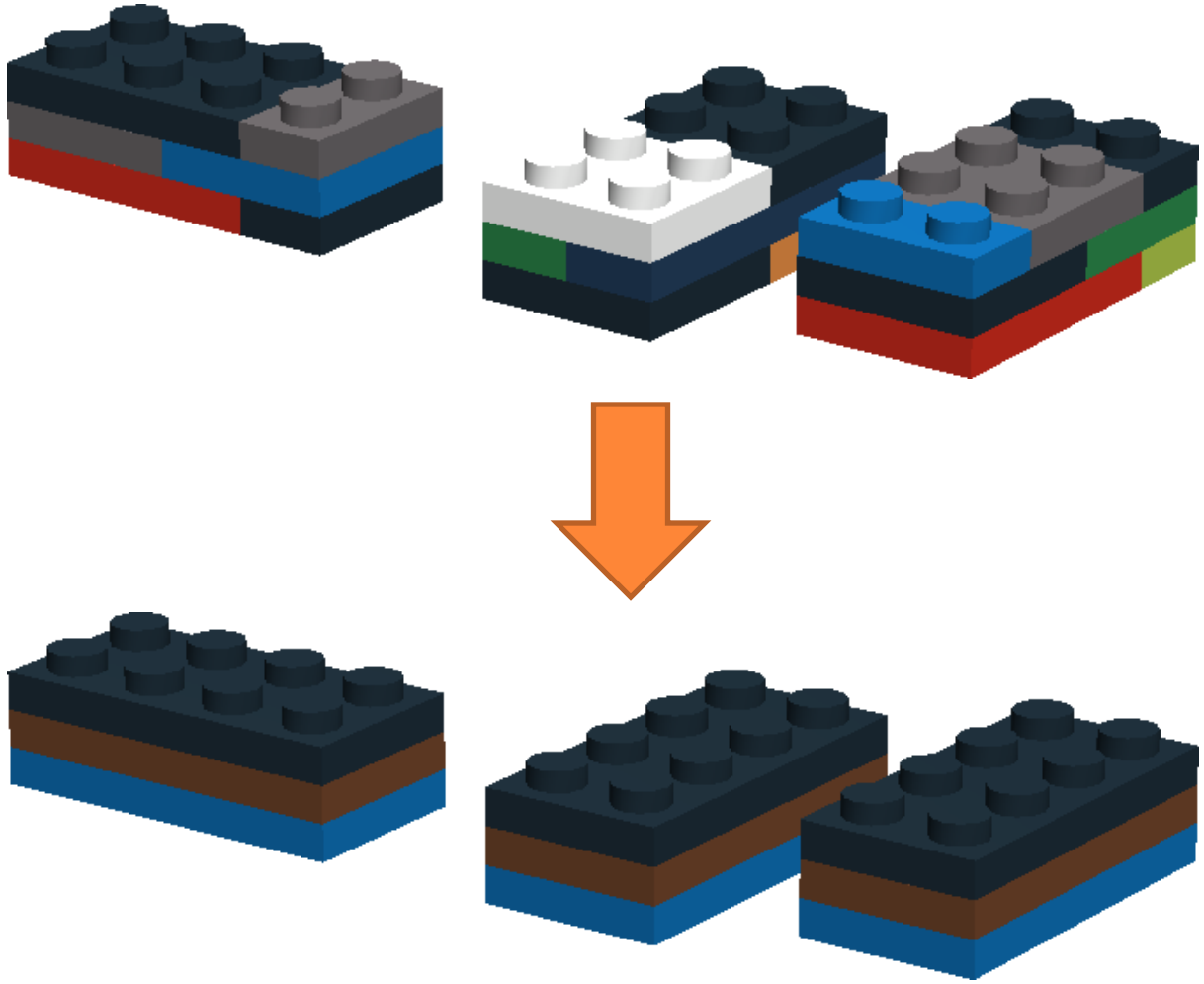

forget…


everything!

OK, then we need
a new type of definition!







- Each polymorphic virus or trojan
  has been created by some ***polymorphic generator***.

- Each polymorphic virus or trojan
  has been created by some ***polymorphic generator***.

- Each polymorphic family
  has its own ***specific*** generator.

- Each polymorphic virus or trojan
  has been created by some ***polymorphic generator***.

- Each polymorphic family
  has its own ***specific*** generator.

- Each polymorphic generator
  has some ***characteristics and limitations***.

We don't need to detect the **virus**

We don't need to detect the **virus**

We can detect the **generator** !

We can easily distinguish common ***compilers*** like Microsoft Visual C++ or Borland Delphi

We can easily distinguish common ***compilers*** like Microsoft Visual C++ or Borland Delphi

We can easily recognize various runtime ***compressors*** like UPX, PECompact or ASPack

We can easily distinguish common ***compilers*** like Microsoft Visual C++ or Borland Delphi

We can easily recognize various runtime ***compressors*** like UPX, PECompact or ASPack

We surely we will be able to detect any ***polymorphic generator***…

Which characteristics can be used?

Which characteristics can be used?

What about…

Which characteristics can be used?

What about…

bugs in the generator?

- Incorrect value in a file header
- Improper resource format
- Anything else…

- Incorrect value in a file header
- Improper resource format
- Anything else…

- Palette of instructions used in the produced code

- Incorrect value in a file header
- Improper resource format
- Anything else…

- Palette of instructions used in the produced code
- Jump flow of produced code

- Incorrect value in a file header
- Improper resource format
- Anything else…

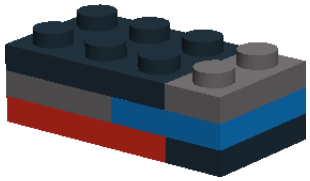
- Palette of instructions used in the produced code
- Jump flow of produced code
- Set of anti-debug and anti-disassembling tricks

- Incorrect value in a file header
- Improper resource format
- Anything else…

- Palette of instructions used in the produced code
- Jump flow of produced code
- Set of anti-debug and anti-disassembling tricks
- Amount of various illogical instructions

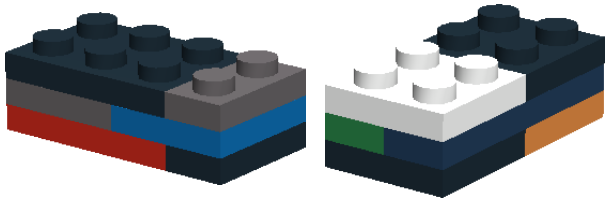
What else can we measure?
Any idea?

Virut (sample No.1)



```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....
00403220  542408E905460000B9E3ECE893F9F7D1  T$...F..........
00403230  FF957F040000C705E4114000FF151420  ..........@.....
00403240  C605E81140004031CB61C38D04378B72  ....@.@1.a...7.r
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $...........N..G
00403260  00000F8533480000C329E0FC8D8506BC  ....3H...)......
00403270  FFFF83E99301EE908DB7DF3455170FB7  ............4U...
00403280  9566BBFFFFEB67ADFDFB526AFFFF957F  .f....g...Rj....
00403290  040000E97D460000F7D0C20400FFD68D  ....}F..........
004032A0  48479080C15B50E9A40000008AE94295  HG...[P.......B.
004032B0  00D938C68B1D14204000F6D611E9B300  ..8.... @.......
004032C0  F583C8FFE91C4600002BC08D0D3D3959  ......F..+...=9Y
004032D0  498D5C241005E411400087431 0EBCD78  I.\$....@..C...x
004032E0  715A5250E8B4FFFFFFE9EF470000B103  qZRP.......G....
004032F0  C32400000090 83EFD78D4900E9B30000  .$........I.....
```

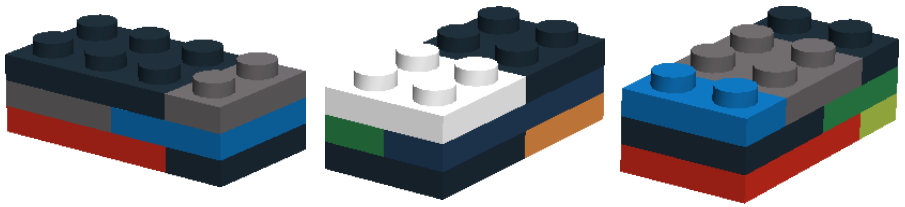
Virut (sample No.2)



```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....
00403220  542408E905460000B9E3ECE893F9F7D1  T$...F..........
00403230  FF957F040000C705E4114000FF151420  ..........@.....
00403240  C605E8114000403 1CB61C38D04378B72  ....@.@1.a...7.r
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $............N..G
00403260  00000F8533480000C329E0FC8D8506BC  ....3H...)......

01009800  2BED81EDE1E7FFFEE9F4480000F6D1F6  +.........H.....
01009810  D190BFD653FFFA86FA86FA8D381B5424  ....S.......8.T$
01009820  048D1283C40CF7F1F7C636741917F7D6  ..........6t....
01009830  BEE7992A5330C18D95F0FFFFFFE9F201  ...*S0..........
01009840  000086D09087F1FFD1F6D083EE4087D7  ..............@..
01009850  F942908BD268C8B1A9ED80C66181C699  .B...h......a...
01009860  AB9894FCE81746000 0E9B44600008D86  ......F....F....
01009870  345C32B48B72248AE78B7A1CE98B0100  4\2..r$...z.....
01009880  0087F790FC8D93E9F7A1F62CB20F8459  ............,...y
01009890  460000E93547000083F400FC5250E98F  F...5G......RP..
010098A0  0000000F854F48000090FC8AE4356CB5  .....OH......5l.
010098B0  9D28FC8D1BF7D4F7D49B8BFF8BFF902B  .(.............+
010098C0  44240485D3F0F856B470000EBA1478A9C  D$..?..kG....G..
010098D0  8D1B909052FC9B6AFFF89090FF956701  ....R..j......g.
010098E0  0000FF956B010000E97401 00008BD2FE  ....k....t......
010098F0  CCFEC490FC4941870424FC90FEC3FECB  .....IA..$......
```

Virut (sample No.3)



```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....
00403220  542408E905460000B9E3ECE893F9F7D1  T$...F..........
00403230  FF957F040000C705E4114000FF151420  ..........@.....
00403240  C605E8114000403 1CB61C38D04378B72  ....@.@1.a...7.r
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $...........N..G
00403260  00000F8533480000C329E0FC8D8506BC  ....3H...)......
```

```
01009800  2BED81EDE1E7FFFEE9F4480000F6D1F6  +.........H.....
01009810  D190BFD653FFFA86FA86FA8D381B5424  ....S.......8.T$
01009820  048D1283C40CF7F1F7C636741917F7D6  ..........6t....
01009830  BEE7992A5330C18D95F0FFFFFFE9F201  ...*S0..........
01009840  000086D09087F1FFD1F6D083EE4087D7  ..............@..
01009850  F942908BD268C8B1A9ED80C66181C699  .B...h.......a...
01009860  AB9894FCE81746000 0E9B446000008D86  ......F....F....
01009870  345C32B48B72248AE7E               
01009880  0087F790FC8D93E9F7F               
01009890  460000E93547000083F               
010098A0  0000000F854F480000 9               
010098B0  9D28FC8D1BF7D4F7D49               
010098C0  44240 48D3F0F856B476               
010098D0  8D1B909052FC9B6AFFF               
010098E0  0000FF956B010000E97               
010098F0  CCFEC490FC4941870 2               
```

```
00417000  8D3EBAD28A6C3ACD2E588AE881C7508E  .>...l:..X....P.
00417010  50D964FF30E902470000 F857E010000  P.d.0..G....W...
00417020  C3FF733CE90801 0000FFD685C00F8408  ..s<............
00417030  48000086D78AD48D149AFF95D2470000  H.............G..
00417040  E9D44700000F84D5FFFFFF28E500E5B8  ..G.........(....
00417050  00000000F980CC00F984F450E9FA4500  ............P..E.
00417060  0080020C51FC38DC9050EB039D5A1AFC  ....Q.8..P...Z..
00417070  524F47906AFF9EFF95D2470000FF95D6  ROG.j.....G.....
00417080  470000E92947000 0905B87D24183C8FF  G...)G...[..A...
00417090  3303E9DB450000598DB8850558 0986D0  3...E..Y....X...
004170A0  66C1E903E9C6470000FFD1F886C08D12  f.....G.........
004170B0  6896EC23FC08F4E865FFFFFFE9764500  h..#....e....vE.
004170C0  008D5424088044240204FE04248F02EB  ..T$..D$....$...
004170D0  3D23195DFEC08D34336AFFE92E460000  =#.]...43j...F..
004170E0  8D8C5824E3AFCAFEC50FB7957300 0000  ..X$........s...
004170F0  C3A40000008D4900E96601 00008BC08D  ......I..f......
```

You cannot catch me!

You cannot catch me!

Oh, we will see...

What about to use some **mathematics** ?

What about to use some **mathematics** ?

```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....
00403220  542408E90546000B9E3ECE893F9F7D1  T$...F..........
00403230  FF957F040000C705E4114000FF151420  ..........@.....
00403240  C605E8114000403 1CB61C38D04378B72  ....@.@1.a...7.r
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $...........N..G
00403260  00000F85334800000C329E0FC8D8506BC  ....3H...)......
00403270  FFFF83E99301EE908DB7DF3455170FB7  ...........4U...
00403280  9566BBFFFFEB67ADFDFB526AFFFF957F  .f....g...Rj....
00403290  040000E97D460000F7D0C20400FFD68D  ....}F..........
004032A0  48479080C15B50E9A40000008AE94295  HG...[P.......B.
004032B0  00D938C68B1D14204000F6D611E9B300  ..8.... @.......
004032C0  F583C8FFE91C4600002BC08D0D3D3959  ......F..+...=9Y
004032D0  498D5C241005E411400087431 0EBCD78  I.\$....@..C...x
004032E0  715A5250E8B4FFFFFFE9EF470000B103  qZRP.......G....
004032F0  C32400000009083EFD78D4900E9B30000  .$........I.....
```

We will add a new column, an **entropy** level

```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B  -
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....  E
00403220  542408E90546000B9E3ECE893F9F7D1  T$...F..........  -
00403230  FF957F040000C705E4114000FF151420  ..........@....   1
00403240  C605E8114000403 1CB61C38D04378B72  ....@.@1.a...7.r  E
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $............N..G  D
00403260  00000F85334800000C329E0FC8D8506BC  ....3H...)......  B
00403270  FFFF83E99301EE908DB7DF3455170FB7  ............4U...  E
00403280  9566BBFFFFEB67ADFDFB526AFFFF957F  .f....g...Rj....  E
00403290  040000E97D460000F7D0C20400FFD68D  ....}F..........  A
004032A0  48479080C15B50E9A40000008AE94295  HG...[P.......B.  C
004032B0  00D938C68B1D142040 00F6D611E9B300  ..8.... @.......  C
004032C0  F583C8FFE91C4600002BC08D0D3D3959  ......F..+...=9Y  C
004032D0  498D5C241005E4114000874310EBCD78  I.\$....@..C...x  C
004032E0  715A5250E8B4FFFFFFE9EF470000B103  qZRP.......G....  D
004032F0  C32400000009083EFD78D4900E9B30000  .$........I.....  9
```

The entropy describes local **density** of data

```
00403200  F6D138F280F2E2F6E1F7D6F7D7008542  ..8............B  -
00403210  BBFFFFEB549F21BF86ED33C083C1908D  ....T.!...3.....  E
00403220  542408E90546000B9E3ECE893F9F7D1  T$...F..........  -
00403230  FF957F040000C705E4114000FF151420  ..........@....   1
00403240  C605E8114000403 1CB61C38D04378B72  ....@.@1.a...7.r  E
00403250  2487D2F7D28AD103F30FB7044EE9FC47  $............N..G  D
00403260  00000F85334800000C329E0FC8D8506BC  ....3H...)......  B
00403270  FFFF83E99301EE908DB7DF3455170FB7  ............4U...  E
00403280  9566BBFFFFEB67ADFDFB526AFFFF957F  .f....g...Rj....  E
00403290  040000E97D460000F7D0C20400FFD68D  ....}F..........  A
004032A0  48479080C15B50E9A40000008AE94295  HG...[P.......B.  C
004032B0  00D938C68B1D14204000F6D611E9B300  ..8.... @.......  C
004032C0  F583C8FFE91C4600002BC08D0D3D3959  ......F..+...=9Y  C
004032D0  498D5C241005E4114000874310EBCD78  I.\$....@..C...x  C
004032E0  715A5250E8B4FFFFFFE9EF470000B103  qZRP.......G....  D
004032F0  C32400000009083EFD78D4900E9B30000  .$........I.....  9
```

Forgot the real data, only the entropy is needed

```
00403000                                        -E-1EDBEEACCCCD9AAAAADCDEDECCDCC
00403400  DCCCCCCCCCCCCCCBAAABBCBCBCCCCCCCBCBCBCBCCCCCCCCCCCDDDDDECDDDB85
00403800  5BBBBBCBCCC3D8-------------------------------------------------
00403C00  ------------------------------------------------D-CCCCDCD7-
00404000  ----------41111112262-------------------D7DEEDCCCCCB----DDEEEEE
00404400  EEEEEEEEEEEDDDDDDEDDDCCCCECDEDCCCBCBCCBBCBBBBCCBECBDDCCCCABABAB
00404800  ACAADD9DDDCCECCCCCDCCDCCCCCCDECCDCCDCCCCCECCDCDDCDCDDCBBCBBBDEDE
00404C00  DDDDDDDEEDDDDDDDDDDEEDDDCDDCDDCCCC---BBCBCCCCBBBBBCDDDCDCDCCBCBD
00405000  BCBCBCCCCCCCCCCDCDDCCCCCAAACAAACBCBBCCBAAACDBCCCCCCCBBBBBDDDDDDD
00405400  DDDDDDDDCDCCCDCCCDDDDDBBBBBCCCCDDCCDCCDDDD9999999DDCDCC8888DEDEE
00405800  EEDDDDEDEEEDDDDDDDEDEDDDCEDCDCEDCDDDDEDDEDDDECCCEEEEEDDEEEEEDBBB
00405C00  BBBDDDECCCEDDDDDDDEDDEDEDEEDDEEEEECEECDDCDDCDDDDDDDDDDCDCCCCBBAA
00406000  AACAACBCBCDBBBCCDCCCDEDDDAEAC9999BABABBBCDCDDDDDBBBDCDEEDCCCCDDD
00406400  EDCCCCCAAABADACAAAACADDDDDDDEDDDEDEDDDDDDDDDDDDDDBBBDCCCCCCCCCCC
00406800  CCDDCDDCECCDCDCDDDDDDDCCDDDEEDEEDDDDEDCCDDDCBCCAAAAABCBACC9999B9
00406C00  ------------------------------------------------------------------
00407000  ------------------------------------------------------------------
00407400  ------------AABAADAAAAAABCDDDDECDDDDCDCBCDAABABAACACCDDDCEAEECCE
00407800  EEEEDDEEEDDCDDDECAEDDEABDDBDEBDDBDDBDCECECECCDECBBBB------------
```

Virut (sample No.1)

Compare to the sample No.2

```
01009800  -DEDDEBDDAADDD9DDDCDDEDEEDEADEDD9EDDEDDDCDCDDECCCCDCCCCCCCCCCCCC
01009C00  CBAAABBCBCCBCCCCCCBCBCBCBCCCCCDCCECCDDDDECCDDA855BBCBBCBCDCD77
0100A000  6-----------------------------------------------------------------
0100A400  ------------------------------------D-C--CCC87E------------411
0100A800  11112252-------------------D5DEEDCCCCCB----DDEEEEEEEEEEEEEEEEDDD
0100AC00  DDDDEDDDCCCCECDEDCCCBCBCCBBCBBBBCCBECBDDCCCCABABABACAADD9DDDCCEC
0100B000  CCCCDCCDCCCCCCDDCCDCCDCCCCCECCDCDDCDCDDCBBCBBBDEDEDDDDDDDEEDDDDD
0100B400  DDDDDEE4DDCD-5422CCC---BBCBCCCCBBBBBCDDDCDCDCCBCBDBCBCBCCCCCCCCC
0100B800  CDCDDCCCCCAAACAAACBCBBCCBAAACDBCCCCCCCBBBBBDDDDDDDDDDDDDDDCDCCCD
0100BC00  CCCDDDDDBBBBBCCCCDD5CDCCDDDD9999999DDCDCC8888DEDEEEEDDDDEDEEEDDD
0100C000  EDDDEDEDDDCEDCDCEDCDDDDEDDEDDDECCCEEEDEDDEEEEEDBBBBBDDDECCCEDDD
0100C400  DDDDEDDDDDDEDDDEEEEECEECDDCDDCDDDDDDDCDDCDCCCCBBAAAACAACBCBCDBBB
0100C800  CCDCCCDEDDDDEBC9999BBBBBBBCDCDDDDDBBBDCDEEDCCCCDDDEDCCDCCAAABADA
0100CC00  CAAAACADDDDDDDEDDDEDEDDDDDDDDDDDDDDBBBDCCCCCCCCCCCCCCDDCDDCECCDCD
0100D000  CDDDDDDDCCDDDEEDDEDDDDEDCCDDBCCBCCBCBCDBCABD9999B9--------------
0100D400  -----------------------------------------------------------------
0100D800  ----------------------------------------------------------------AB
0100DC00  AAACAAAAABACDDDDDCDDDDCDCBCDAABABAACACBBCEECCCCCCCCDDEEEDCDDADDE
0100E000  DCCDCCBDBB6B6D3----------------
```

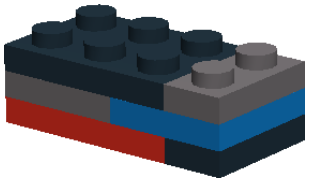
Virut (sample No.2)

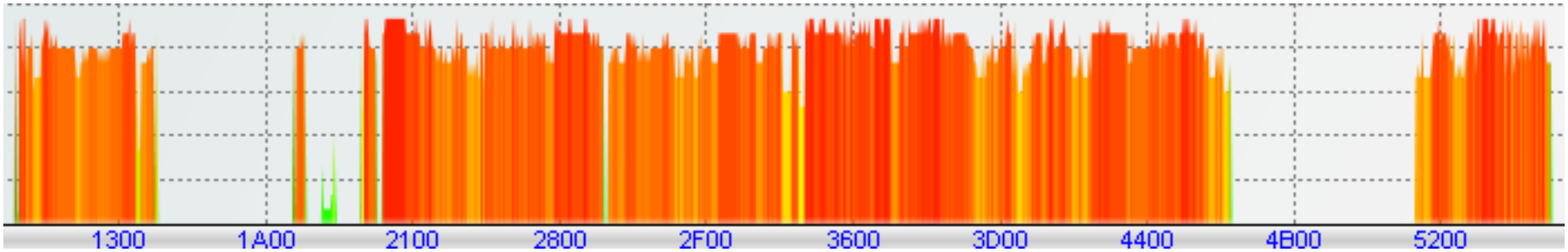
And also to the sample No.3

```
00417000  E--------DDDDDC9DEADEDCCCC9DEDCECEEBBDDCDCCDCCCCCCCCCCCDCCBAAACB
00417400  CBCBCCCCCCCBCBCBCBCBCCCCCCDCCCDDDDDECDDDB755BBBBBCBCCCAD82------
00417800  ----------------------------------------------------------------
00417C00  ------------------------------------A9D6-------------4111111225
00418000  2-----------------DD5DEEDCCDCEB----DDEEEEEEEEEEEEEEEEDDDDDEDEDD
00418400  DCCCCCEDDCCCBCCBCCBCBBCBBCCBCCBDCCCCDABABAABABAD99DDDCCCCCCCCCDC
00418800  CCCCCCEDCCDCCDCCCCCCECCCDDDCDCDCDBBBDBBDEDEDDDDDDDEDDDDDDDDDEEED
00418C00  DDCDDCDDCDCC---EBBCBDCCCBBBBBCDDCDDCDCCBCBDBCBCBCCCCCCCCCCDCECDC
00419000  CCACAAABAACBCBBCCBAABCCBCCCCCCBBCBBBDDEDDDDDDDDDDDDCECCCCCCCDDDD
00419400  DBBBBBCCCCDCECDCCDDDD9999999DCDDCC8888DEDEEEEDDDDEDEEEDDDDEDDEDD
00419800  DEDCECDDCDDCCCCDEDDDDDECCCCEEEEDDDEEEEEDBBBBBDDDCCCEDDDDEDDDDDD
00419C00  DDDDEDDDEEEEECEECDDCDDCDDDEDDEDDCCDCCCBBABAAAAACCBCBCDBBBCCCECCD
0041A000  EDDDEEBC9999BBCBBBBCCDDDDDDBBCCDDEDDCCCCDDDDDCCDCCAAAAACAAAAAAAC
0041A400  DDDDDDDDDDDEDEDDDDDDDDDDDDDDBBCCCCCCCCCCCCCCCCDDCDDCDCCCDDCDDDDDD
0041A800  DCCDDDEDEEEEDDDDDCCDDDBCCBCBCCBCDBCAC99999B-9--------------------
0041AC00  ----------------------------------------------------------------
0041B000  ------------------------------------------------------AABAADAAA
0041B400  AAAECDDDDECDDD6CDCBCDAABAABABABDBCCCCAEACECECDCDDDEDCDCCDCEEECCD
0041B800  DDEDEDDED--------------------------------------------------------
```

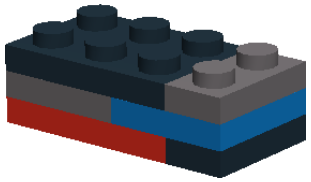
Virut (sample No.3)

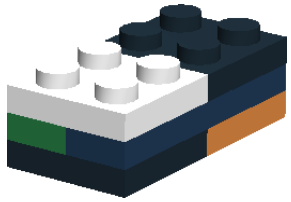
…Virut in charts

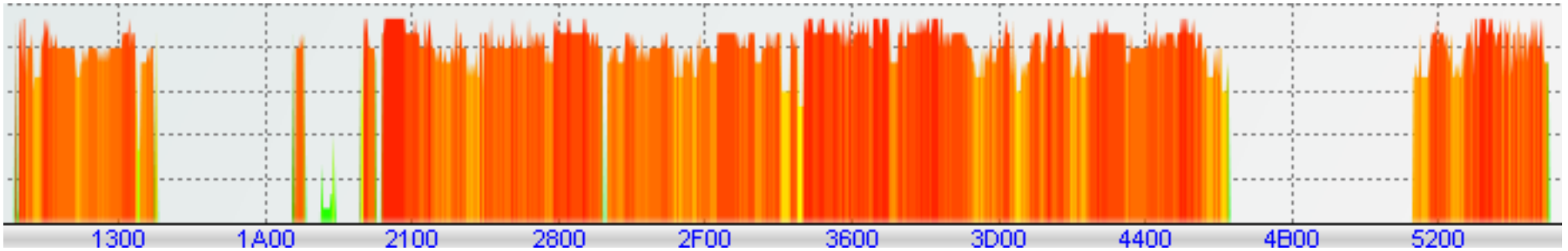


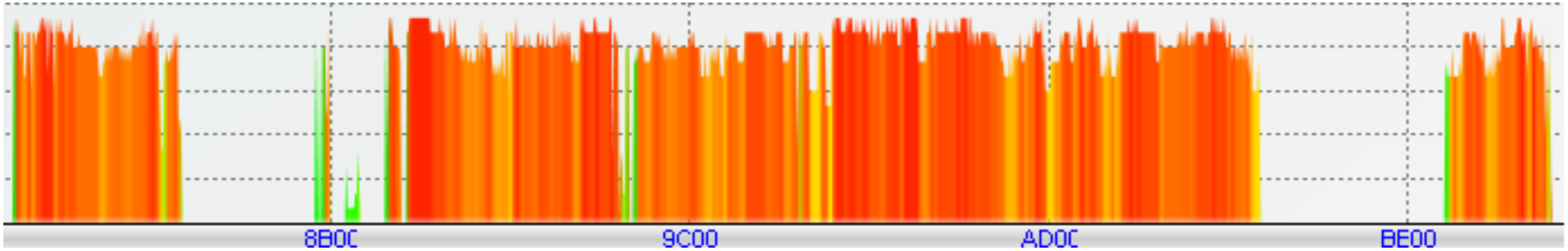


…Virut in charts

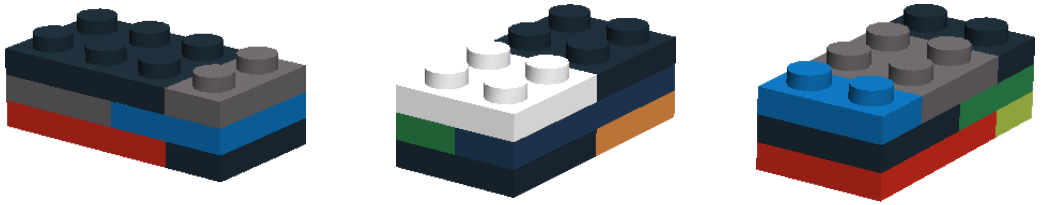


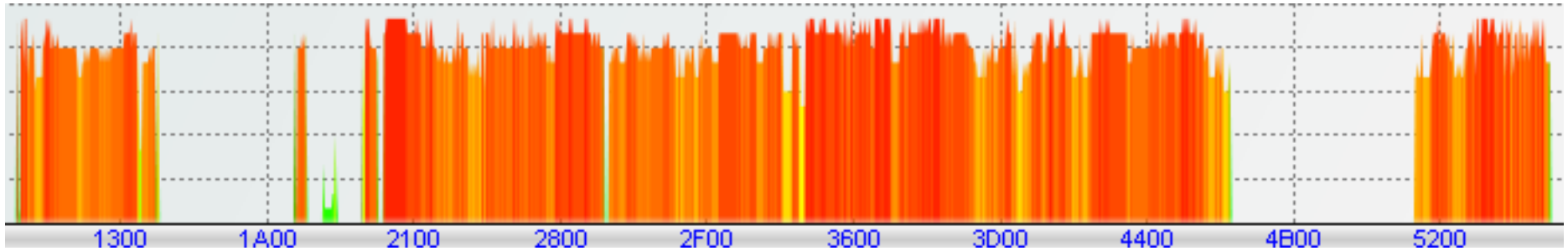


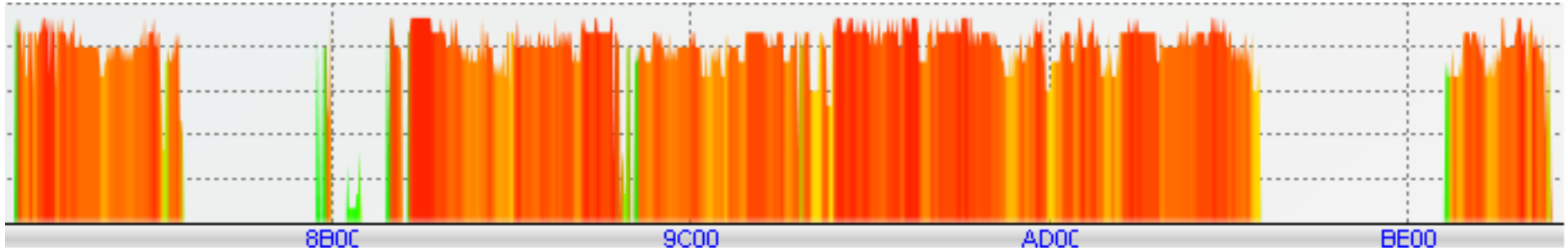


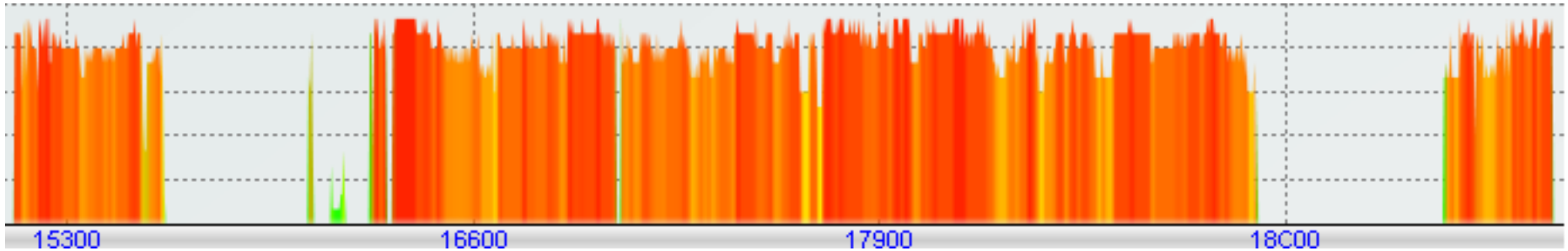


…Virut in charts







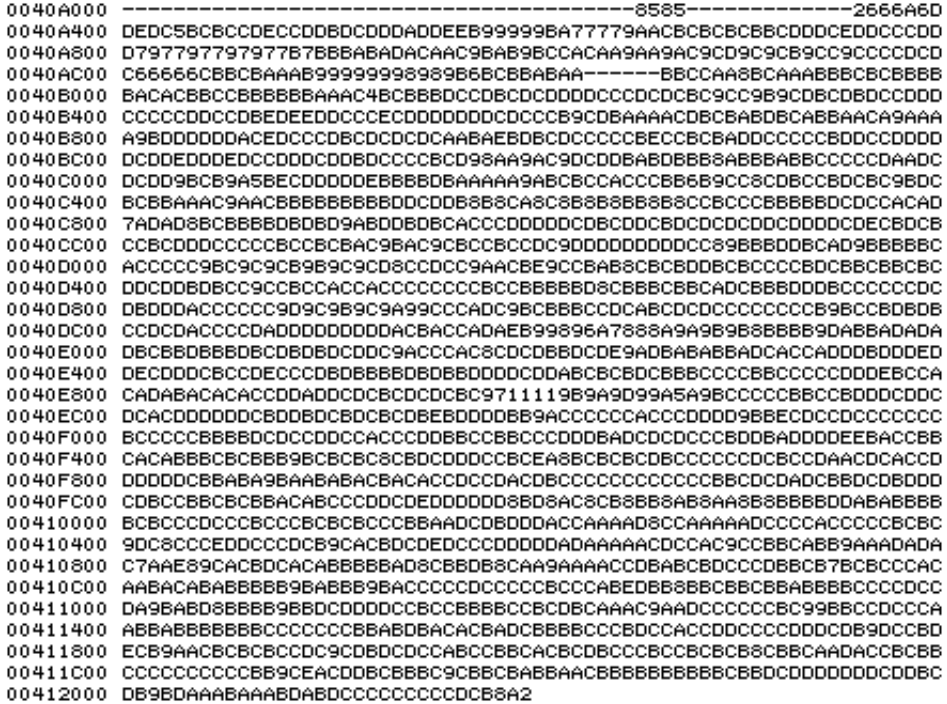


We only have to compute an **average picture** and set the allowed variance

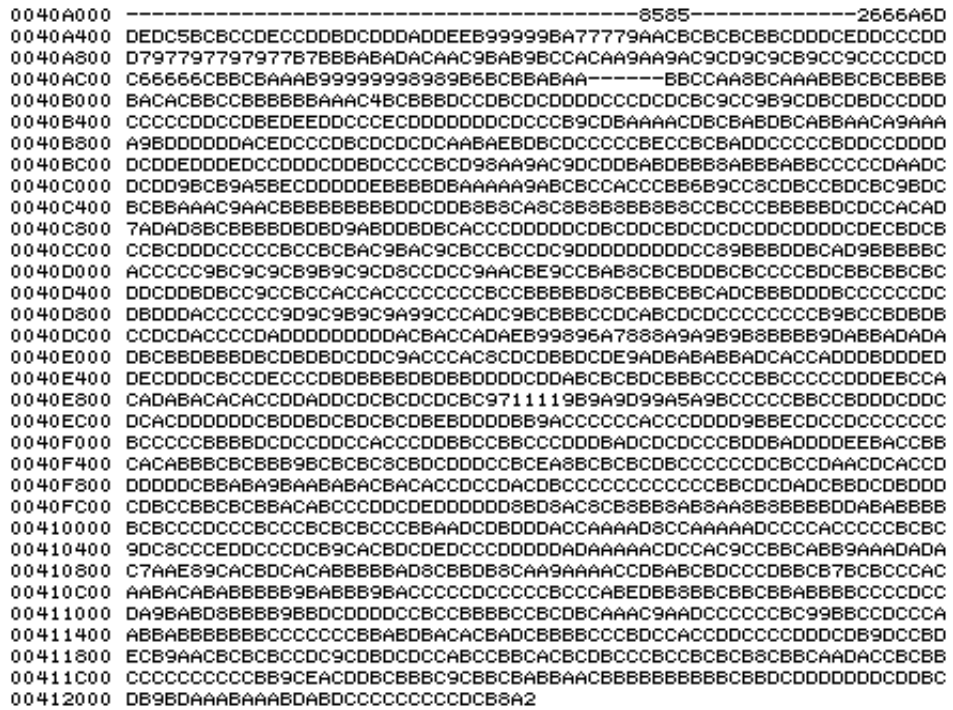
We only have to compute an **average picture** and set the allowed variance

The detection is **easy and fast**

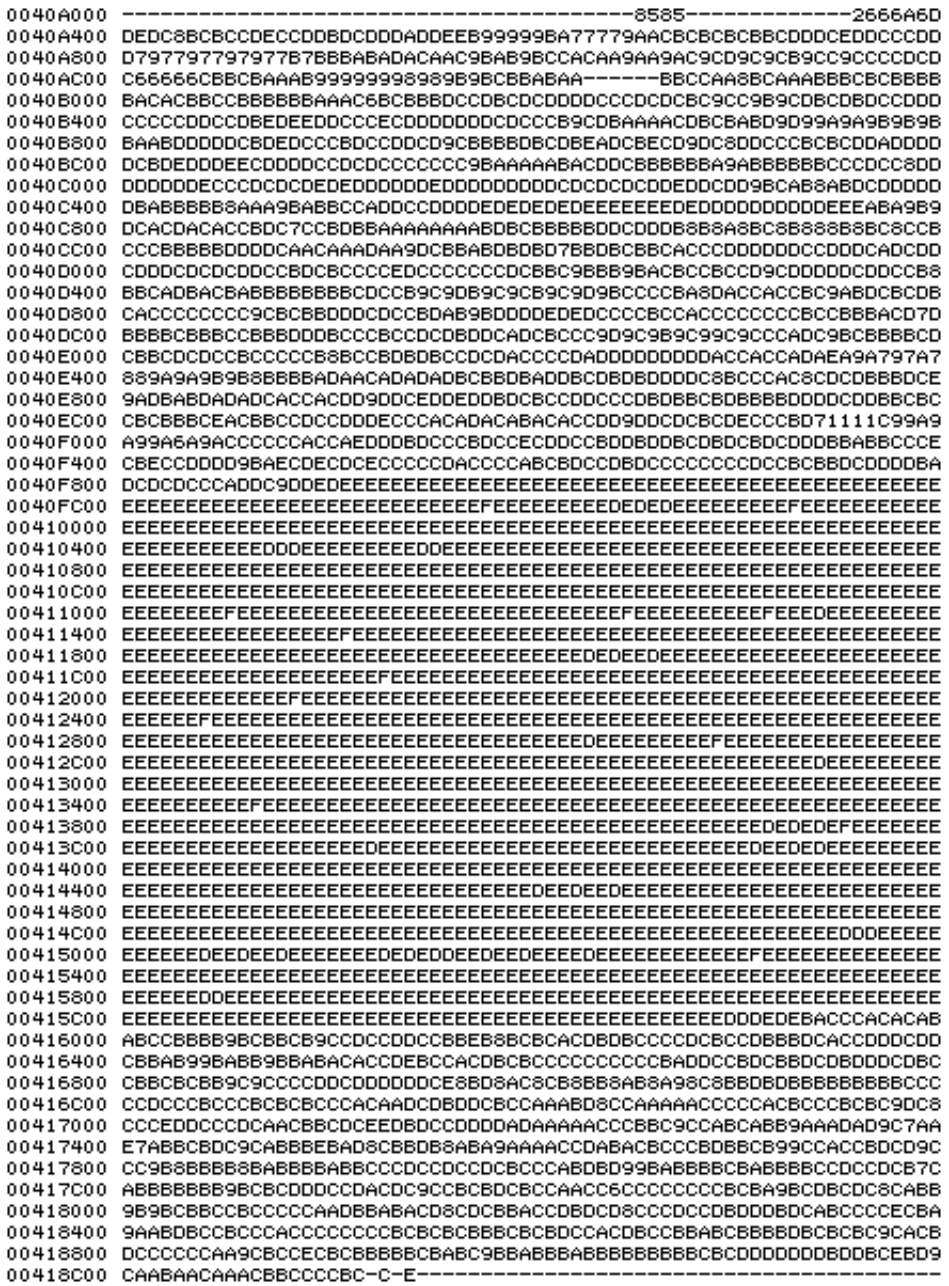
The magic is, to find
a completely **new look at data**
(whatever it may be)



REGEDIT.EXE (clean)



REGEDIT.EXE (clean)

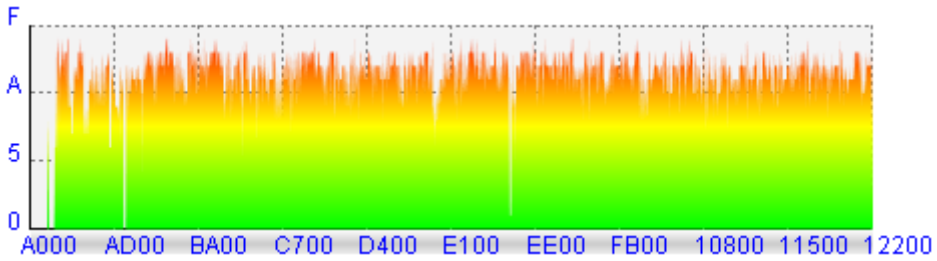


REGEDIT.EXE

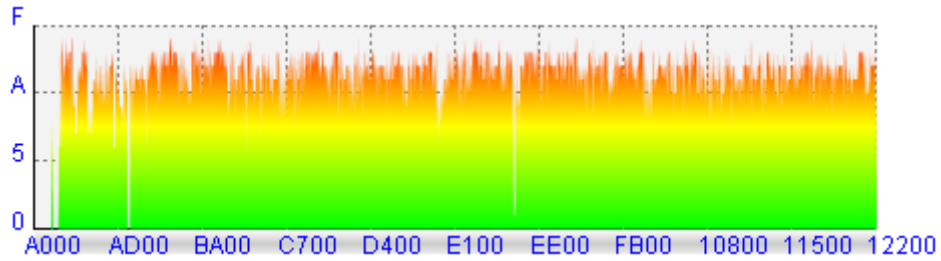
REGEDIT.EXE
(infected by ZMist)

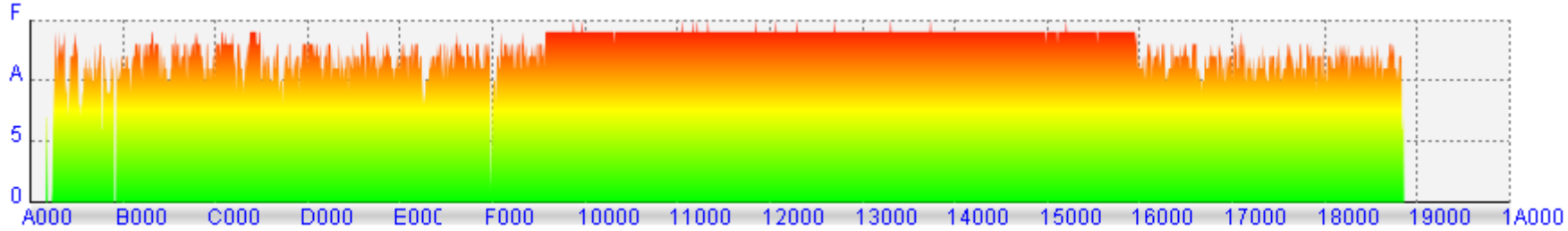
```
0040A000 ----------------------------------------8585-------------2666A6D
0040A400 DEDC8BCBCCDECCDDBDCDDDADDEEB99999BA77779AACBCBCBCBBCDDDCEDDCCCDD
0040A800 D7977977979977B7BBBABADACAAC9BAB9BCCACAA9AA9AC9CD9C9CB9CC9CCCCDCD
0040AC00 C66666CBBCBAAAB999999989B9B9CBBABAA------BBCCAA8BCAAABBBCBCBBBB
0040B000 BACACBBCCBBBBBBAAAC6BCBBBDCCDBCDCDDDDCCCDCDCBC9CC9B9CDBCDBDCCDDD
0040B400 CCCCCDDCCDBEDEEDDCCCECDDDDDDDCDCCCB9CDBAAAACDBCBABD9D99A9A9B9B9B
0040B800 BAABDDDDDCBDEDCCCBDCCDDCD9CBBBBDBCDBEADCBECD9DC8DDCCCBCBCDDADDDD
0040BC00 DCBDEDDDEECDDDDCCDCDCCCCCCC9BAAAAABACDDCBBBBBBA9ABBBBBBCCCDCC8DD
0040C000 DDDDDDECCCDCDCDEDEDDDDDDEDDDDDDDDDCDCDCDCDDEDDCDD9BCAB8ABDCDDDDD
0040C400 DBABBBBB8AAA9BABBCCADDCCDDDDEDEDEDEEEEEEEDEDDDDDDDDDEEEABA9B9
0040C800 DCACDACACCBDC7CCBDBBAAAAAAAABDBCBBBBBDDCDDDB8B8A8BC8B888B8BC8CCB
0040CC00 CCCBBBBBDDDDCAACAAADAA9DCBBABDBDBD7BBDBCBBCACCCDDDDDDCCDDDCADCDD
0040D000 CDDDCDCDCDDCCBDCBCCCCEDCCCCCCCCDCBBC9BBB9BACBCCBCCD9CDDDDDCDDCCB8
0040D400 BBCADBACBABBBBBBBBCDCCB9C9DB9C9CB9C9D9BCCCCBA8DACCACCBC9ABDCBCDB
0040D800 CACCCCCCCC9CBCBBDDDCDCCBDAB9BDDDDEDEDCCCCBCCACCCCCCCCBCCBBBACD7D
0040DC00 BBBBCBBBCCBBBDDDBCCCBCCDCDBDDCADCBCCC9D9C9B9C99C9CCCADC9BCBBBBCD
0040E000 CBBCDCDCCBCCCCCB8BCCBDBDBCCDCDACCCCDADDDDDDDDDACCACCADAEA9A797A7
0040E400 889A9A9B9B8BBBBADAACADADADBCBBDBADDBCDBDBDDDDC8BCCCAC8CDCDBBBDCE
0040E800 9ADBABDADADCACCACDD9DDCEDDEDDBDCBCCDDCCCDBDBBCBDBBBBDDDDCDDBBCBC
0040EC00 CBCBBBCEACBBCCDCCDDDECCCACADACABACACCDD9DDCDCBCDECCCBD71111C99A9
0040F000 A99A6A9ACCCCCCACCAEDDDBDCCCBDCCECDDCCBDDBDDBCDBDCBDCDDDBBABBCCCE
0040F400 CBECCDDDD9BAECDECDCECCCCCDACCCCABCBDCCDBDCCCCCCCCDCCBCBBDCDDDDBA
0040F800 DCDCDCCCADDC9DDEDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
0040FC00 EEEEEEEEEEEEEEEEEEEEEEEEEEEEFEEEEEEEEEDEDEDEEEEEEEEEFEEEEEEEEEE
00410000 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00410400 EEEEEEEEEEEDDDEEEEEEEEEDDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00410800 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00410C00 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00411000 EEEEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEFEEEEEEEEEEFEEEDEEEEEEEE
00411400 EEEEEEEEEEEEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00411800 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEDEDEEEEEEEEEEEEEEEEEEEEEEEE
00411C00 EEEEEEEEEEEEEEEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00412000 EEEEEEEEEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00412400 EEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00412800 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00412C00 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00413000 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00413400 EEEEEEEEEEFEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00413800 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00413C00 EEEEEEEEEEEEEEEEEEEDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00414000 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00414400 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00414800 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00414C00 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00415000 EEEEEEDEEDEEDEEEEEEEDEDEDDEEDEEDEEEEDEEEEEEEEEEEEEEEEEEEEEEEE
00415400 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00415800 EEEEEEDDEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00415C00 EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
00416000 ABCCBBBB9BCBBCB9CCDCCDDCCBBEB8BCBCACDBDBCCCCDCBCCDBBBDCACCDDDCDD
00416400 CBBAB99BABB9BBABACACCDEBCCACDBCBCCCCCCCCCCBADDCCBDCBBDCDBDDDCDBC
00416800 CBBCBCBB9C9CCCCDDCDDDDDDCE8BD8AC8CB8BB8AB8A98C8BBDBDBBBBBBBBBCCC
00416C00 CCDCCCBCCCBCBCBCCCACAADCDBDDCBCCAAABD8CCAAAAACCCCCACBCCCBCBC9DC8
00417000 CCCEDDCCCDCAACBBCDCEEDBDCCDDDDADAAAAACCCBBC9CCABCABB9AAADAD9C7AA
00417400 E7ABBCBDC9CABBBEBAD8CBBDB8ABA9AAAACCDABACBCCCBDBBCB99CCACCBDCD9C
00417800 CC9B8BBBB8BABBBBABBCCCDCCDCCDCBCCCABDBD99BABBBBCBABBBBCCDCCDCB7C
00417C00 ABBBBBBB9BCBCDDDCCDACDC9CCBCBDCBCCAACC6CCCCCCCCCBCBA9BCDBCDC8CABB
00418000 9B9BCBBCCBCCCCCAADBBABACD8CDCBBACCDBDCD8CCCDCCDBDDDBDCABCCCCECBA
00418400 9AABDBCCBCCCACCCCCCCCCBCBCBCBBBCBCBDCCACDBCCBBABCBBBBDBCBCBC9CACB
00418800 DCCCCCCAA9CBCCECBCBBBBBCBABC9BBABBBABBBBBBBBBCBCDDDDDDDBDDBCEBD9
00418C00 CAABAACAAACBBCCCCBC-C-E----------------------------------------
```

REGEDIT.EXE

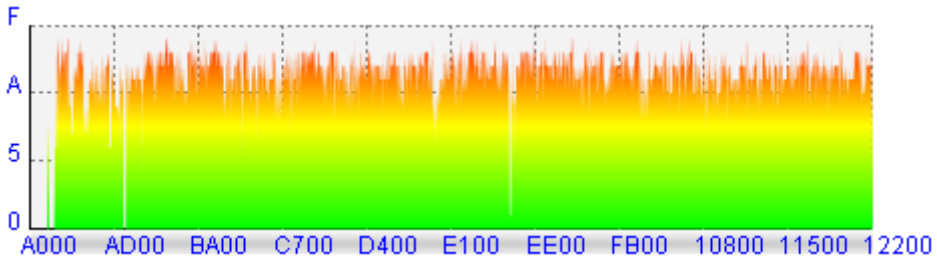

F
A
5
0
A000
AD00
BA00
C700
D400
E100
EE00
FB00
10800
11500
12200

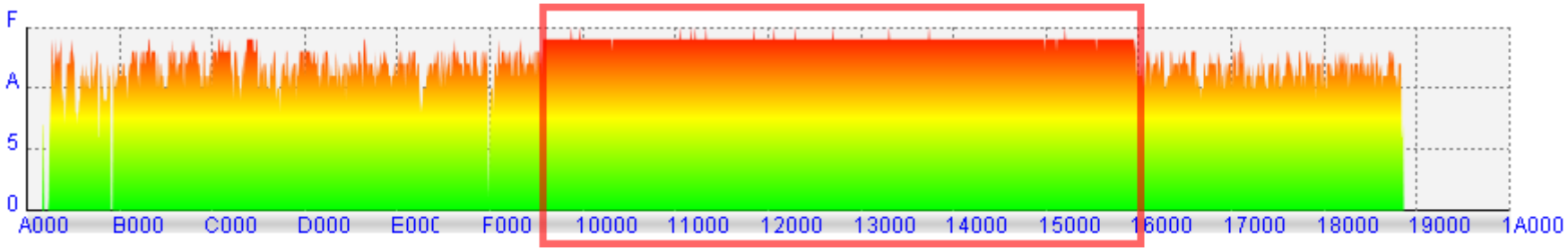
REGEDIT.EXE


F
A
5
0
A000
AD00
BA00
C700
D400
E100
EE00
FB00
10800
11500
12200


F
A
5
0
A000
B000
C000
D000
E00
F000
10000
11000
12000
13000
14000
15000
16000
17000
18000
19000
1A000

REGEDIT.EXE


F
A
5
0
A000 AD00 BA00 C700 D400 E100 EE00 FB00 10800 11500 12200


F
A
5
0
A000 B000 C000 D000 E00(F000 10000 11000 12000 13000 14000 15000 16000 17000 18000 19000 1A000

Using entropy maps, all of you can be malware experts!

OK, so how can we compute the entropy map?

1. Each sequence of 16 subsequent bytes is evaluated by local entropy value. A function which computes it, also examines larger surrounding than those 16 bytes only.

2. The entropy value is computed for each particular byte. The function starts with the value 15. Then the function is stepping in interval 1 and check adjacent bytes, whether they are the same as this reference byte. If yes, the entropy is decreased by one. On the first non-matching byte the function stops.

3. Then the function performs the same comparison, but in interval of 2 (i.e. it examines bytes in offset 2, 4, 6, 8 etc.), then in interval of 3 (offset 3, 6, 9, 12...), interval of 4 up to interval of 32. The lowest entropy, which has been found, is taken as the proper entropy value for this examined byte.

4. As the last step, we take each 16 subsequent bytes and assign the lowest value of their entropy as the entropy of this group.

1. Each sequence of 16 subsequent bytes is evaluated by local entropy value. A function which computes it, also examines larger surrounding than those 16 bytes only.

2. The entropy value is computed for each particular byte. The function starts with the value 15. Then the function is stepping in interval 1 and check adjacent bytes, whether they are the same as this reference byte. If yes, the entropy is decreased by one. On the first non-matching byte the function stops.

3. Then the function performs the same comparison, but in interval of 2 (i.e. it examines bytes in offset 2, 4, 6, 8 etc.), then in interval of 3 (offset 3, 6, 9, 12...), interval of 4 up to interval of 32. The lowest entropy, which has been found, is taken as the proper entropy value for this examined byte.

4. As the last step, we take each 16 subsequent bytes and assign the lowest value of their entropy as the entropy of this group.

1. Each sequence of 16 subsequent bytes is evaluated by local entropy value. A function which computes it, also examines larger surrounding than those 16 bytes only.

2. The entropy value is computed for each particular byte. The function starts with the value 15. Then the function is stepping in interval 1 and check adjacent bytes, whether they are the same as this reference byte. If yes, the entropy is decreased by one. On the first non-matching byte the function stops.

3. Then the function performs the same comparison, but in interval of 2 (i.e. it examines bytes in offset 2, 4, 6, 8 etc.), then in interval of 3 (offset 3, 6, 9, 12...), interval of 4 up to interval of 32. The lowest entropy, which has been found, is taken as the proper entropy value for this examined byte.

4. As the last step, we take each 16 subsequent bytes and assign the lowest value of their entropy as the entropy of this group.

1. Each sequence of 16 subsequent bytes is evaluated by local entropy value. A function which computes it, also examines larger surrounding than those 16 bytes only.

2. The entropy value is computed for each particular byte. The function starts with the value 15. Then the function is stepping in interval 1 and check adjacent bytes, whether they are the same as this reference byte. If yes, the entropy is decreased by one. On the first non-matching byte the function stops.

3. Then the function performs the same comparison, but in interval of 2 (i.e. it examines bytes in offset 2, 4, 6, 8 etc.), then in interval of 3 (offset 3, 6, 9, 12...), interval of 4 up to interval of 32. The lowest entropy, which has been found, is taken as the proper entropy value for this examined byte.

4. As the last step, we take each 16 subsequent bytes and assign the lowest value of their entropy as the entropy of this group.

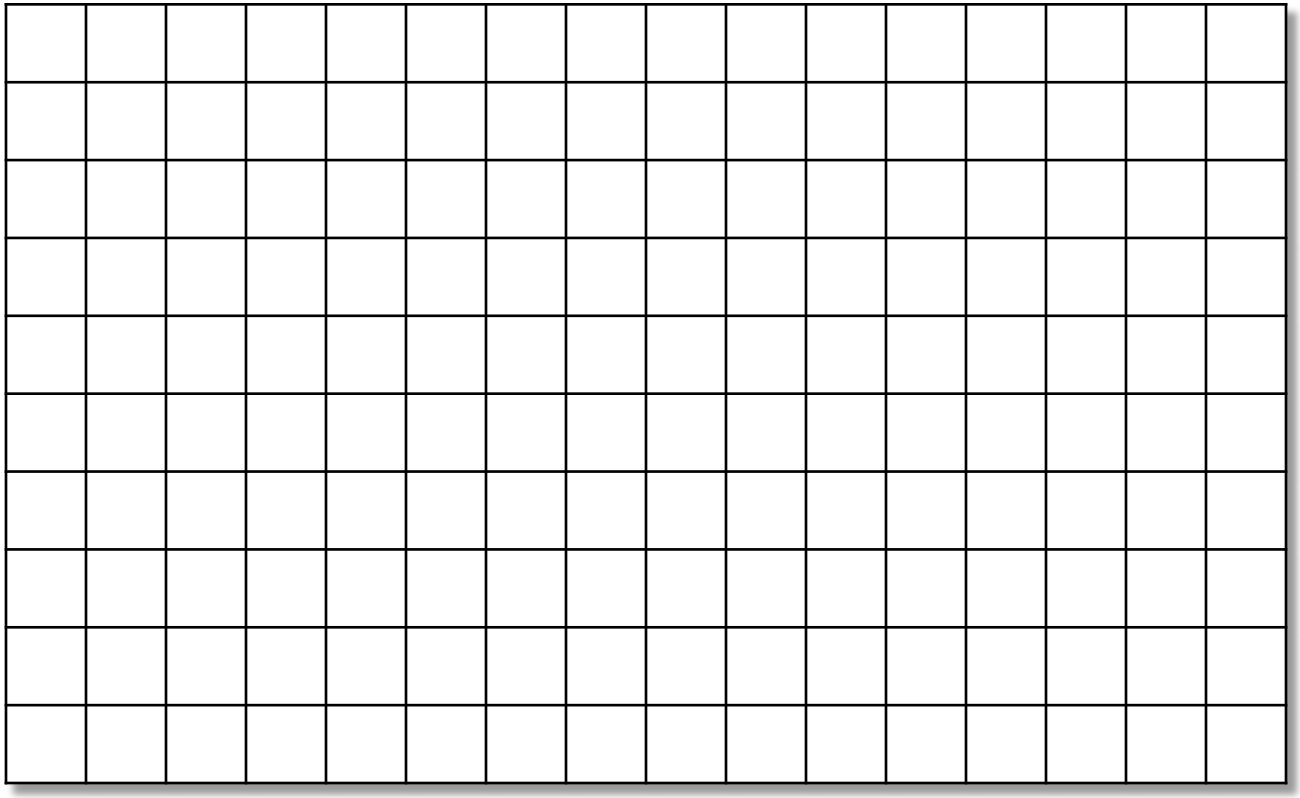
Do you understand?
Not yet?

Do you understand?
Not yet?

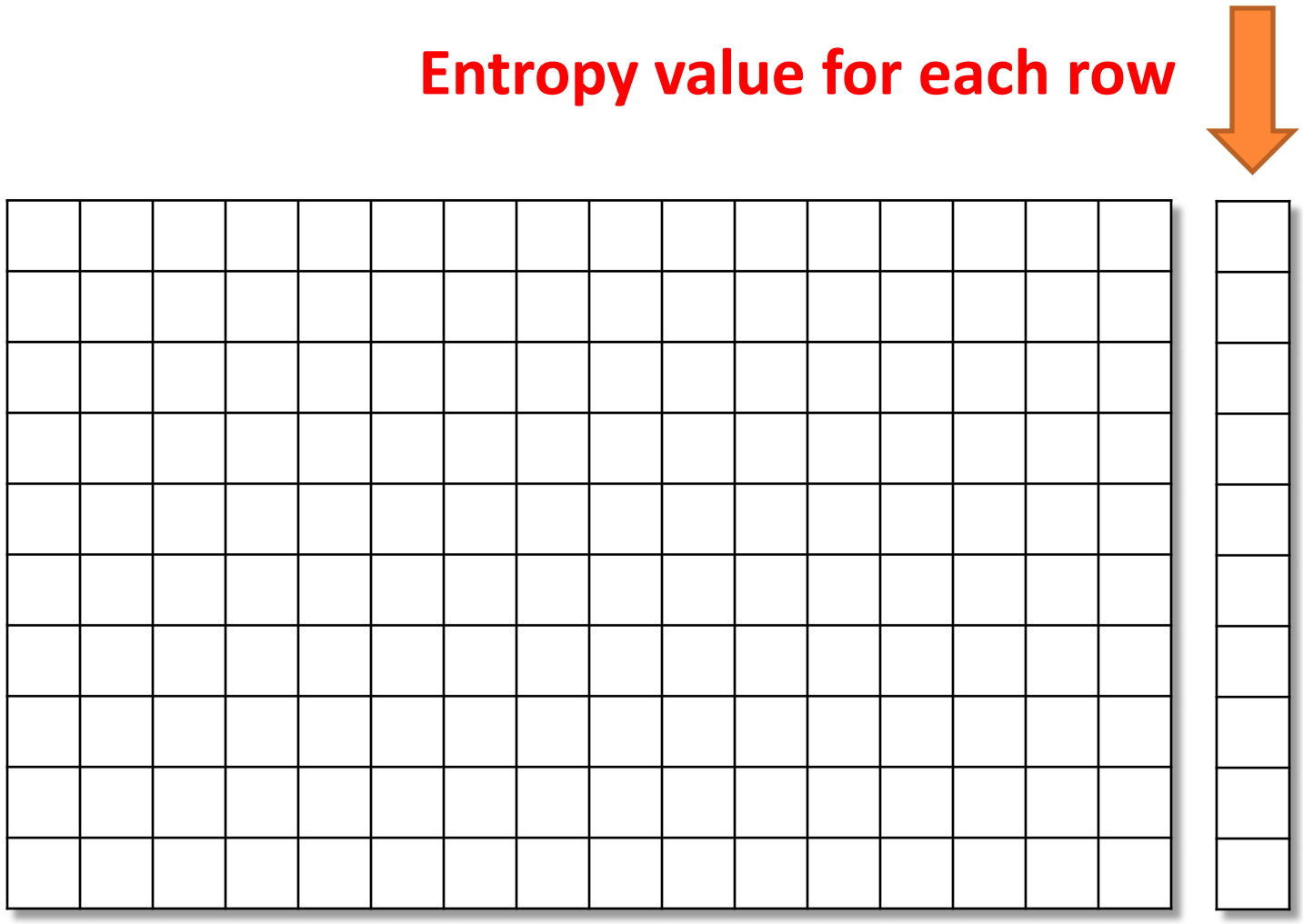
So let us to explain the whole algorithm **in details** !

Examined data (buffer of 160 Bytes)

HEX dump
16 x 10 bytes



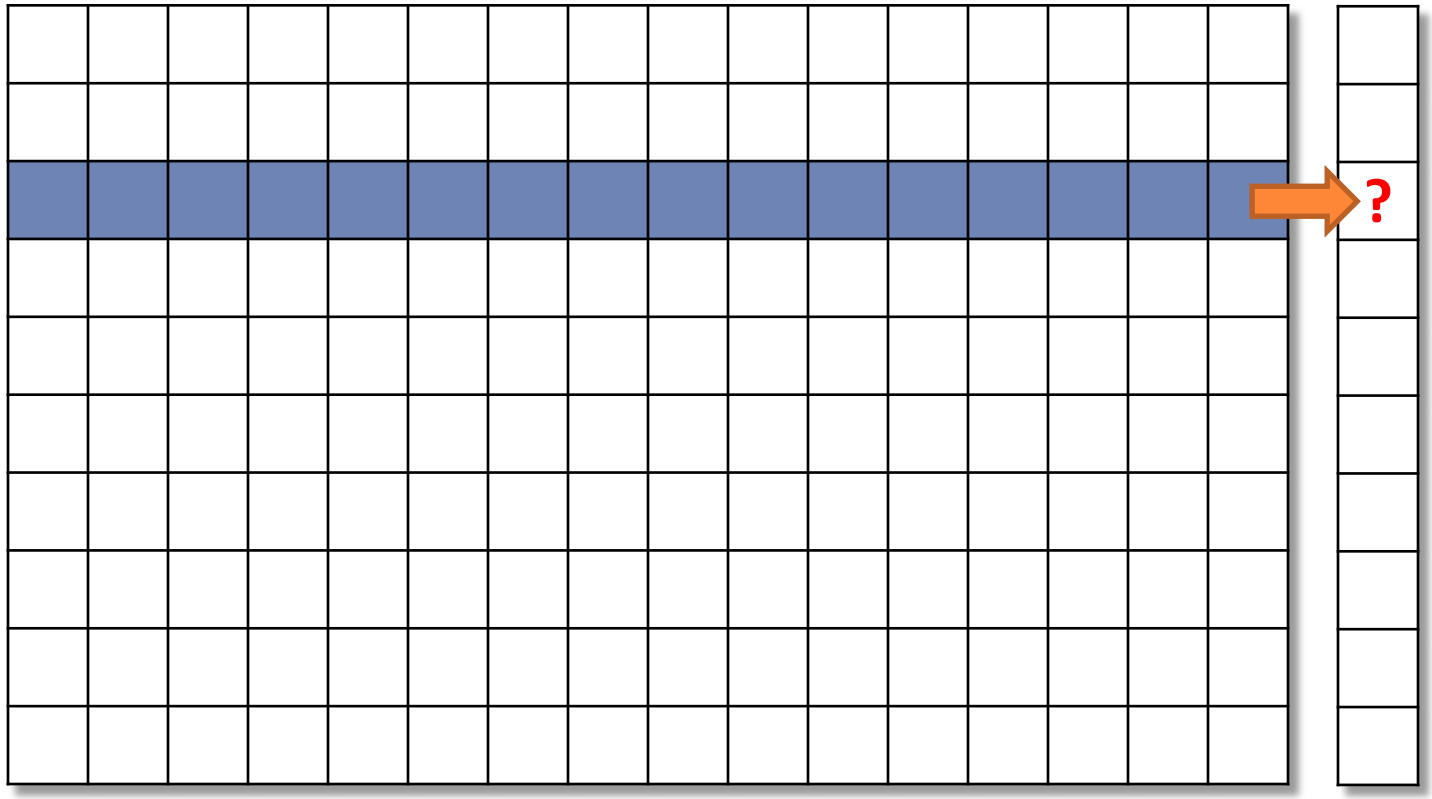
Entropy value for each row



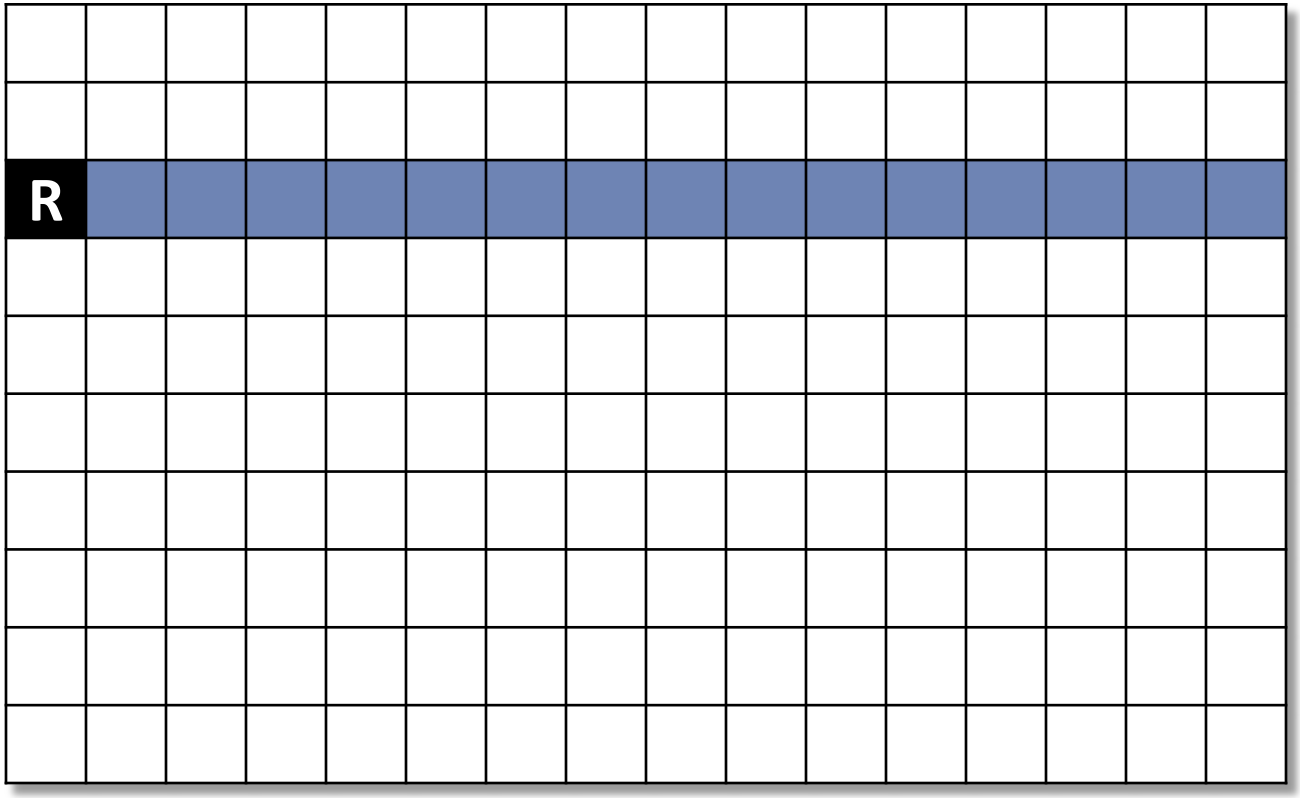
HEX dump

16 x 10 bytes

For this row

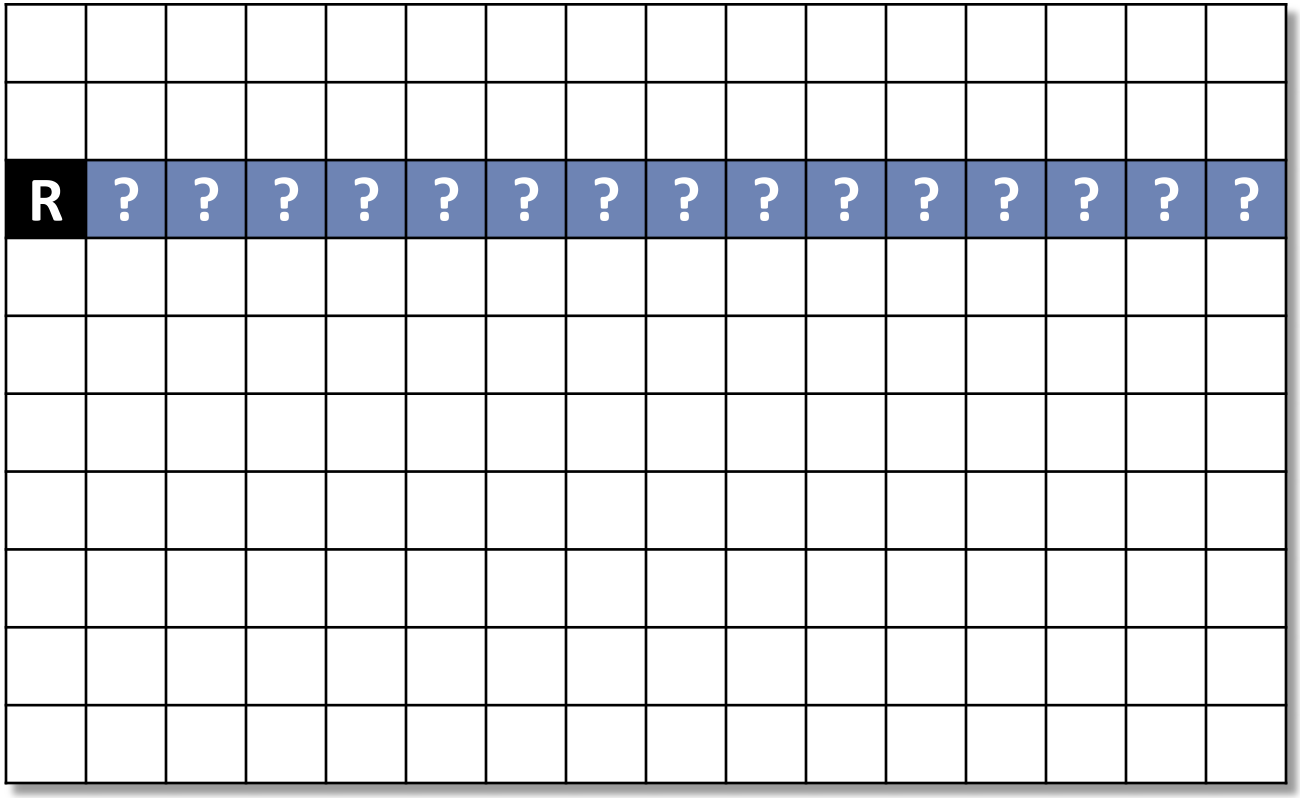


The first byte



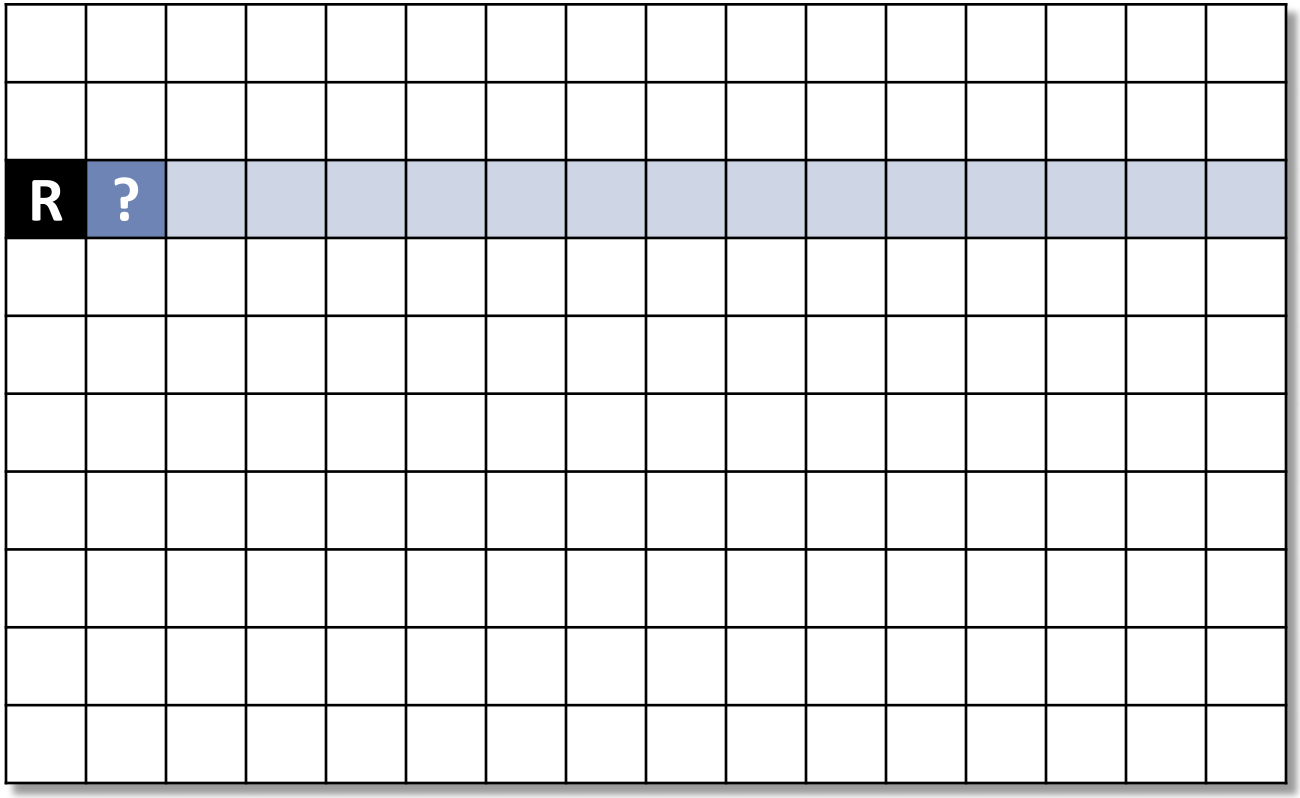
Step = 1, forward

Entropy = 15



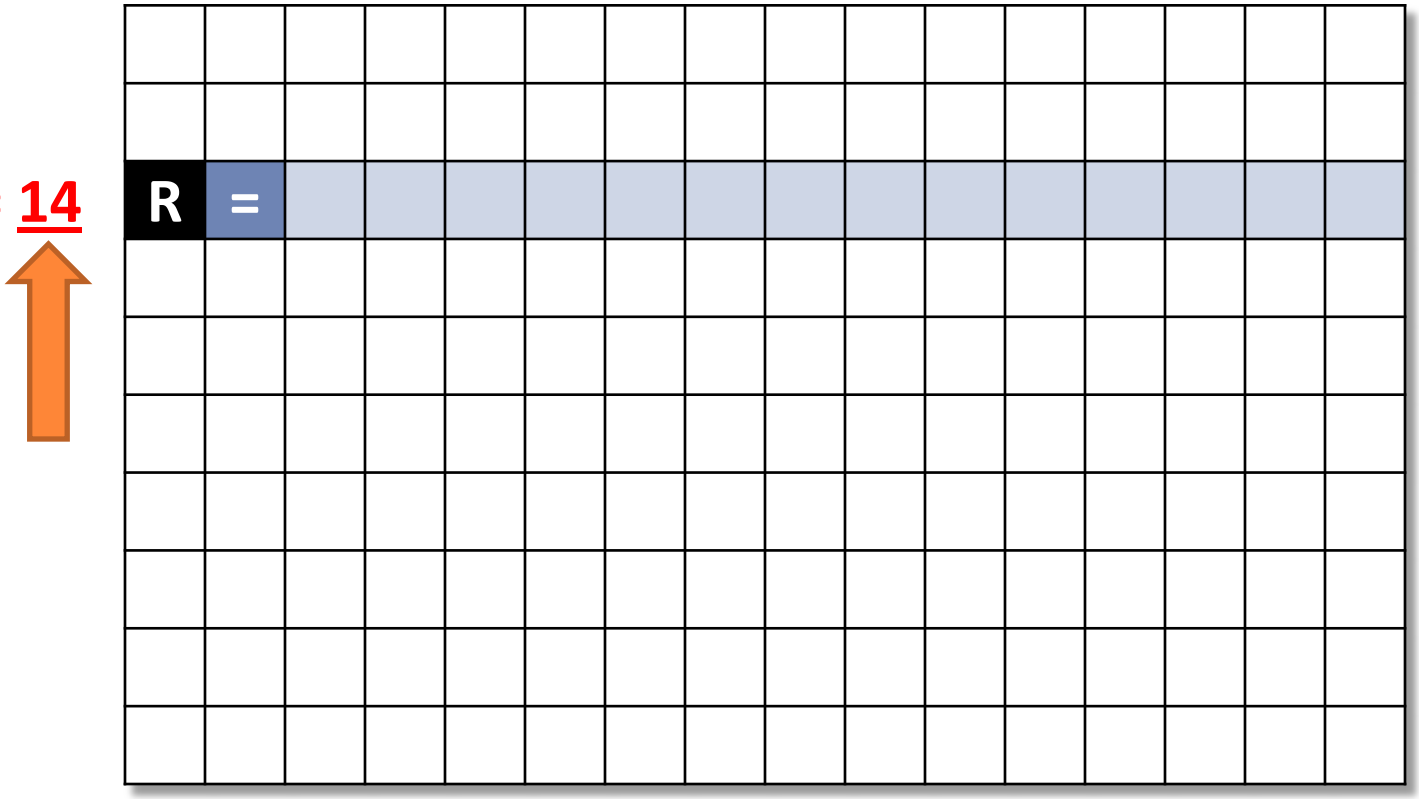
Step = 1, forward

Entropy = 15



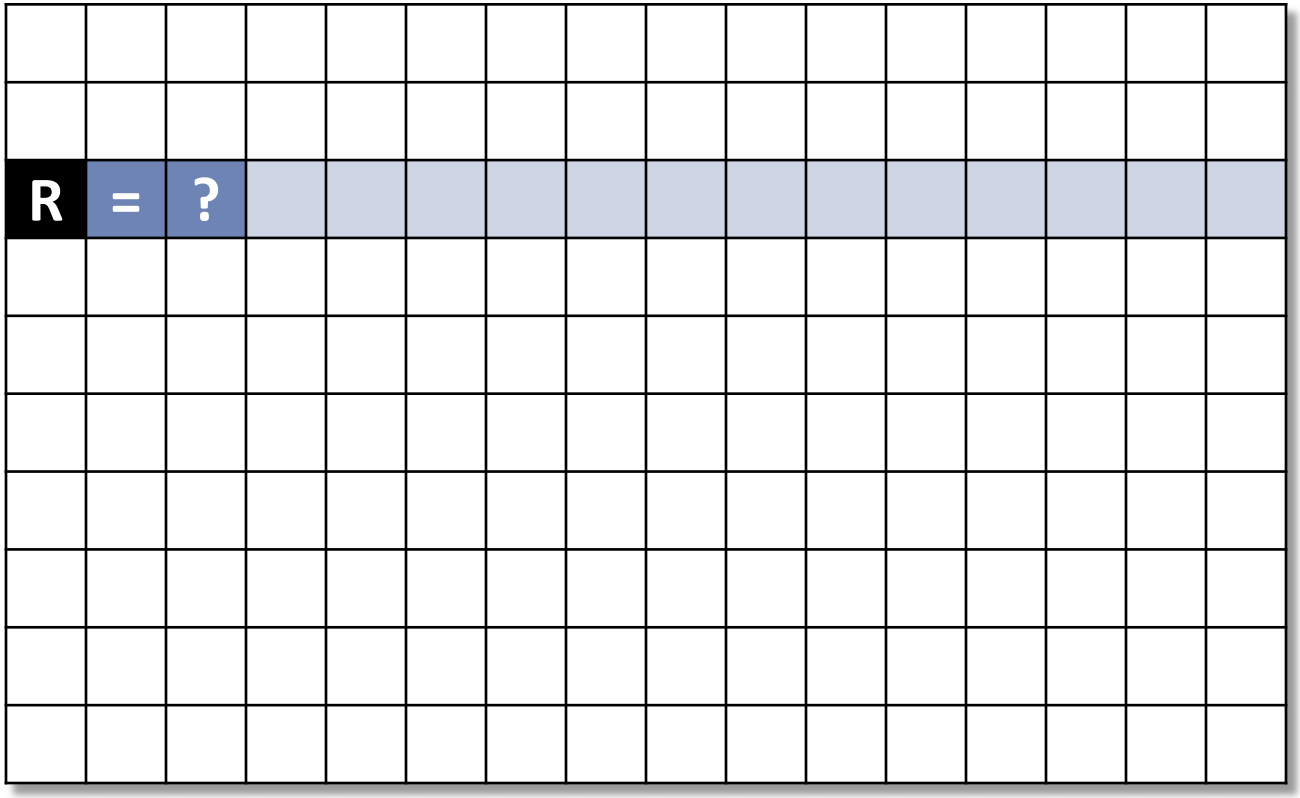
Step = 1, forward

Entropy = 14



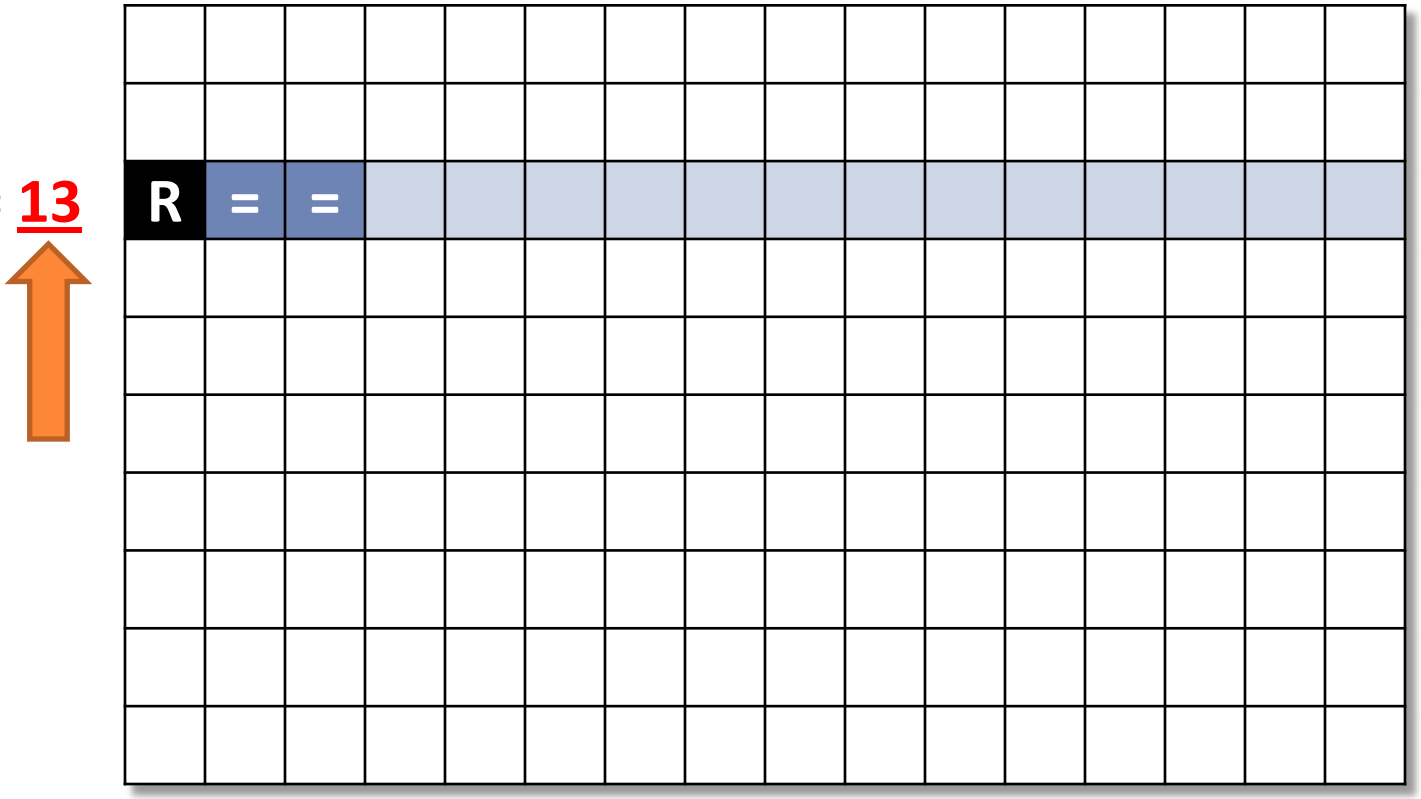
Step = 1, forward

Entropy = 14



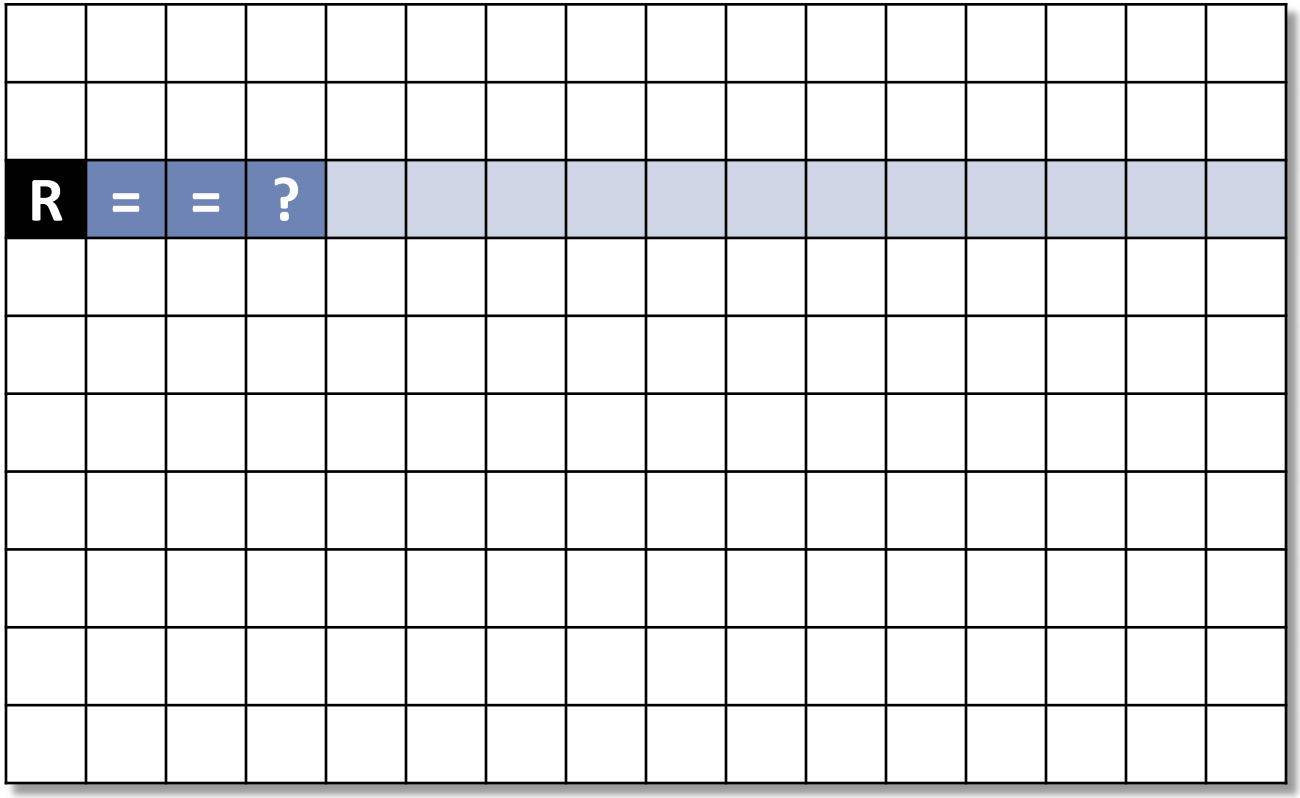
Step = 1, forward

Entropy = 13



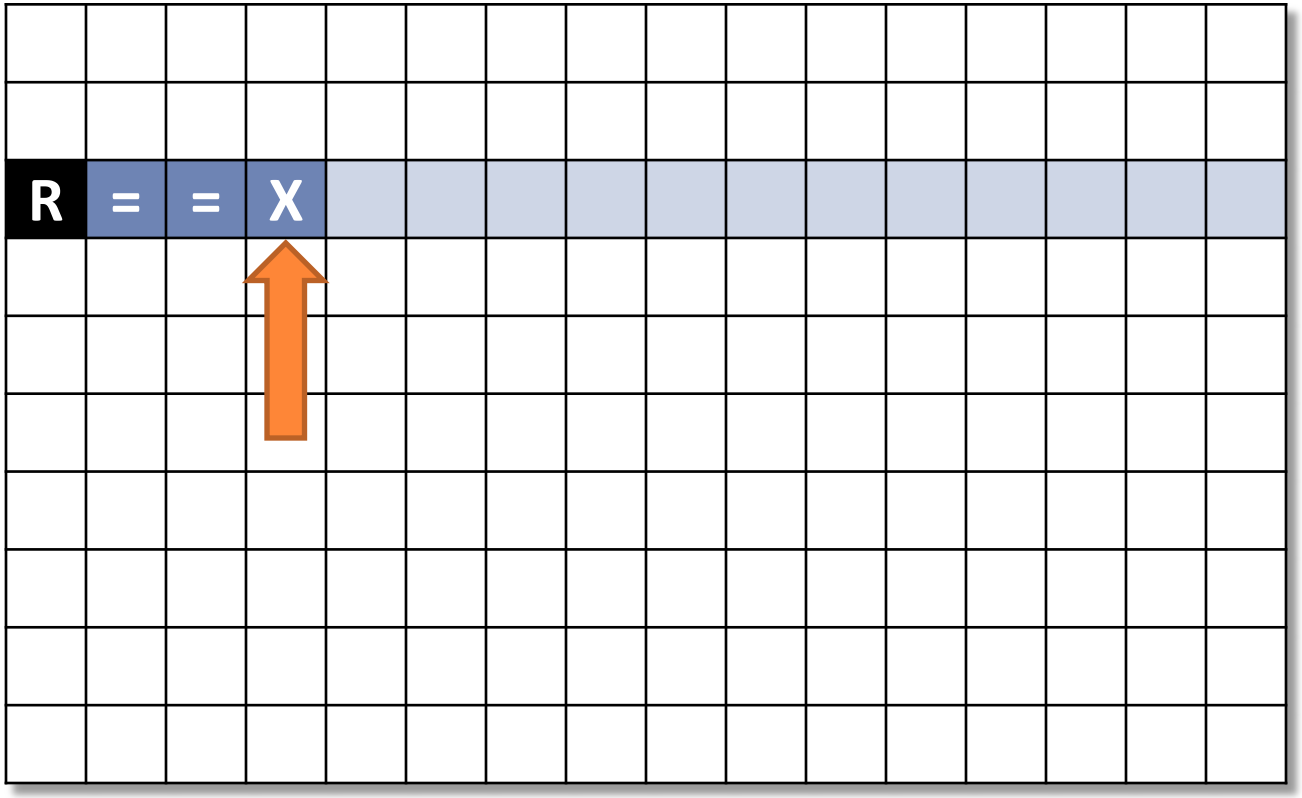
Step = 1, forward

Entropy = 13

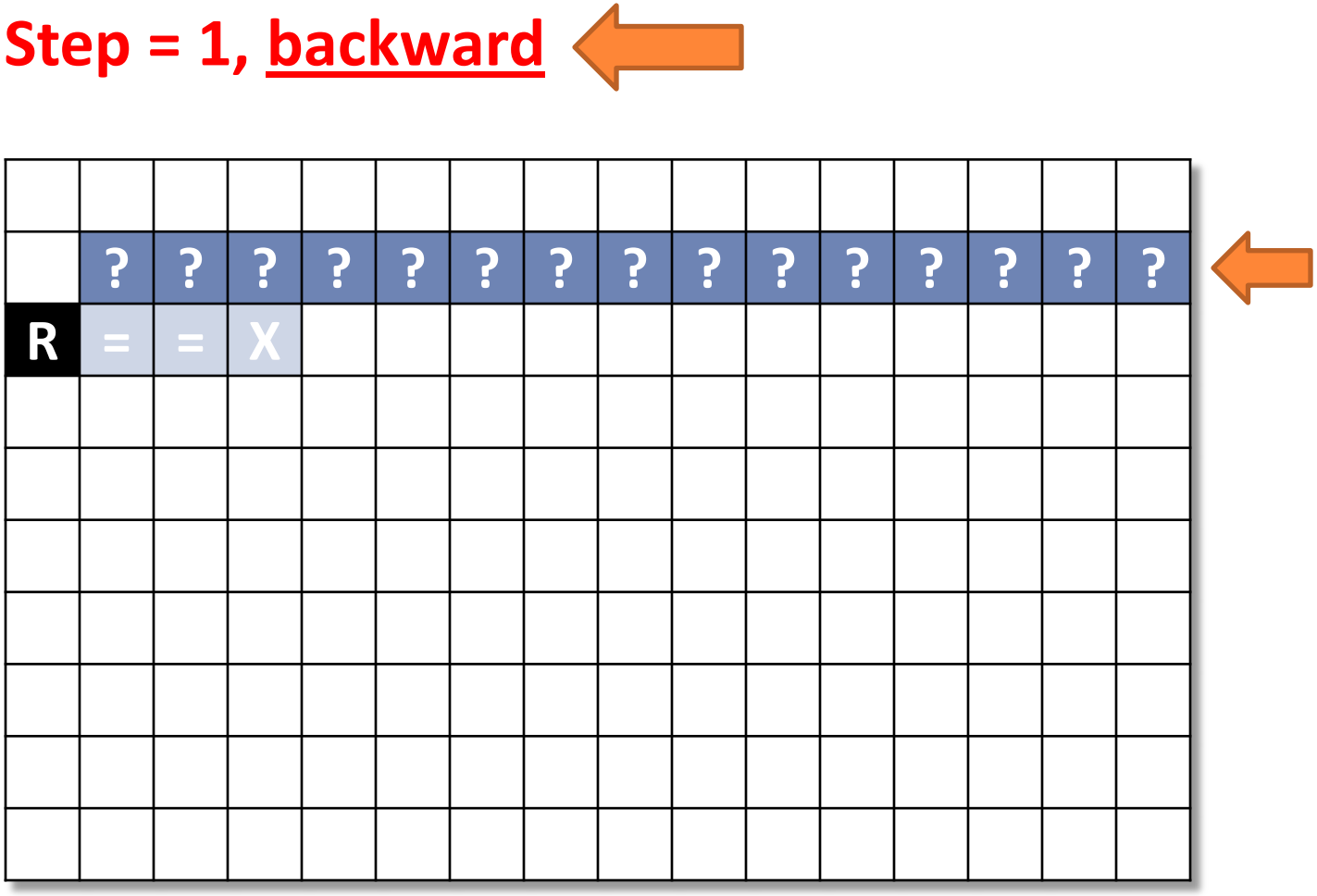


Step = 1, forward

Entropy = 13



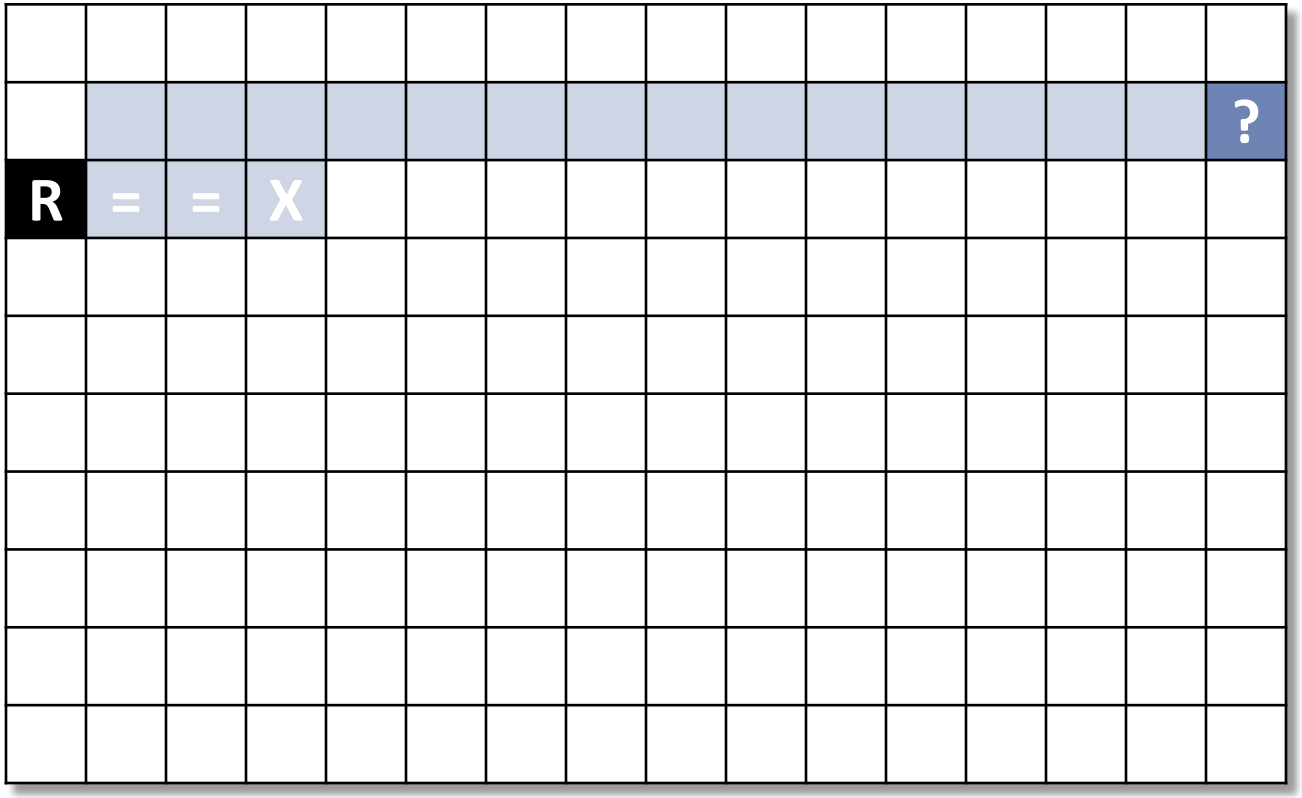
Step = 1, backward



Entropy = 13

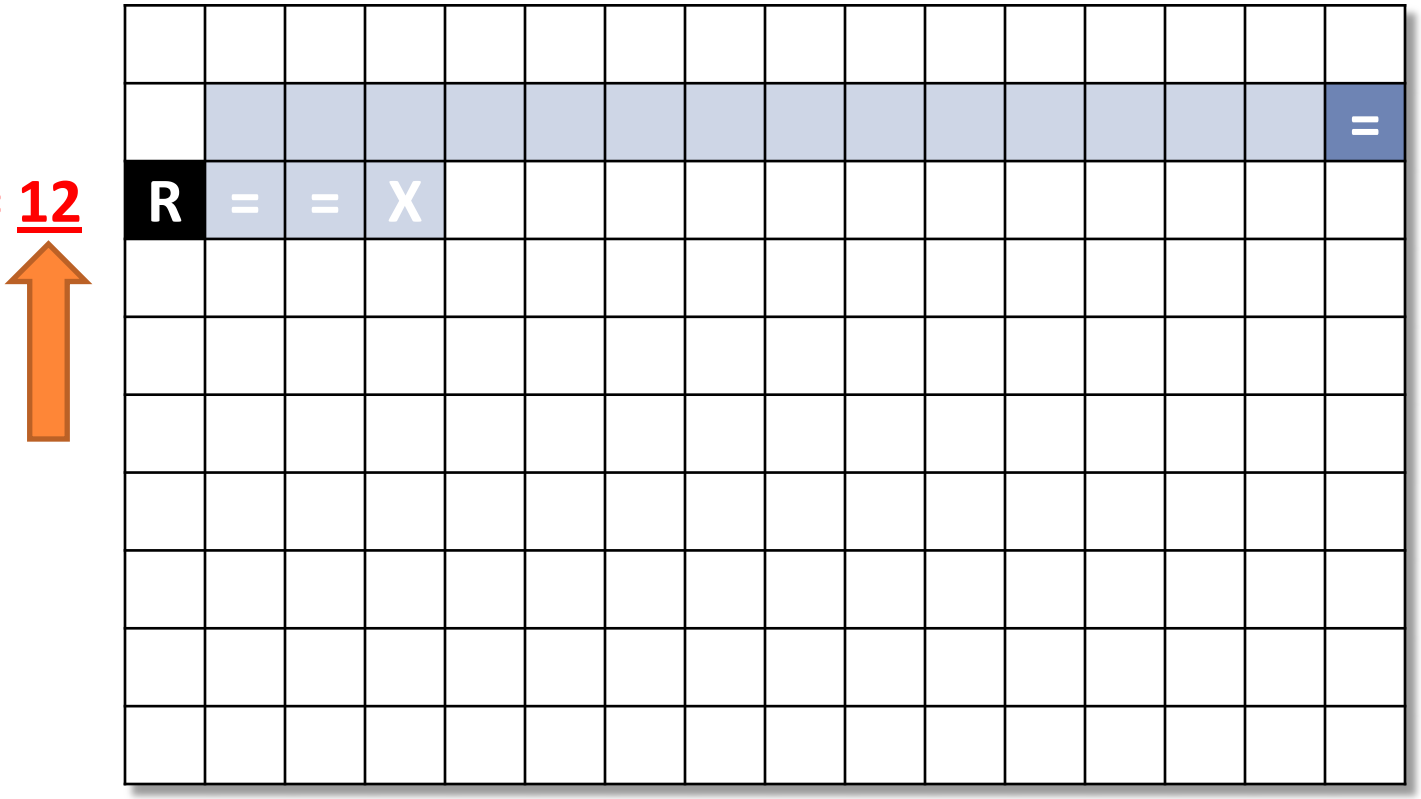
Step = 1, backward

Entropy = 13



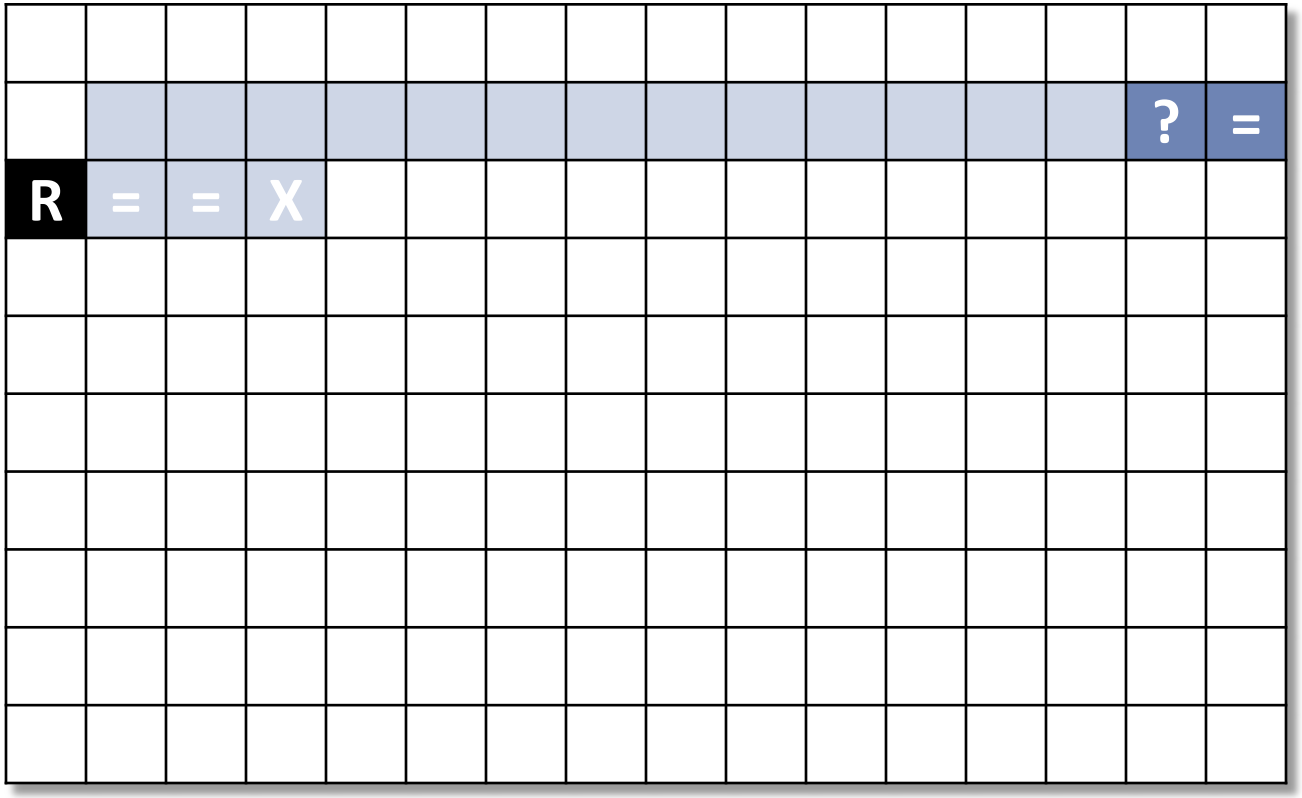
Step = 1, backward

Entropy = 12



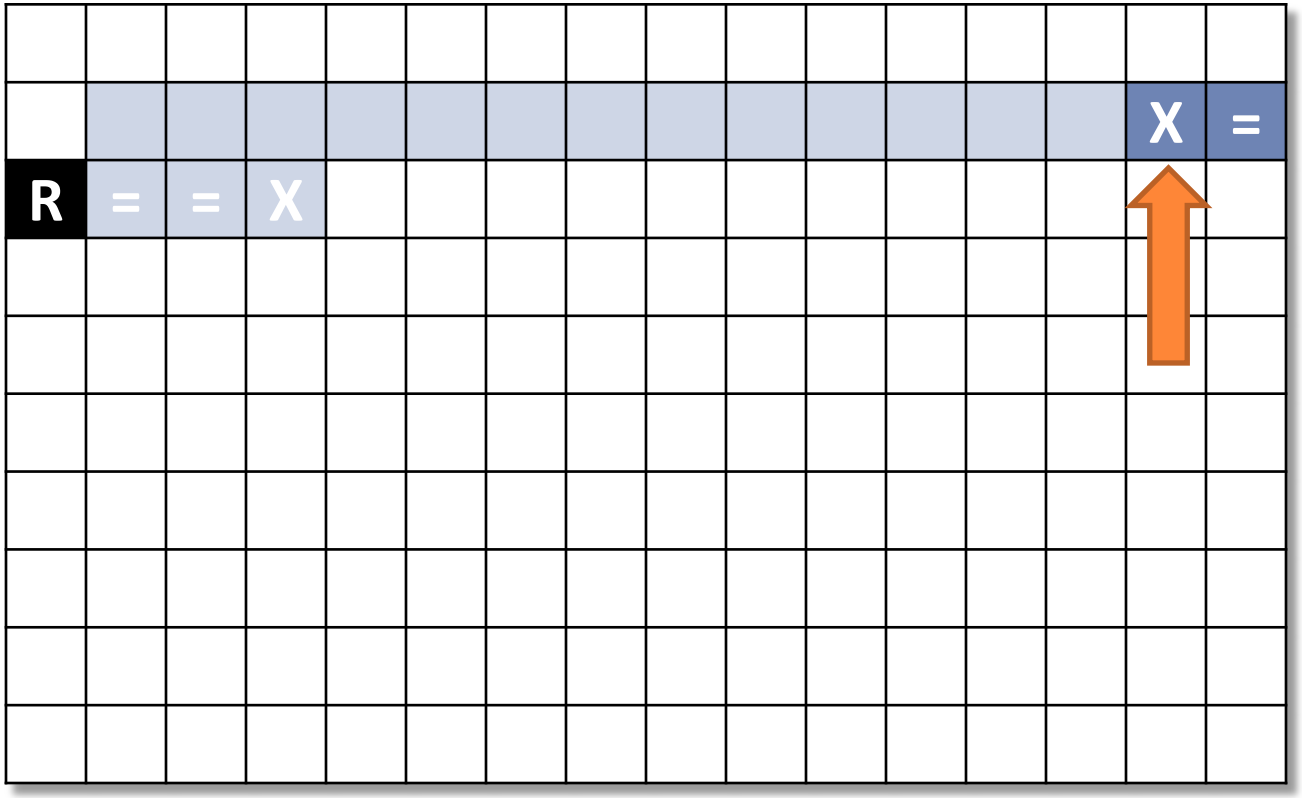
Step = 1, backward

Entropy = 12

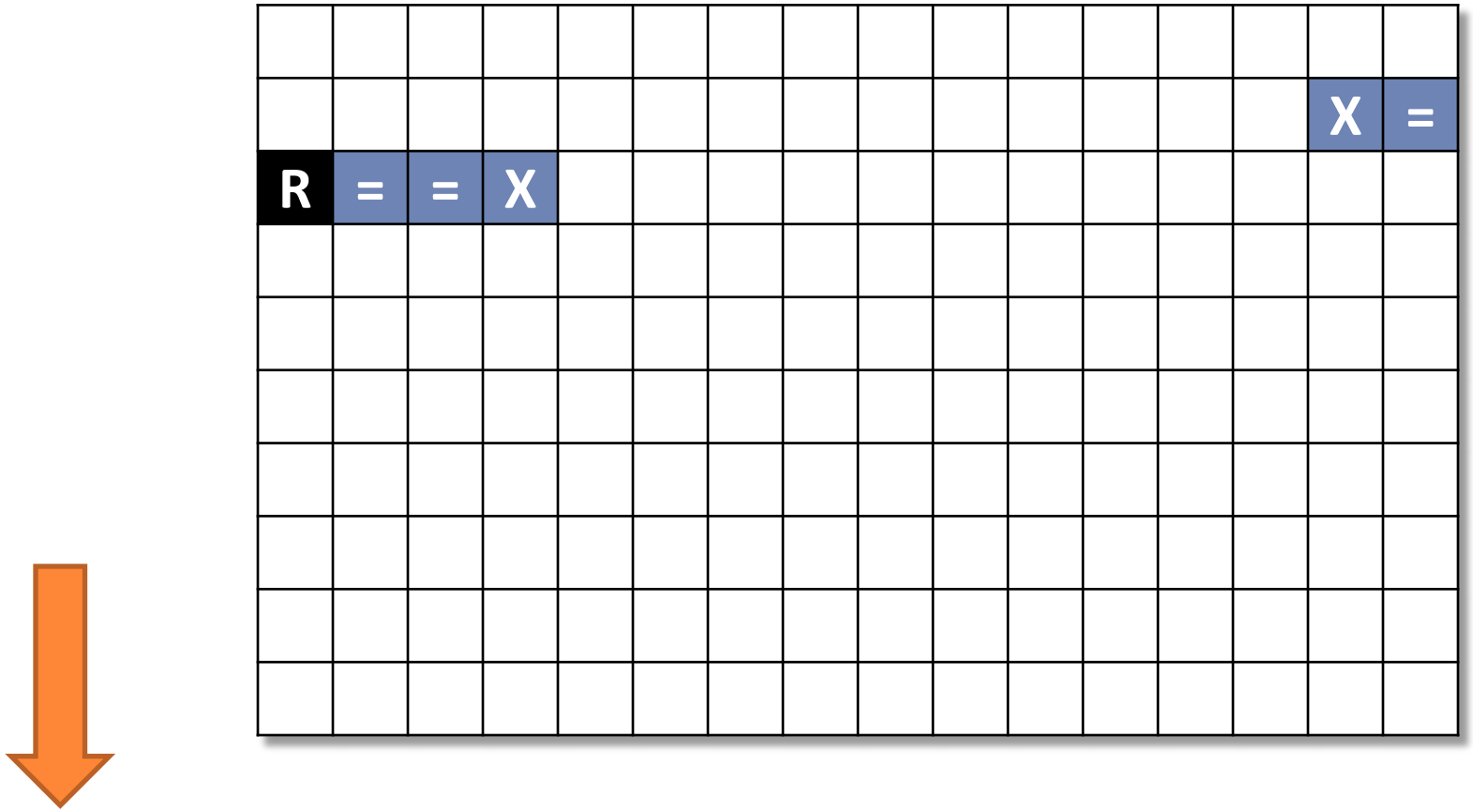


Step = 1, backward

Entropy = 12



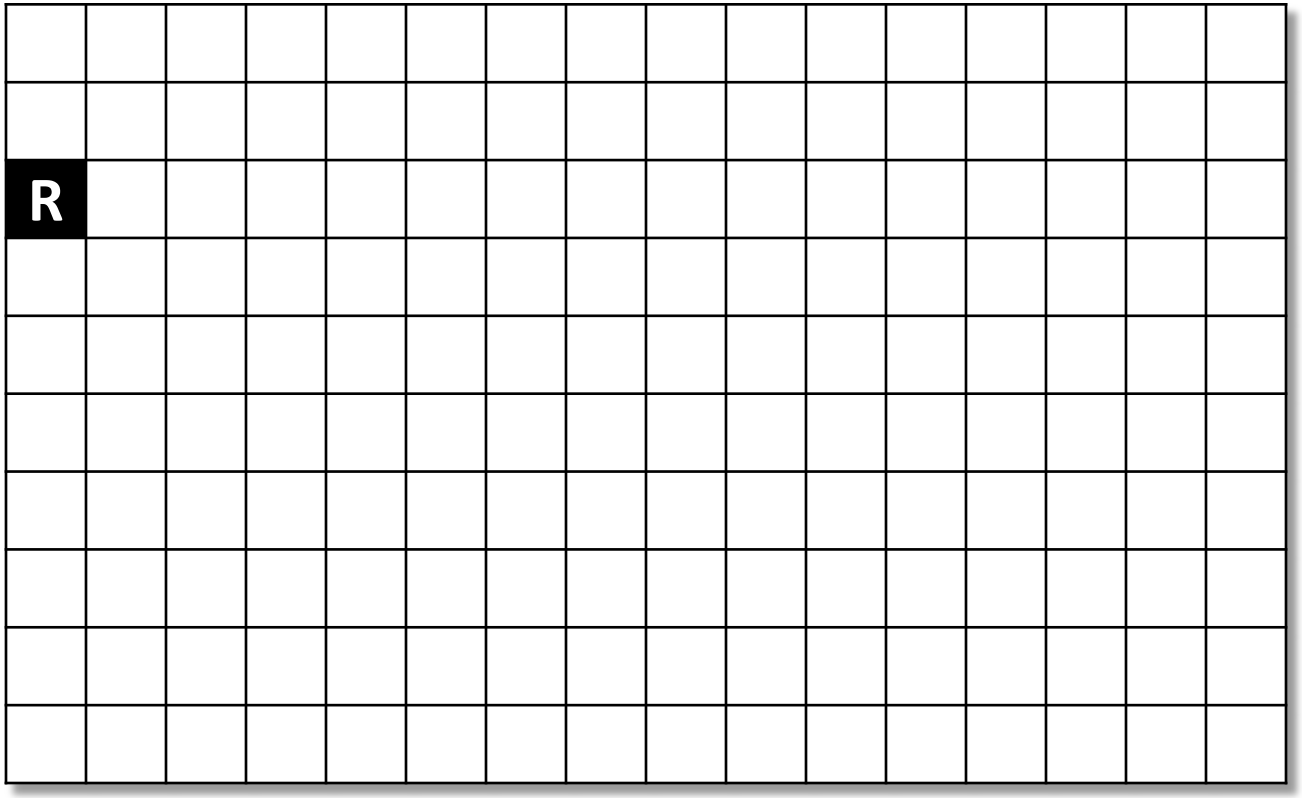
Step = 1, result



Entropy[1] = 12

Step = 2, forward

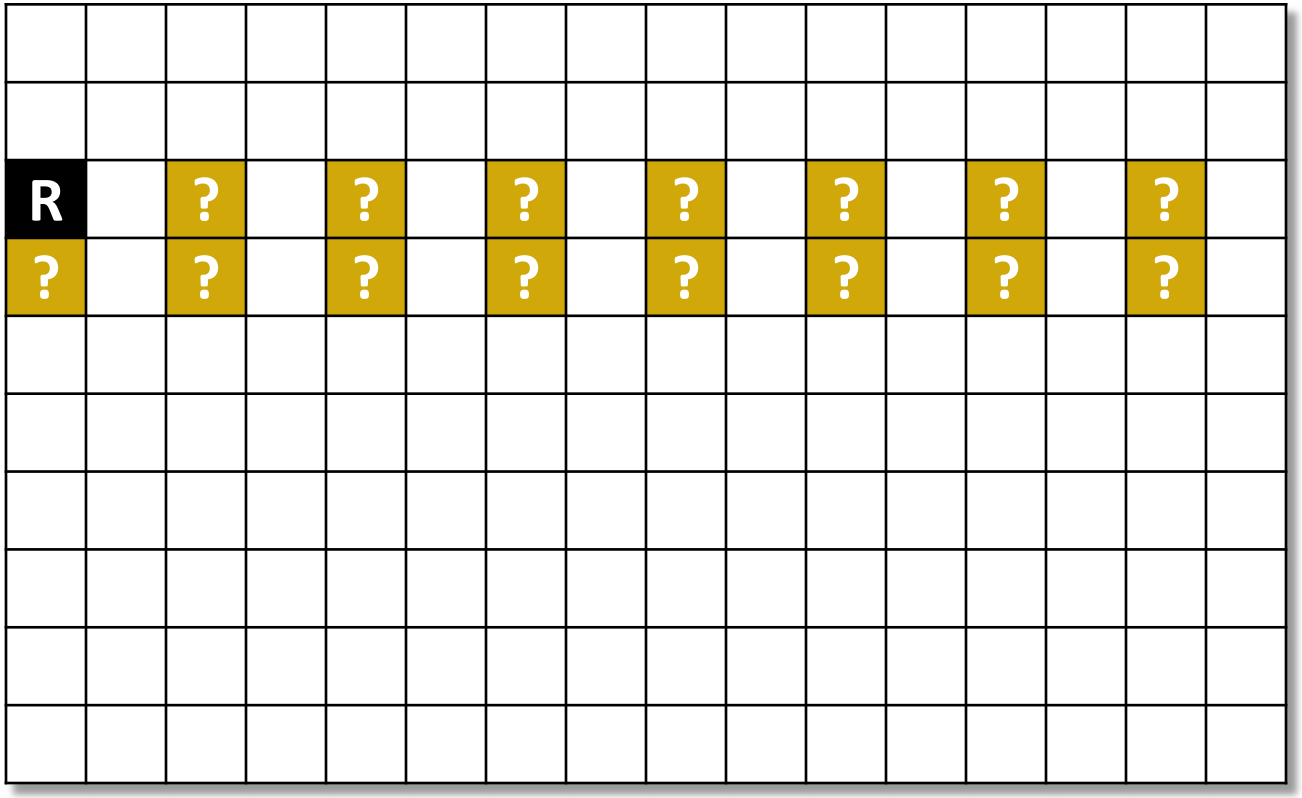
Entropy = 15



Entropy[1] = 12

Step = 2, forward

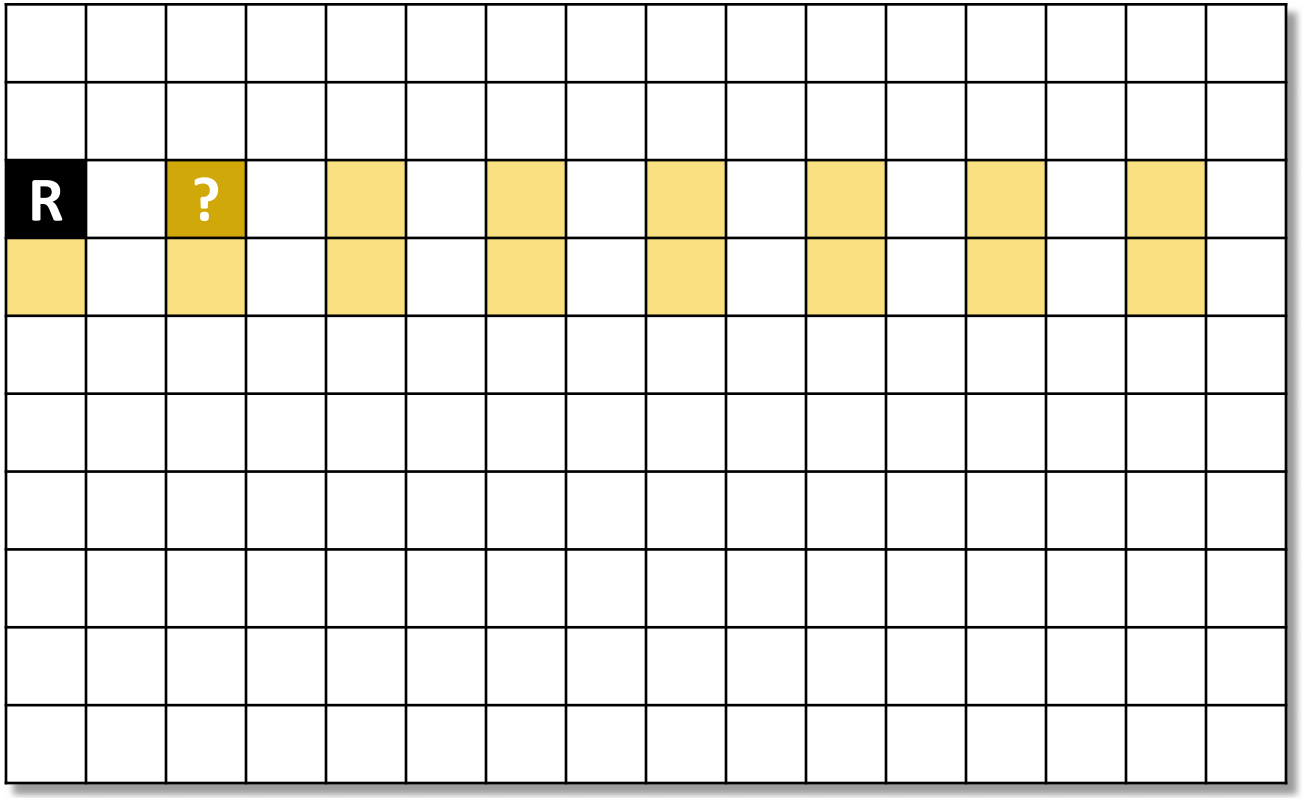
Entropy = 15



Entropy[1] = 12

Step = 2, forward

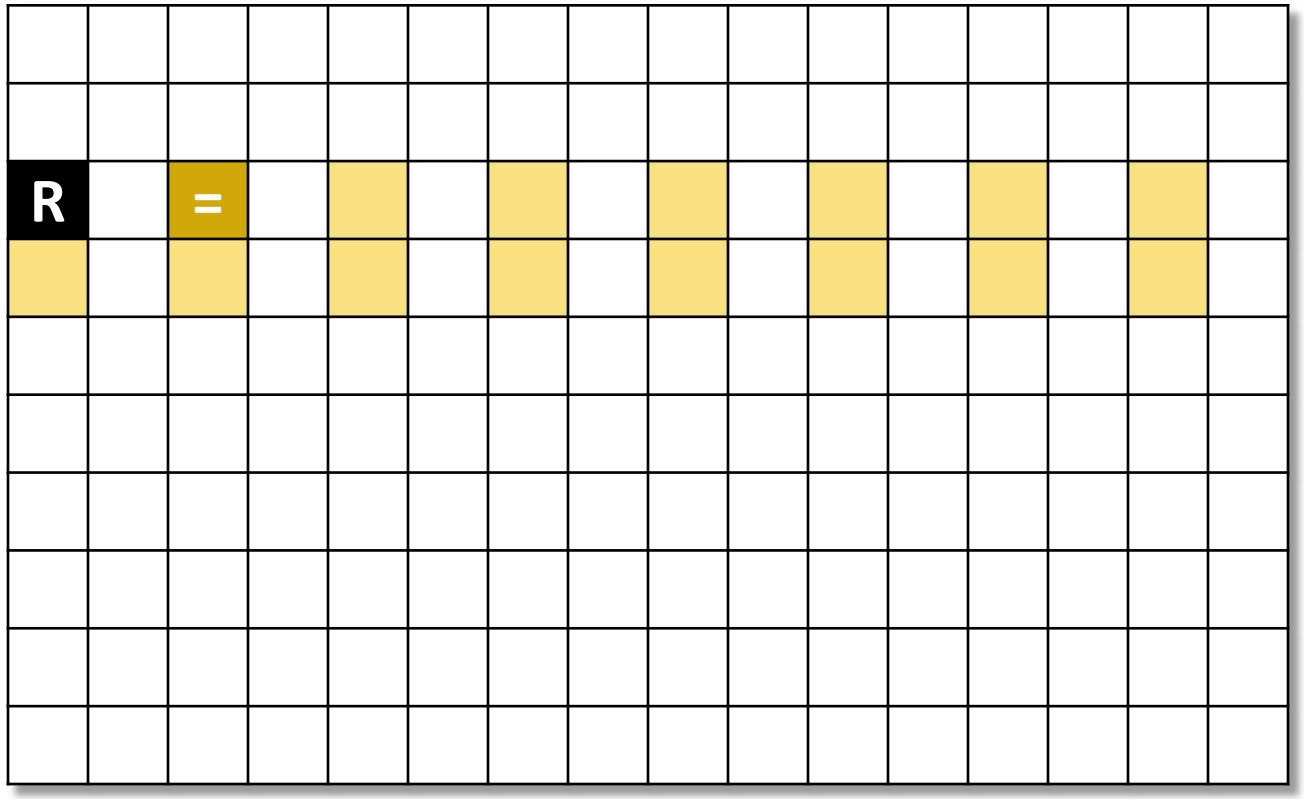
Entropy = 15



Entropy[1] = 12

Step = 2, forward

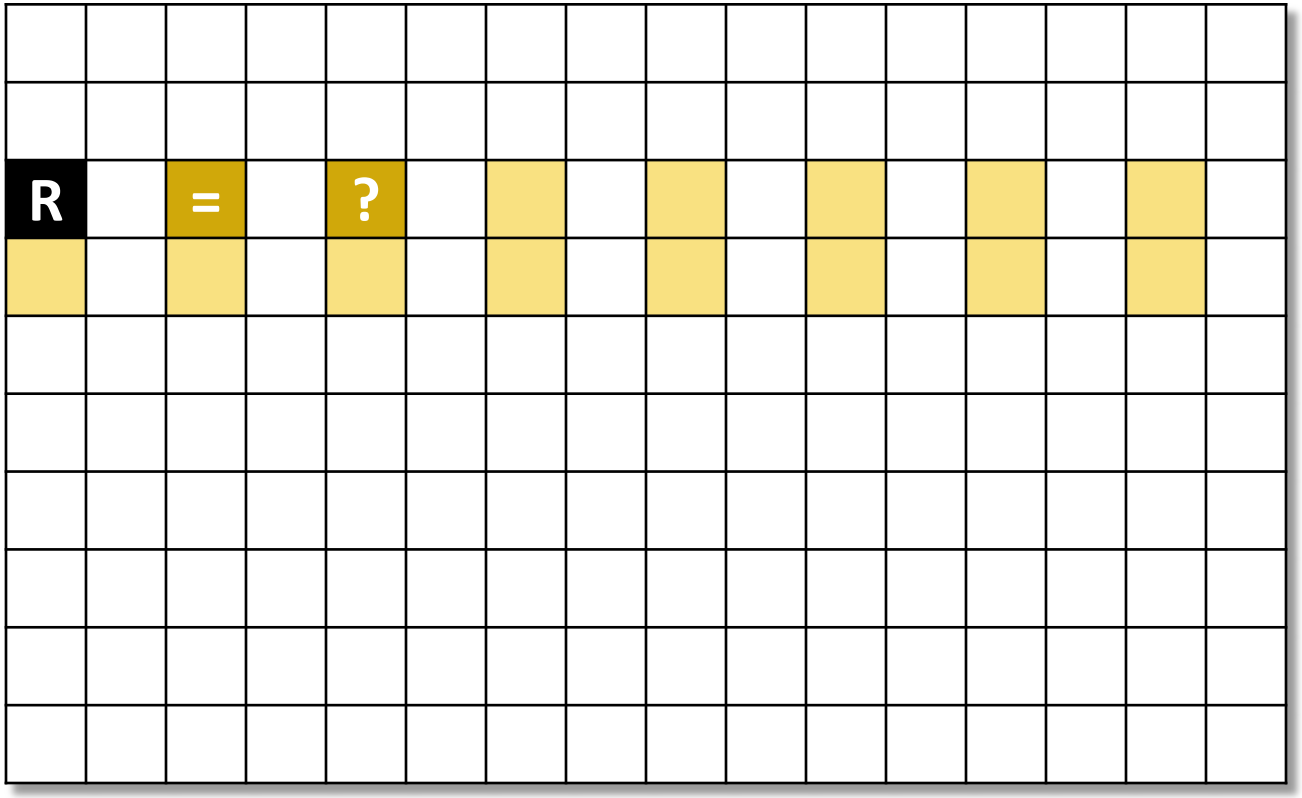
Entropy = 14



Entropy[1] = 12

Step = 2, forward

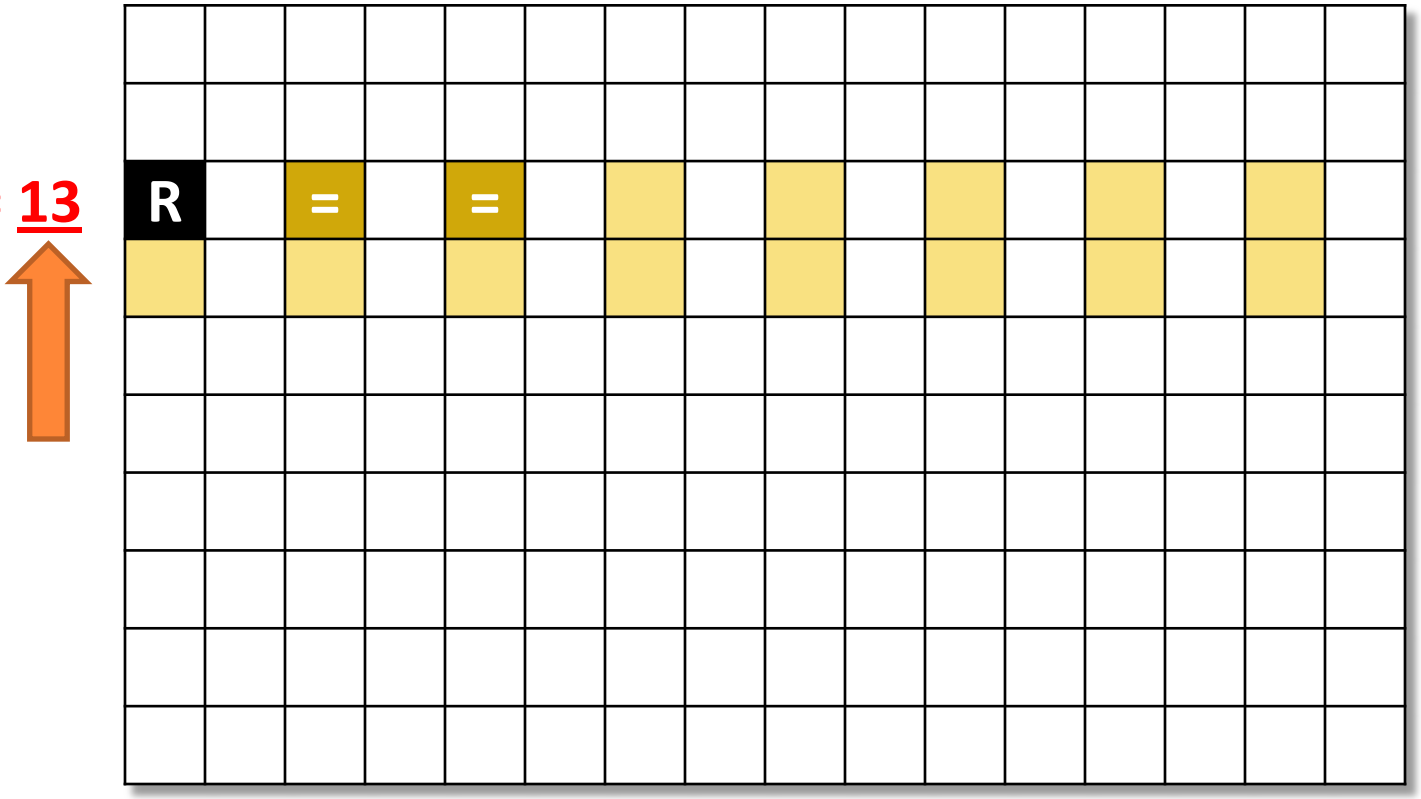
Entropy = 14



Entropy[1] = 12

Step = 2, forward

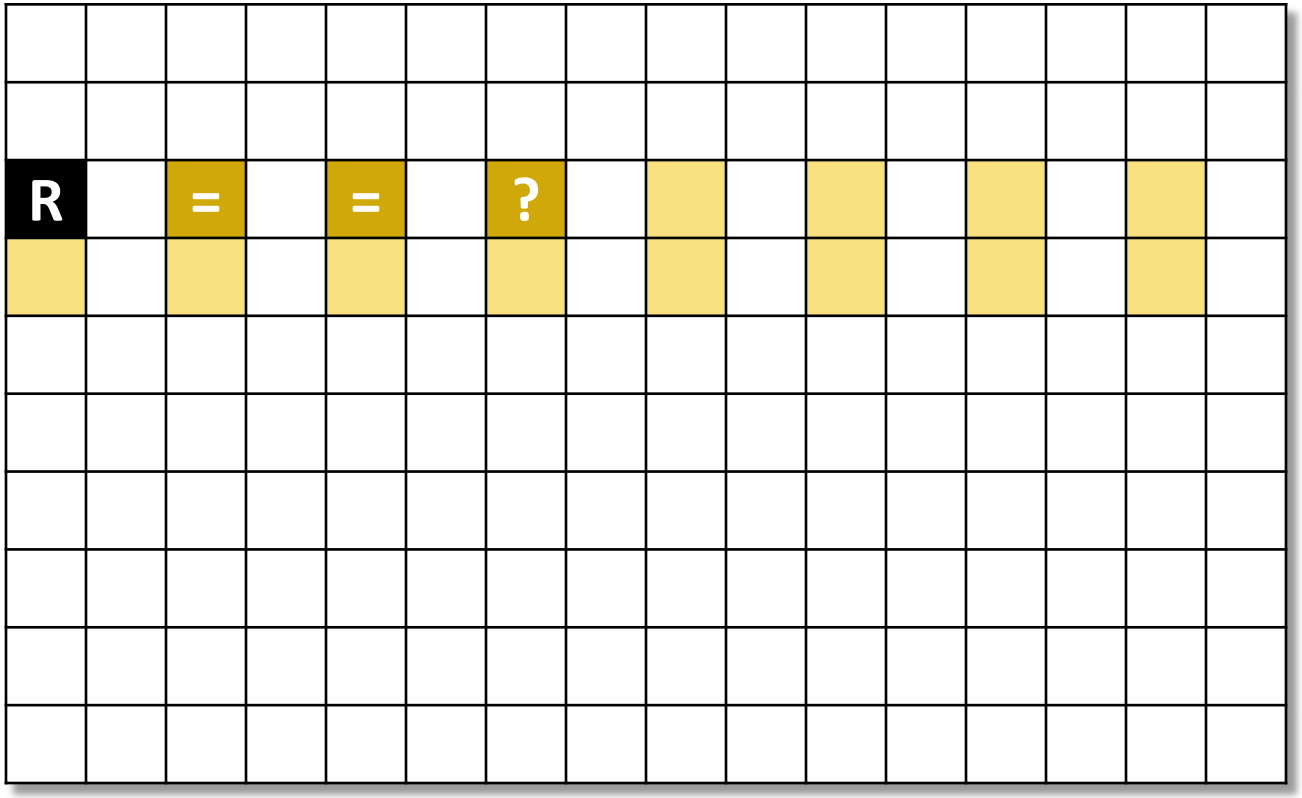
Entropy = 13



Entropy[1] = 12

Step = 2, forward

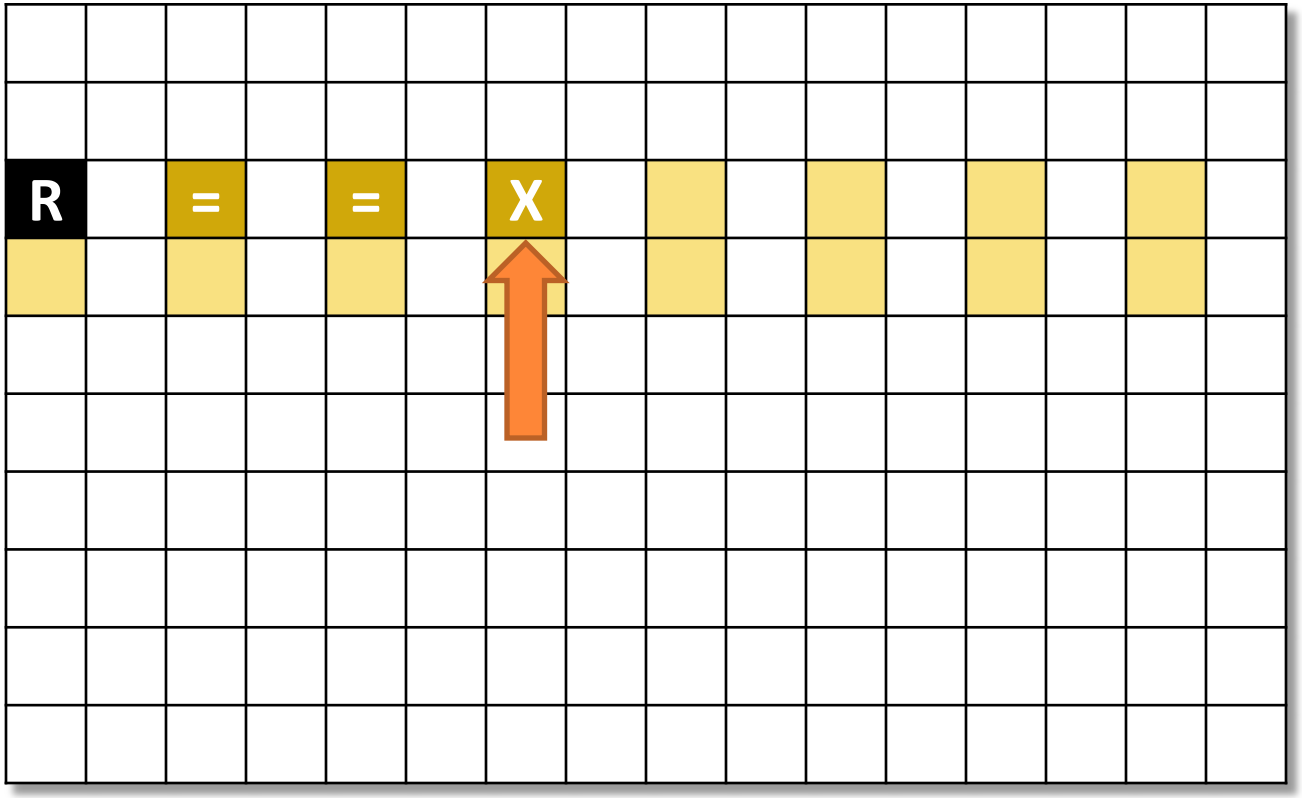
Entropy = 13



Entropy[1] = 12

Step = 2, forward

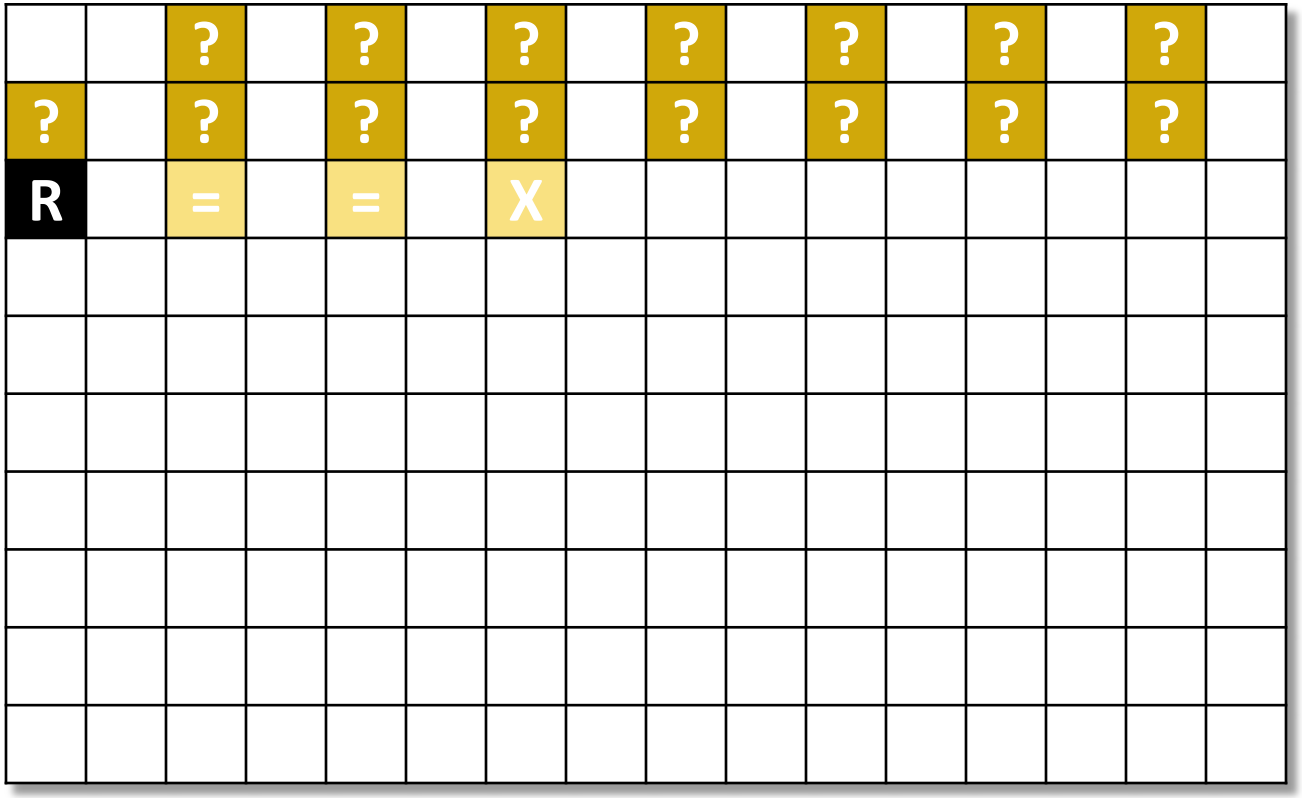
Entropy = 13



Entropy[1] = 12

Step = 2, backward

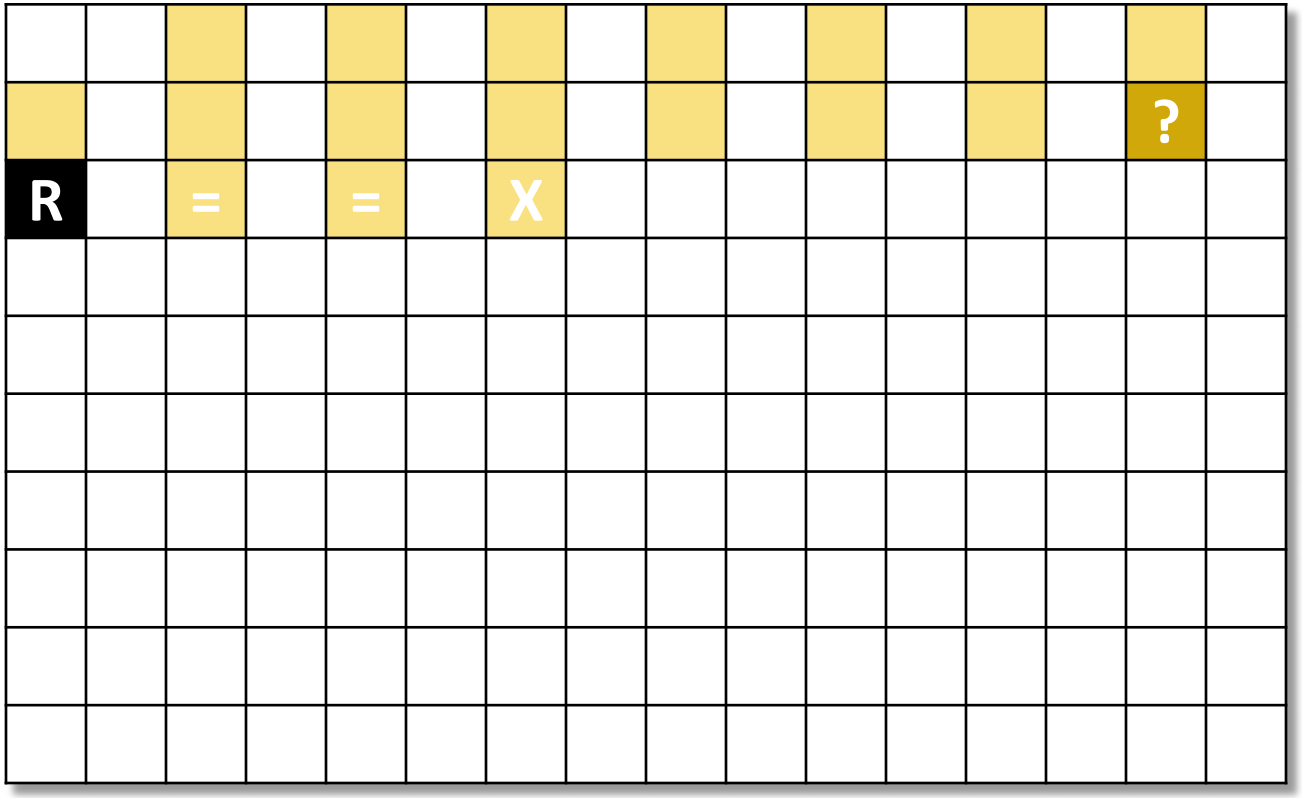
Entropy = 13



Entropy[1] = 12

Step = 2, backward

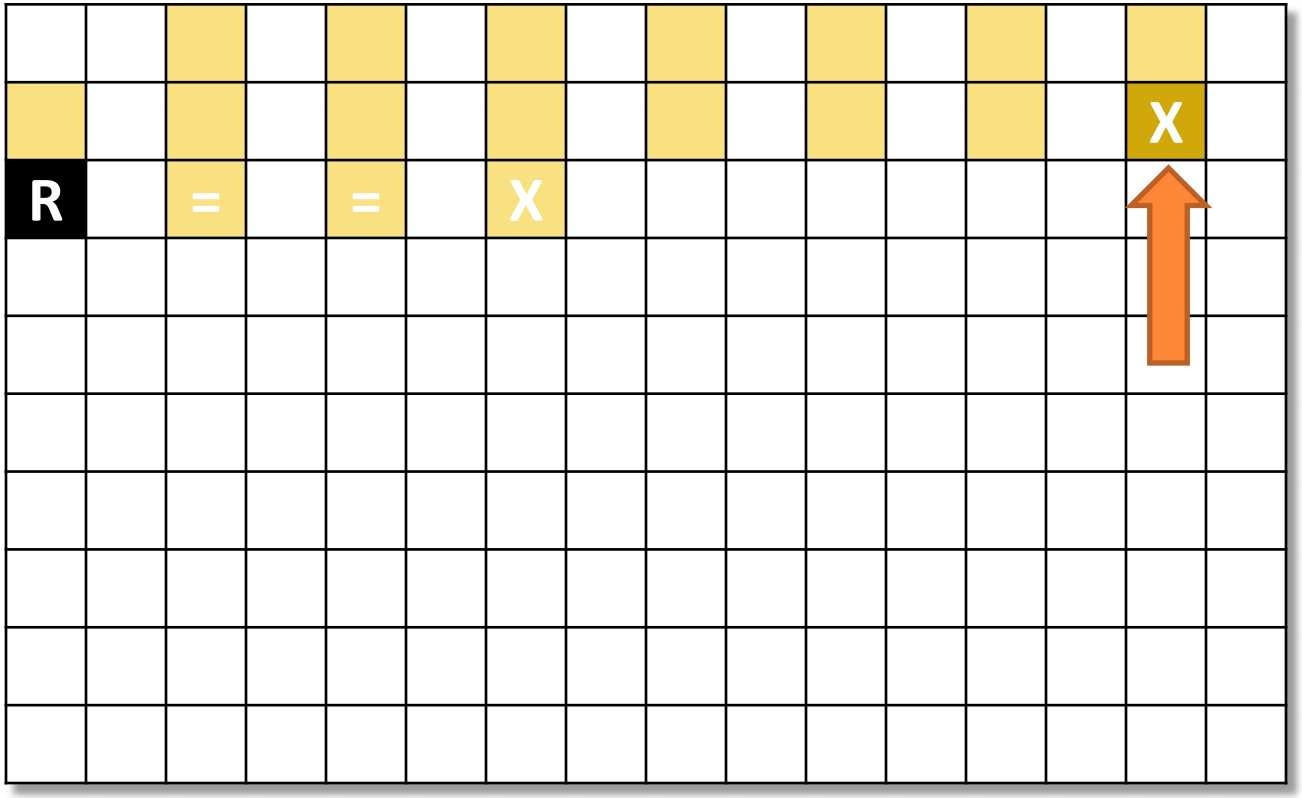
Entropy = 13



Entropy[1] = 12

Step = 2, backward

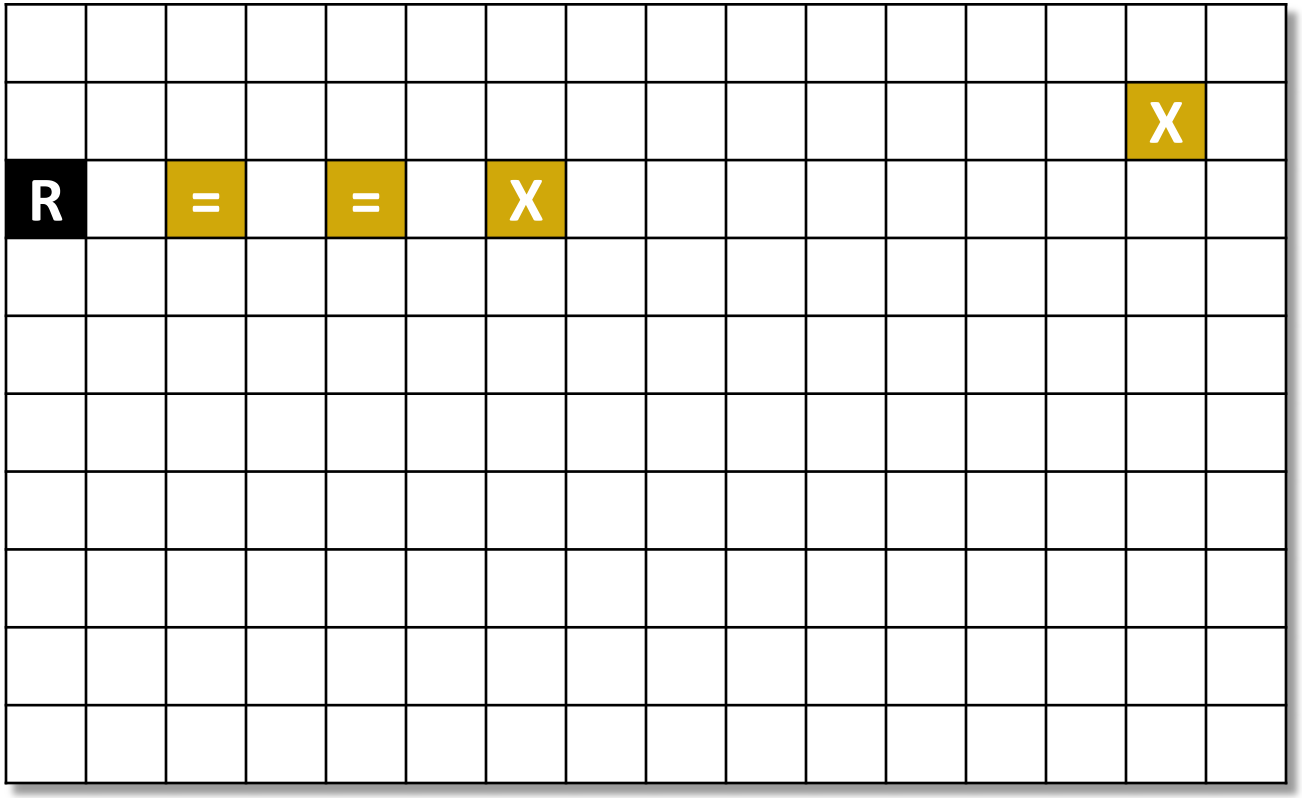
Entropy = 13



Entropy[1] = 12

Step = 2, result

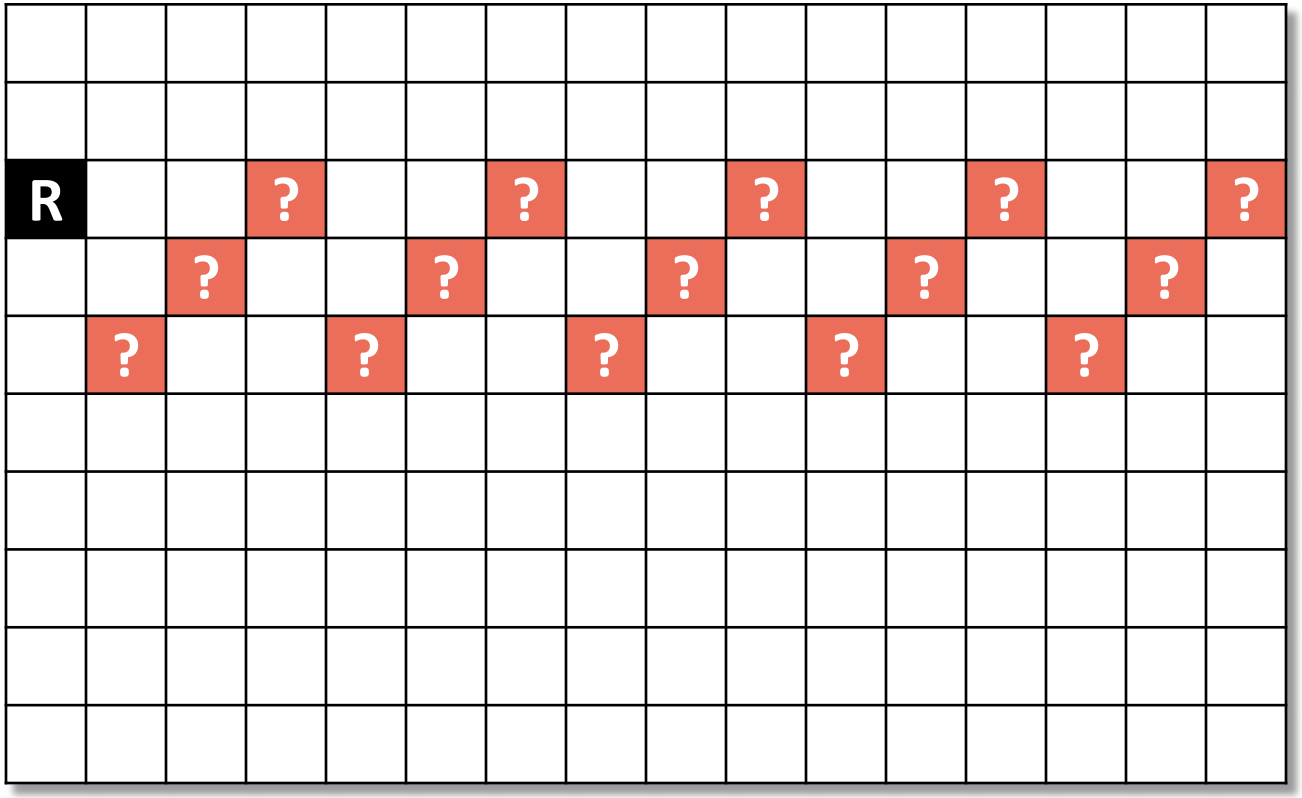
Entropy = 13



Entropy[1] = 12, 13

Step = 3, forward

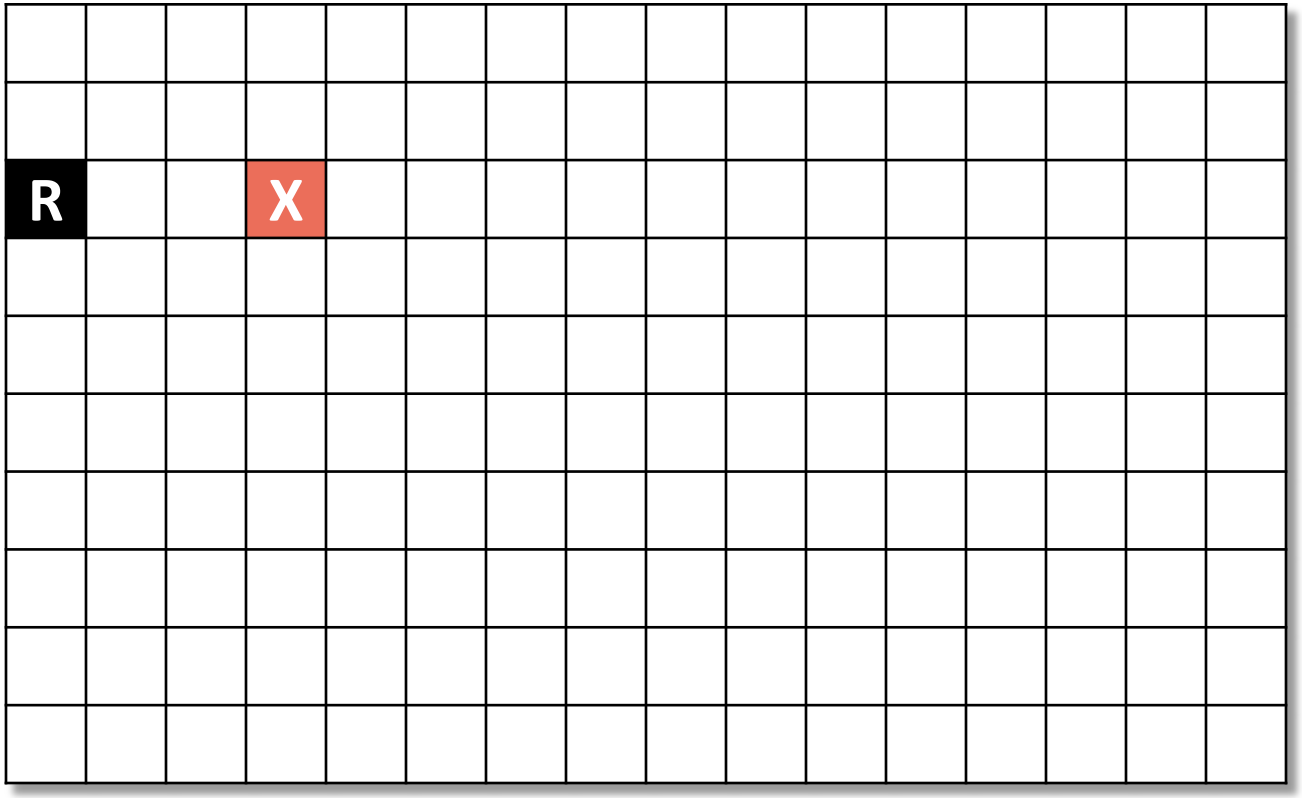
Entropy = 15



Entropy[1] = 12, 13

Step = 3, forward

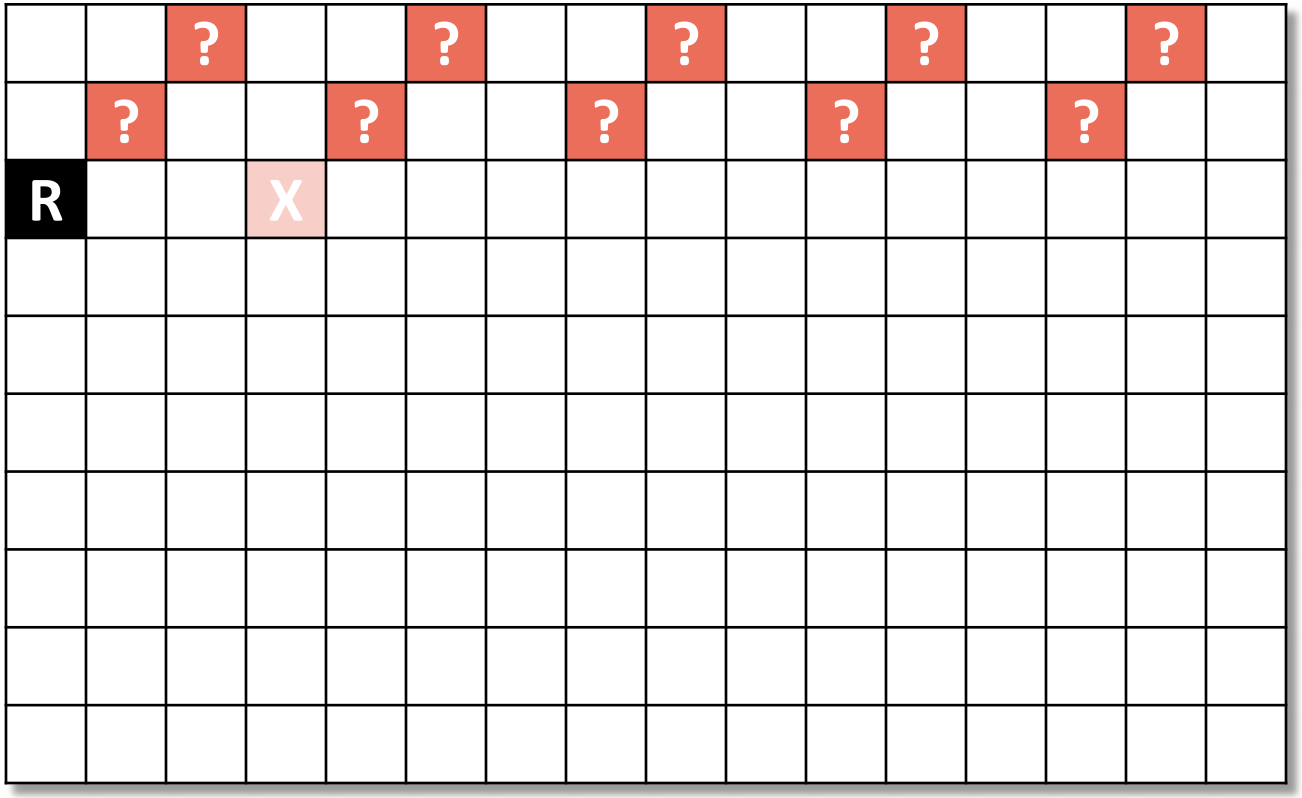
Entropy = 15



Entropy[1] = 12, 13

Step = 3, backward

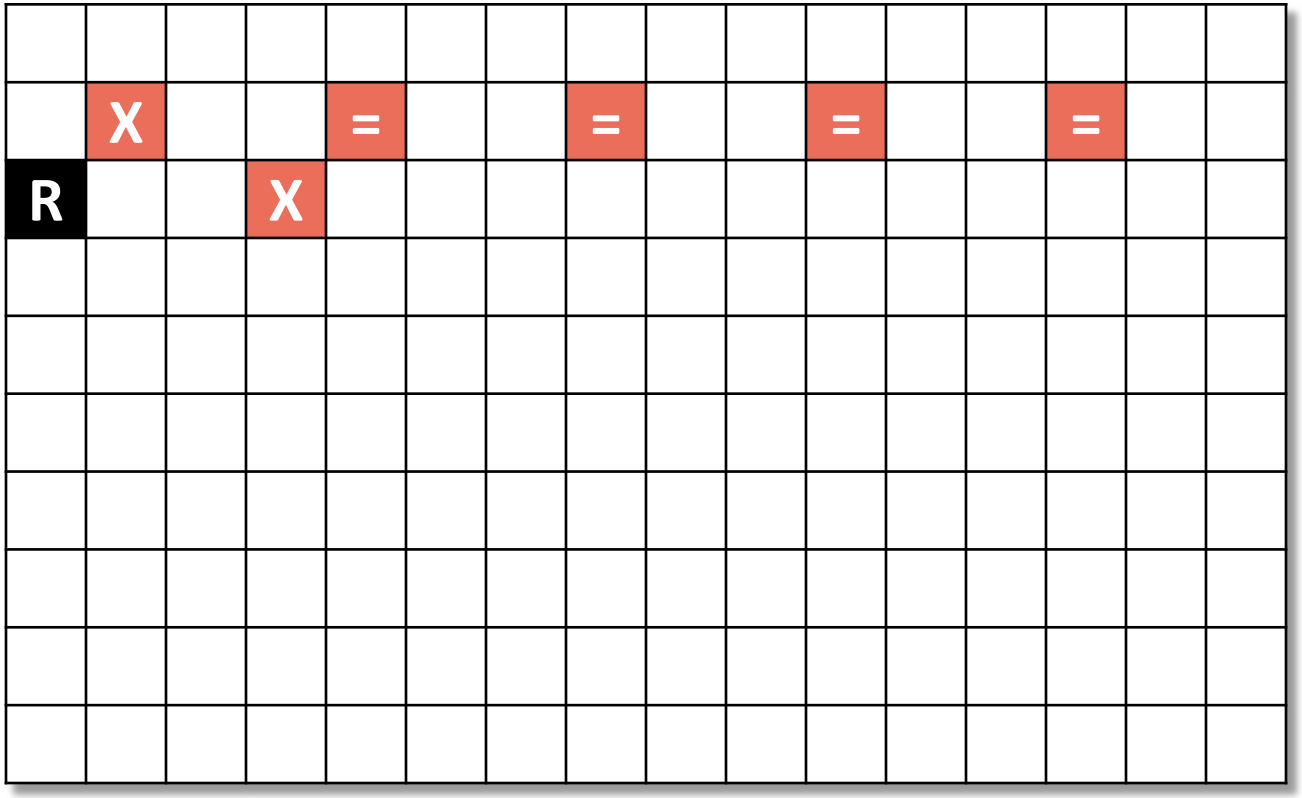
Entropy = 15



Entropy[1] = 12, 13

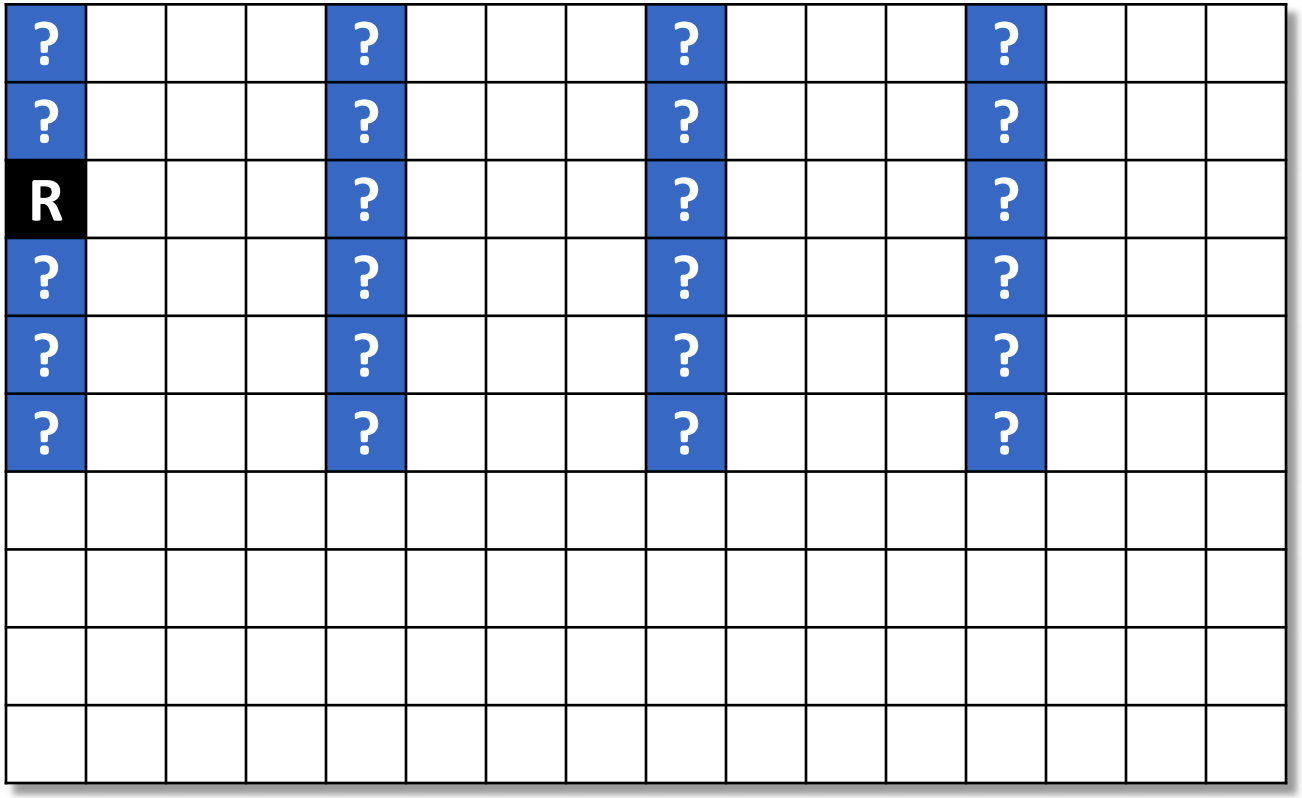
Step = 3, result

Entropy = 11



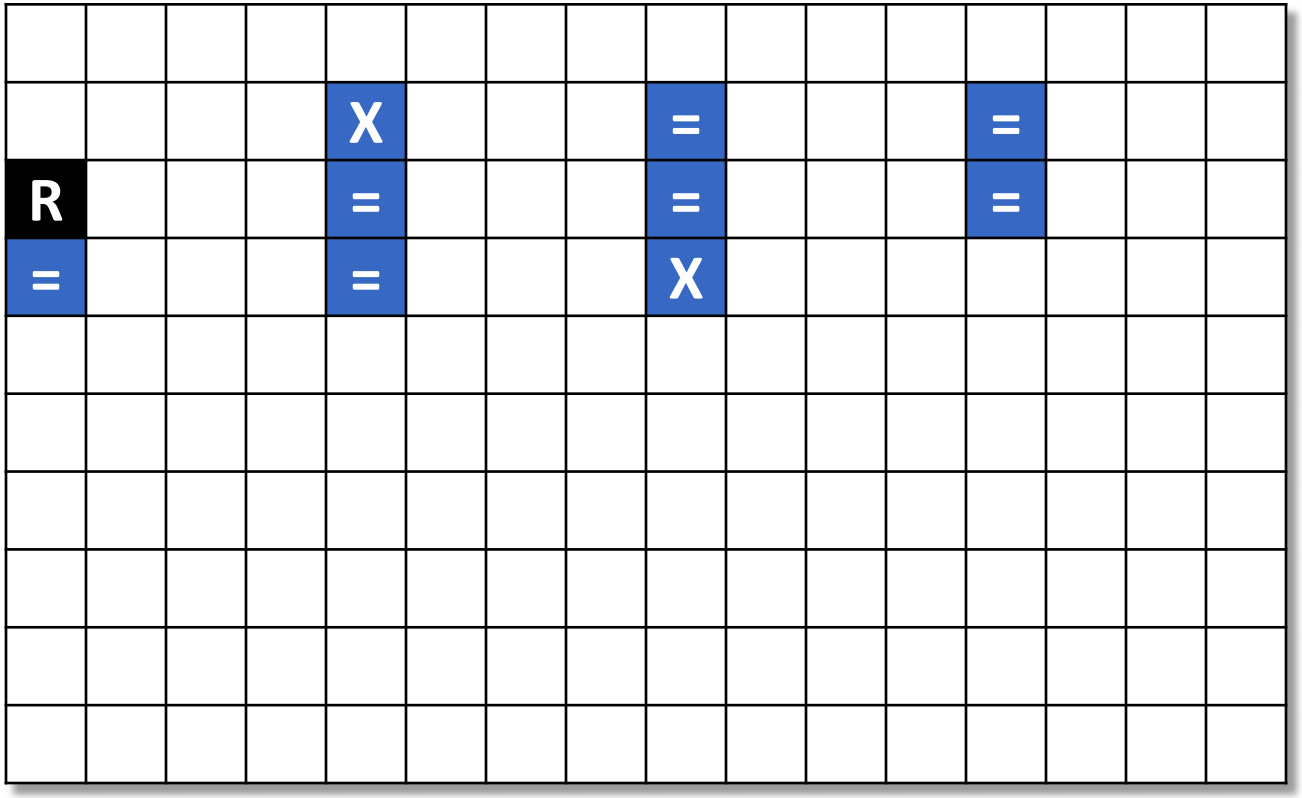
Entropy[1] = 12, 13, 11

Step = 4



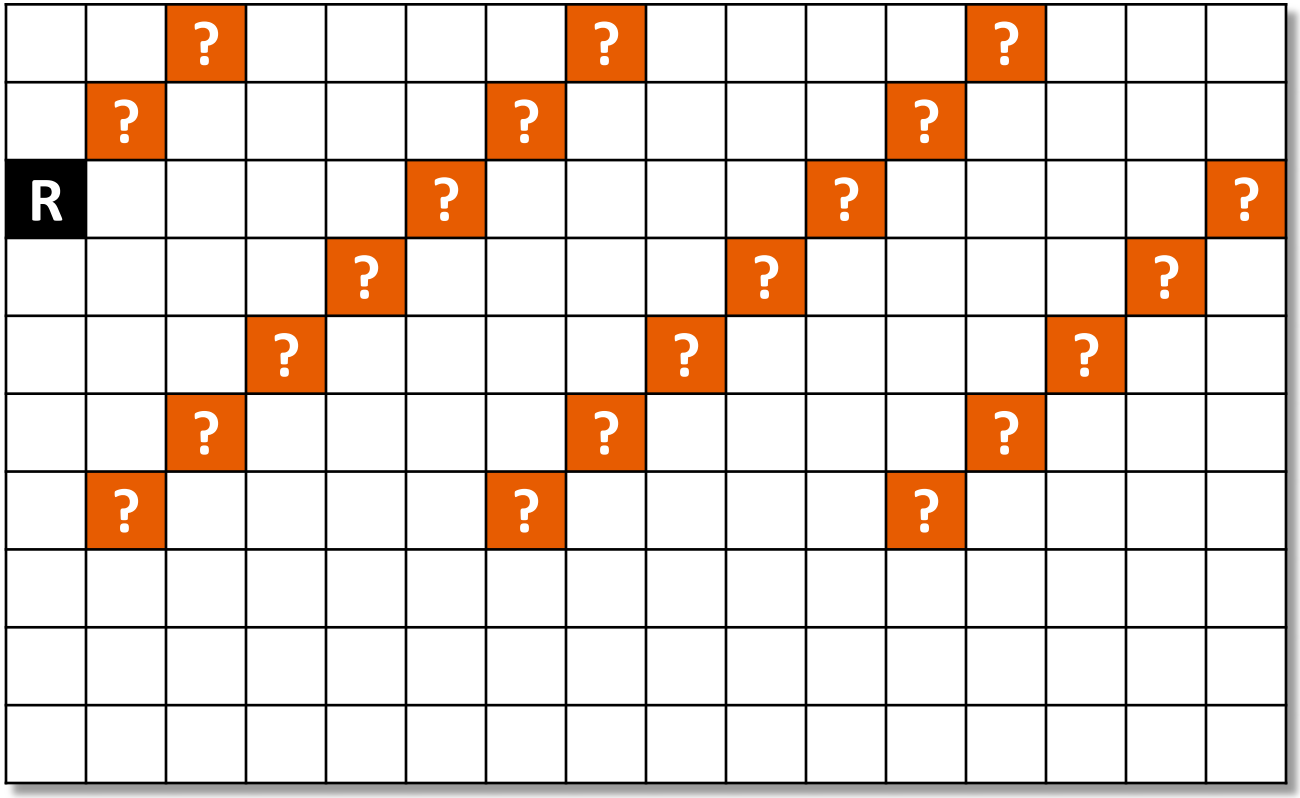
Entropy[1] = 12, 13, 11

Step = 4, result



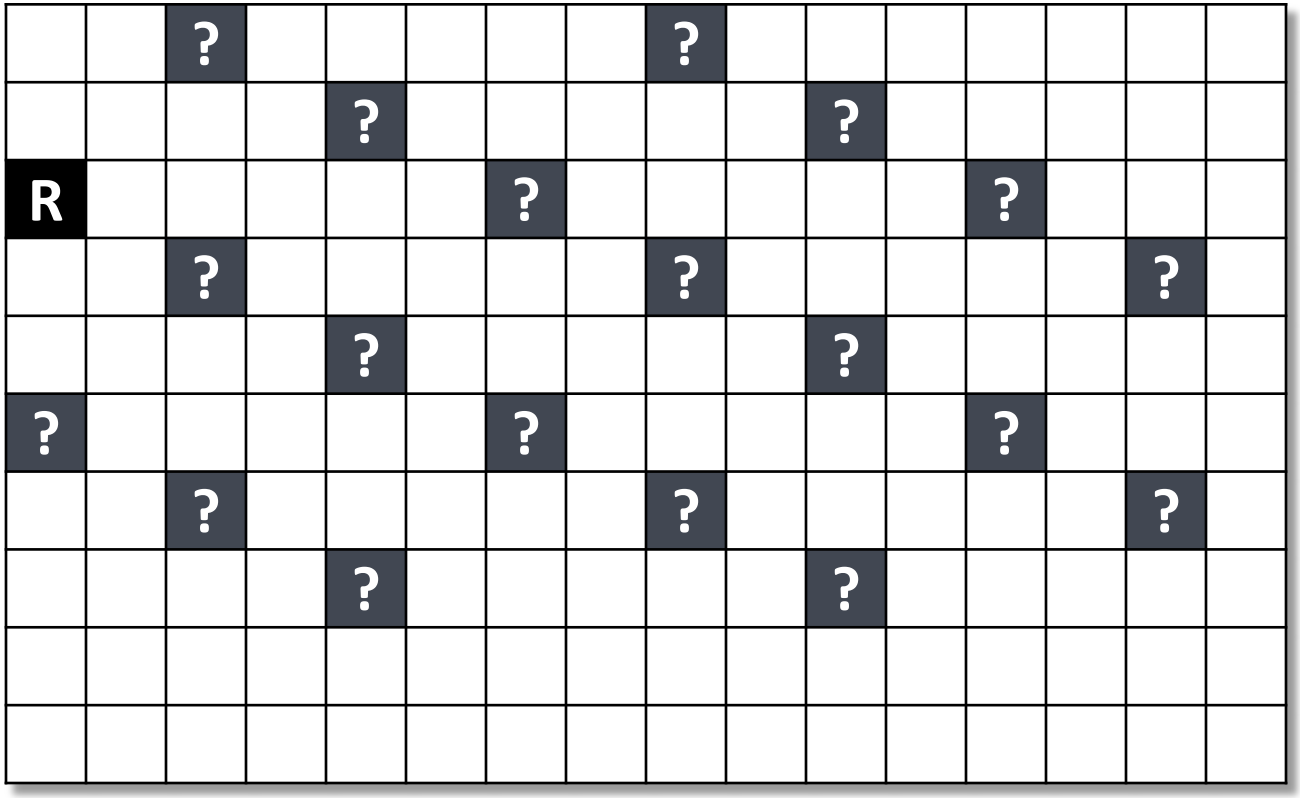
Entropy[1] = 12, 13, 11, 8

Step = 5



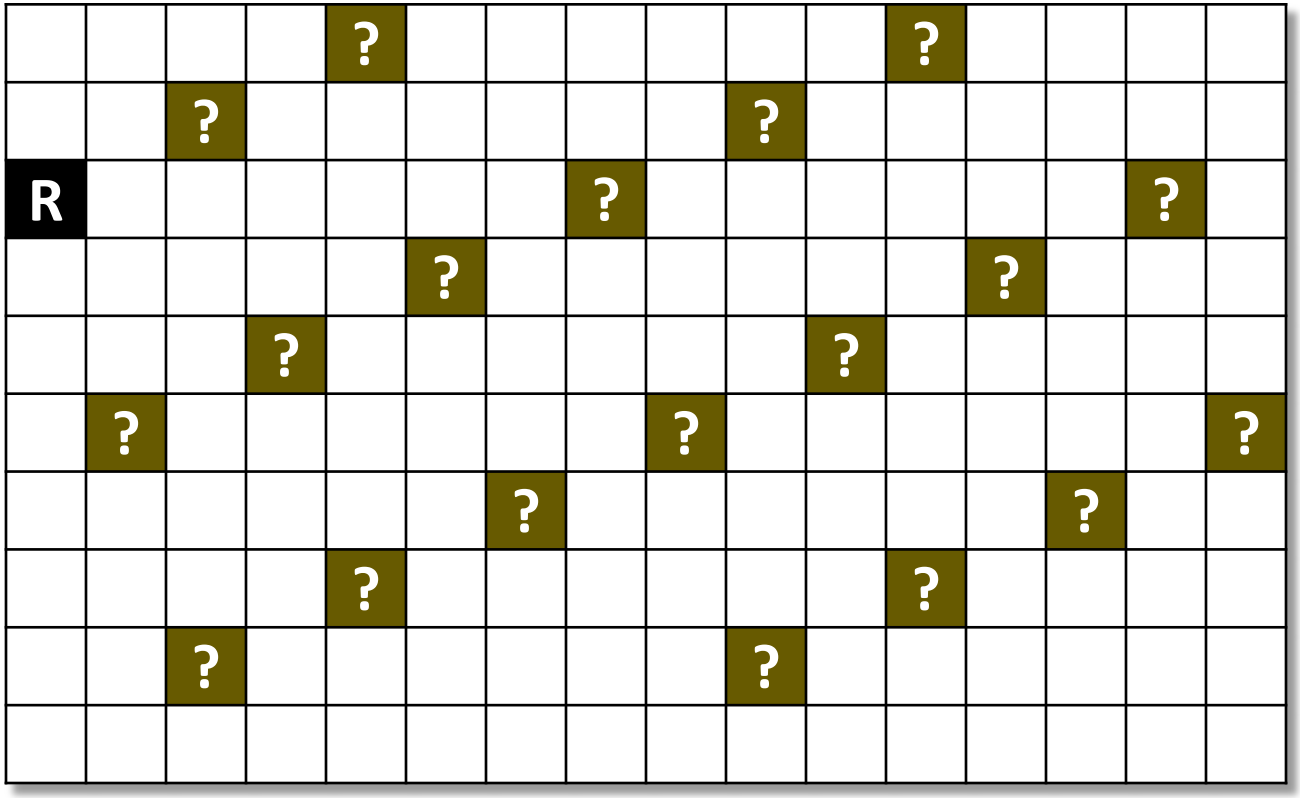
Entropy[1] = 12, 13, 11, 8

Step = 6



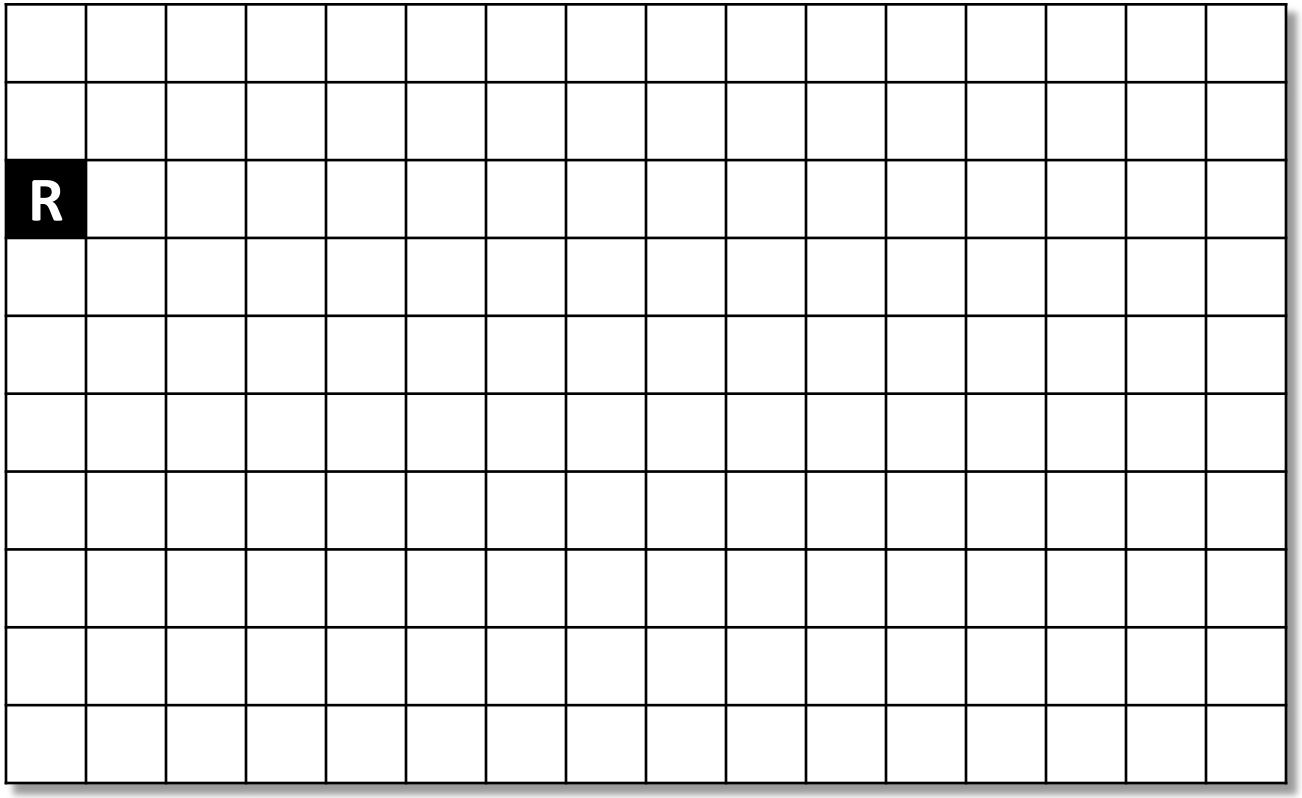
Entropy[1] = 12, 13, 11, 8, 9

Step = 7



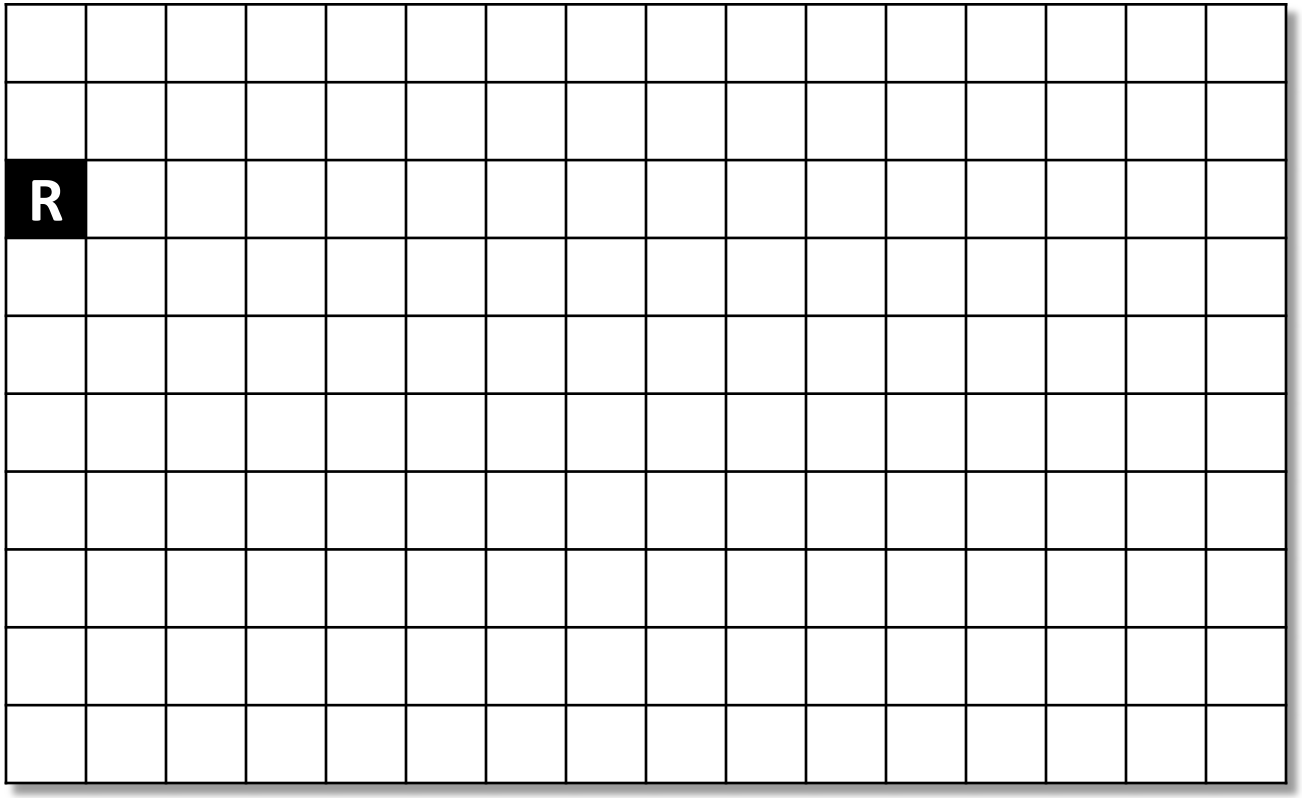
Entropy[1] = 12, 13, 11, 8, 9, 11

Step = 8 .. 48



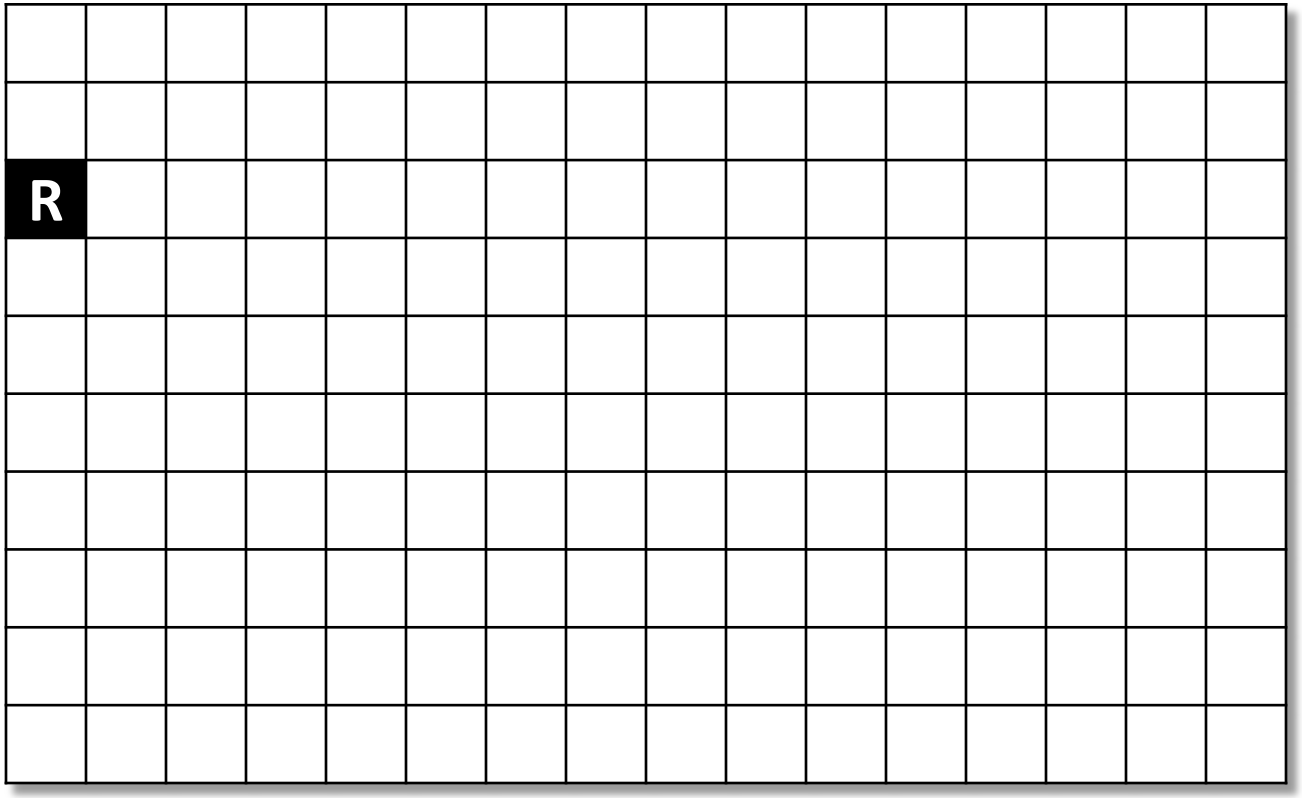
Entropy[1] = 12, 13, 11, 8, 9, 11, …

For all steps…



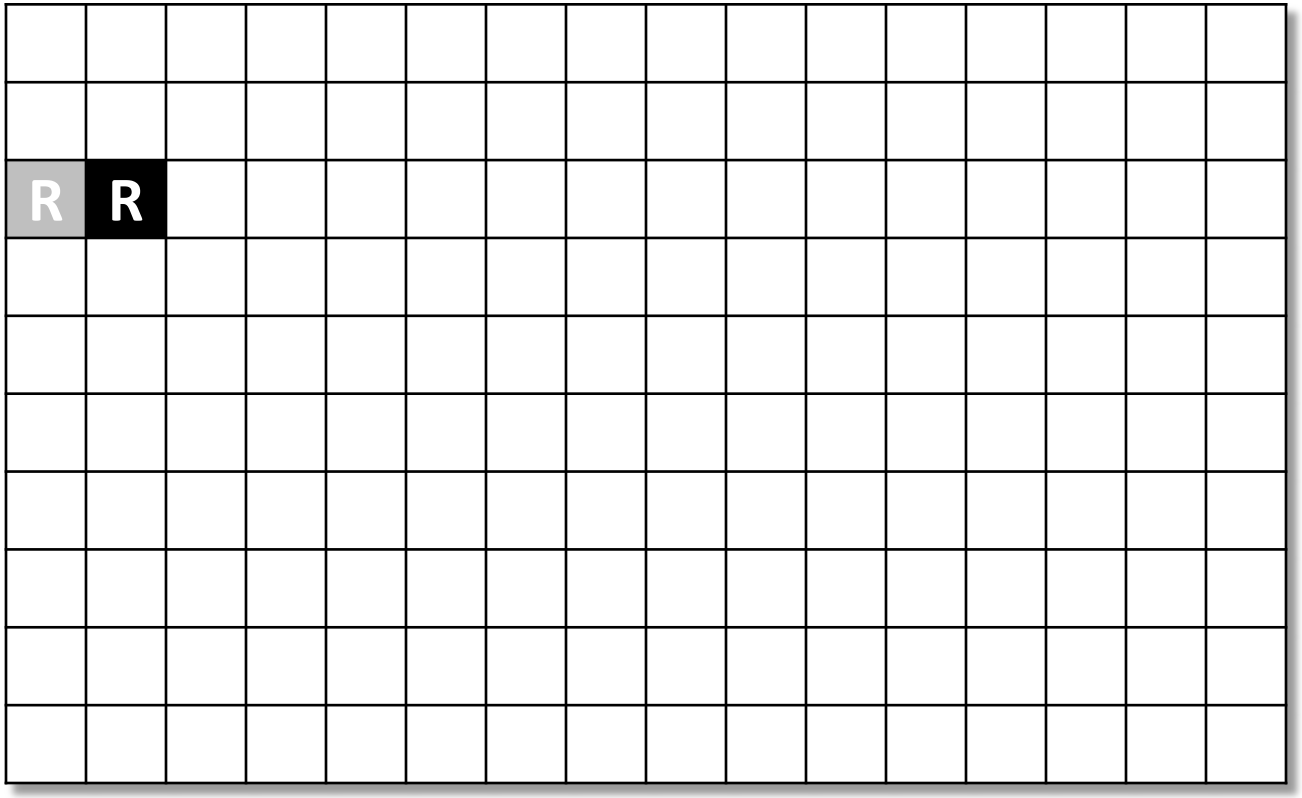
Entropy[1] = min(12, 13, 11, 8, 9, 11, …)

For all steps…



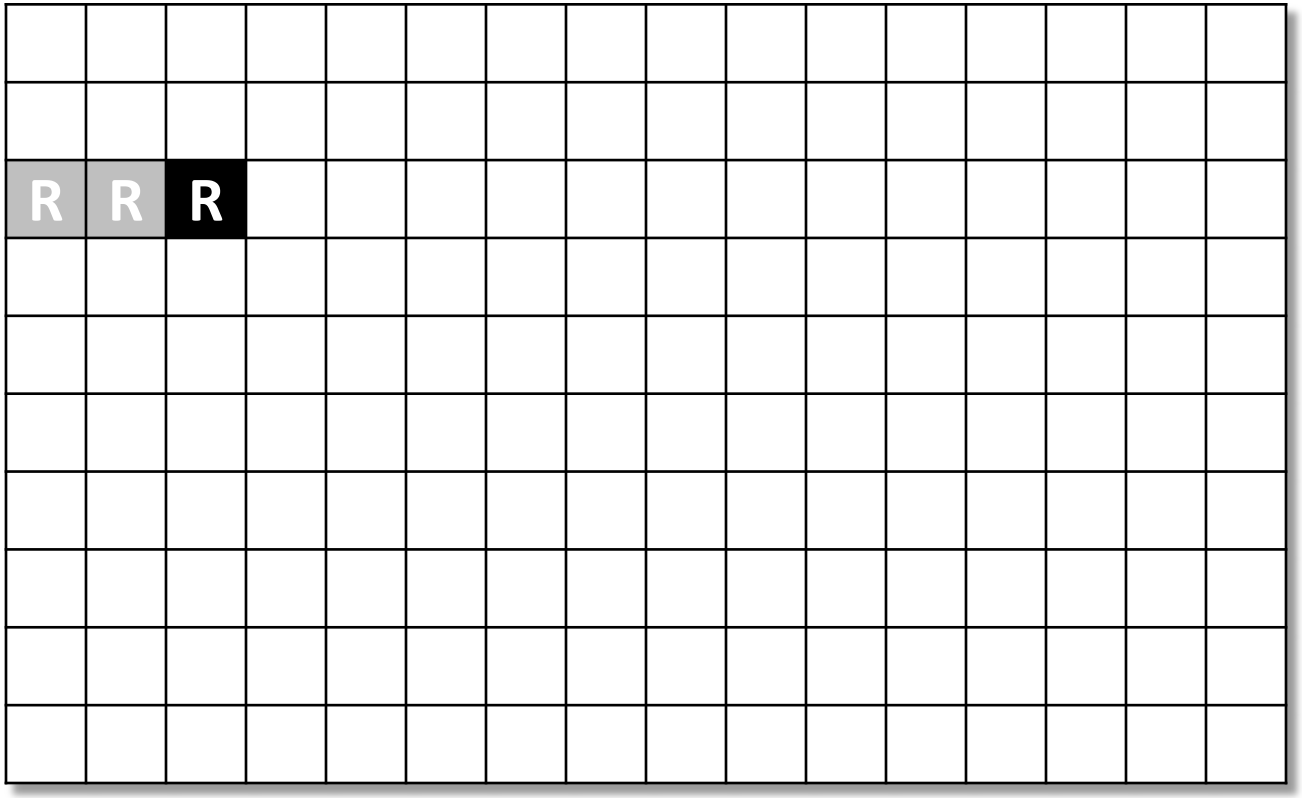
Entropy[1] = 8

Continue: Step = 1 .. 48



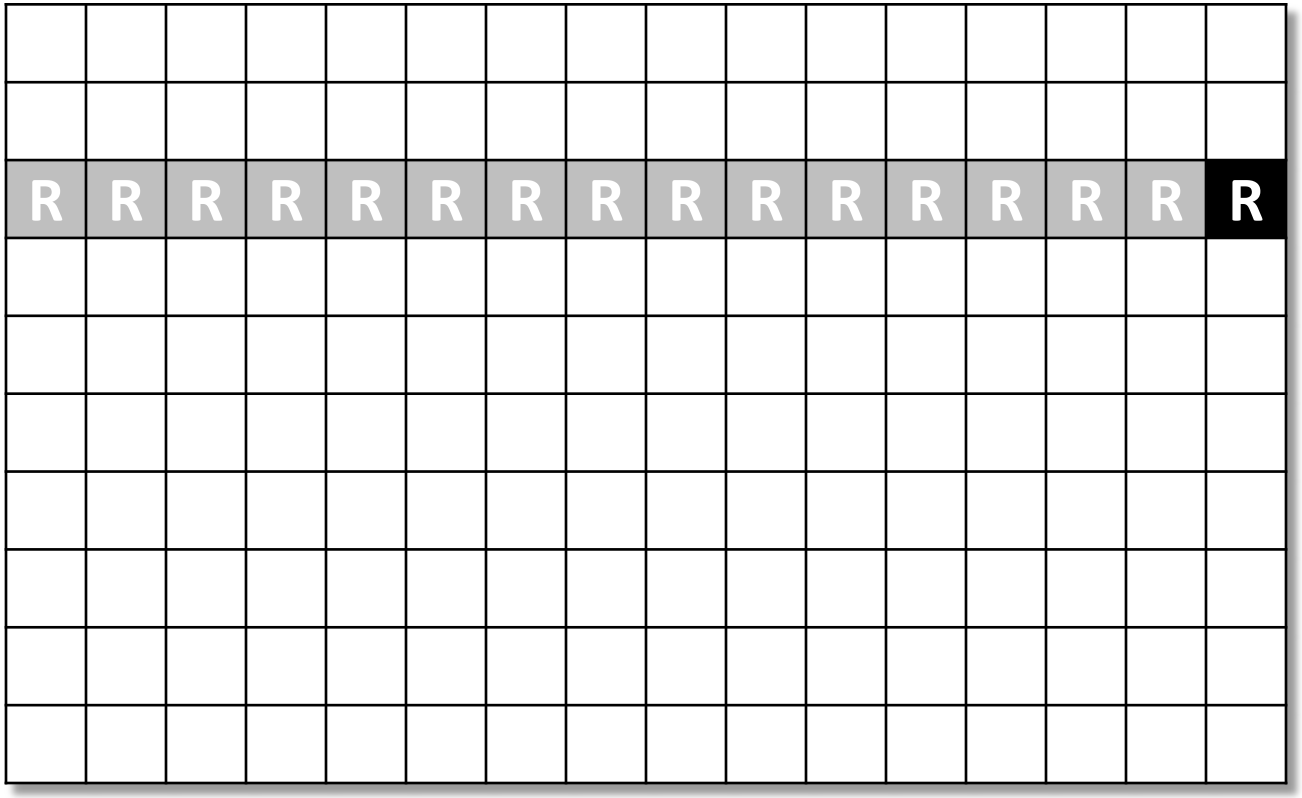
Entropy[2] = ?

And again:



Entropy[3] = ?

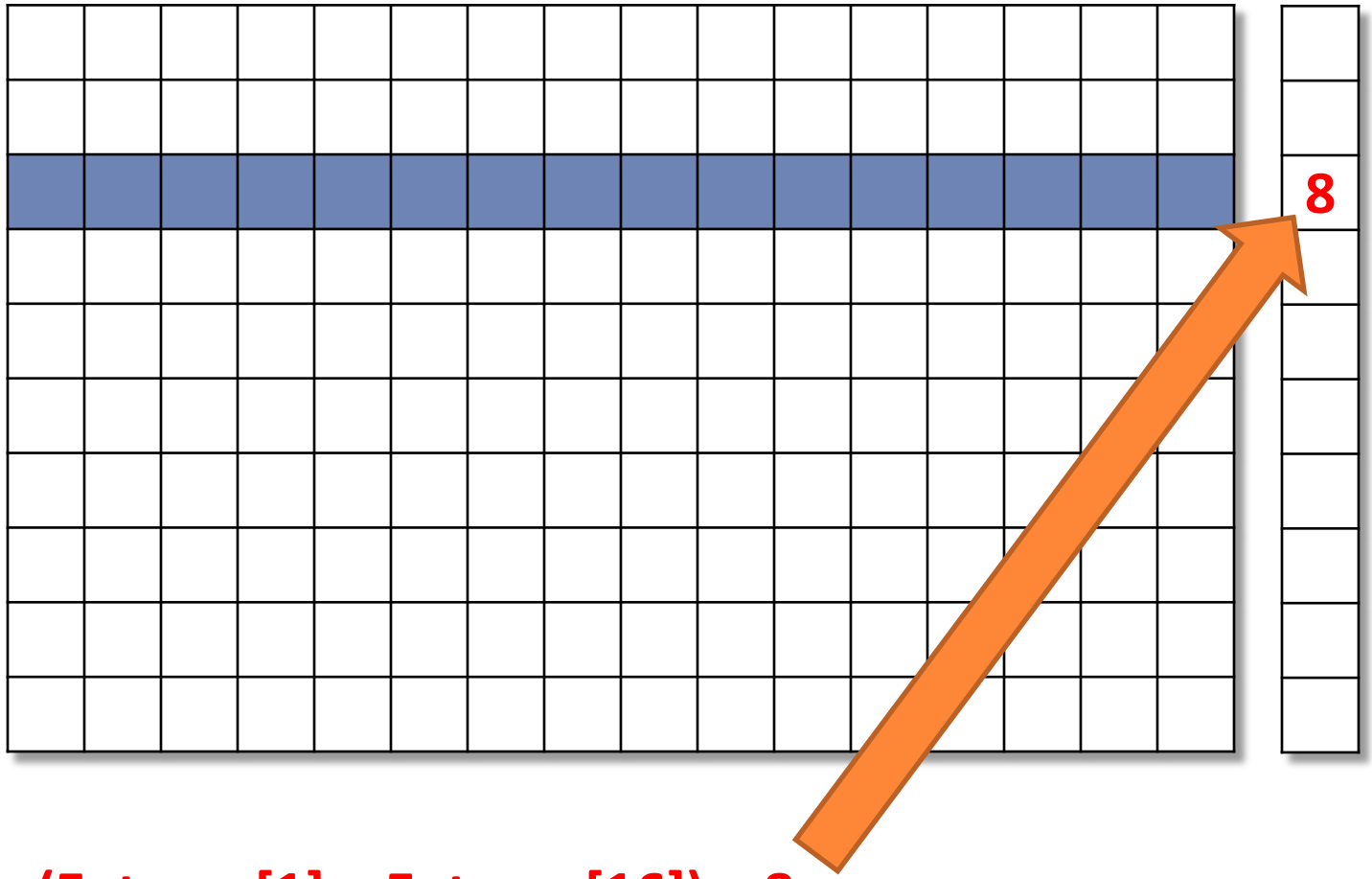
To the end of the row:



Entropy[16] = ?

Result…

…for this row



Entropy = min(Entropy[1] .. Entropy[16]) = 8

And now some
final adjustments:

Many zeroes in data

Many zeroes in data

1. Count the **total number** of zeroes on the line

2. Compute

```
Entropy[zeroes] = 15 – count(zeroes)
```

3. Result is

```
Entropy = min(Entropy[zeroes], Entropy)
```

Many zeroes in data

1. Count the **total number** of zeroes on the line

2. Compute

```
Entropy[zeroes] = 15 - count(zeroes)
```

3. Result is

```
Entropy = min(Entropy[zeroes], Entropy)
```

Many zeroes in data

1. Count the **total number** of zeroes on the line

2. Compute

```
Entropy[zeroes] = 15 - count(zeroes)
```

3. Result is

```
Entropy = min(Entropy[zeroes], Entropy)
```

Text characters in data

Text characters in data

1. Count the longest **continuous text** on the line (including TAB, CR, LF and zero)
2. If the text is shorter than a limit (10 chars), ignore it
3. Otherwise compute

```
Entropy[text] = 15 - length(text)
```

4. Result is

```
Entropy = min(Entropy[text], Entropy)
```

Text characters in data

1. Count the longest **continuous text** on the line (including TAB, CR, LF and zero)

2. If the text is shorter than a limit (10 chars), ignore it

3. Otherwise compute

```
Entropy[text] = 15 - length(text)
```

4. Result is

```
Entropy = min(Entropy[text], Entropy)
```

Text characters in data

1. Count the longest **continuous text** on the line (including TAB, CR, LF and zero)

2. If the text is shorter than a limit (10 chars), ignore it

3. Otherwise compute

```
Entropy[text] = 15 - length(text)
```

4. Result is

```
Entropy = min(Entropy[text], Entropy)
```

Text characters in data

1. Count the longest **continuous text** on the line (including TAB, CR, LF and zero)

2. If the text is shorter than a limit (10 chars), ignore it

3. Otherwise compute

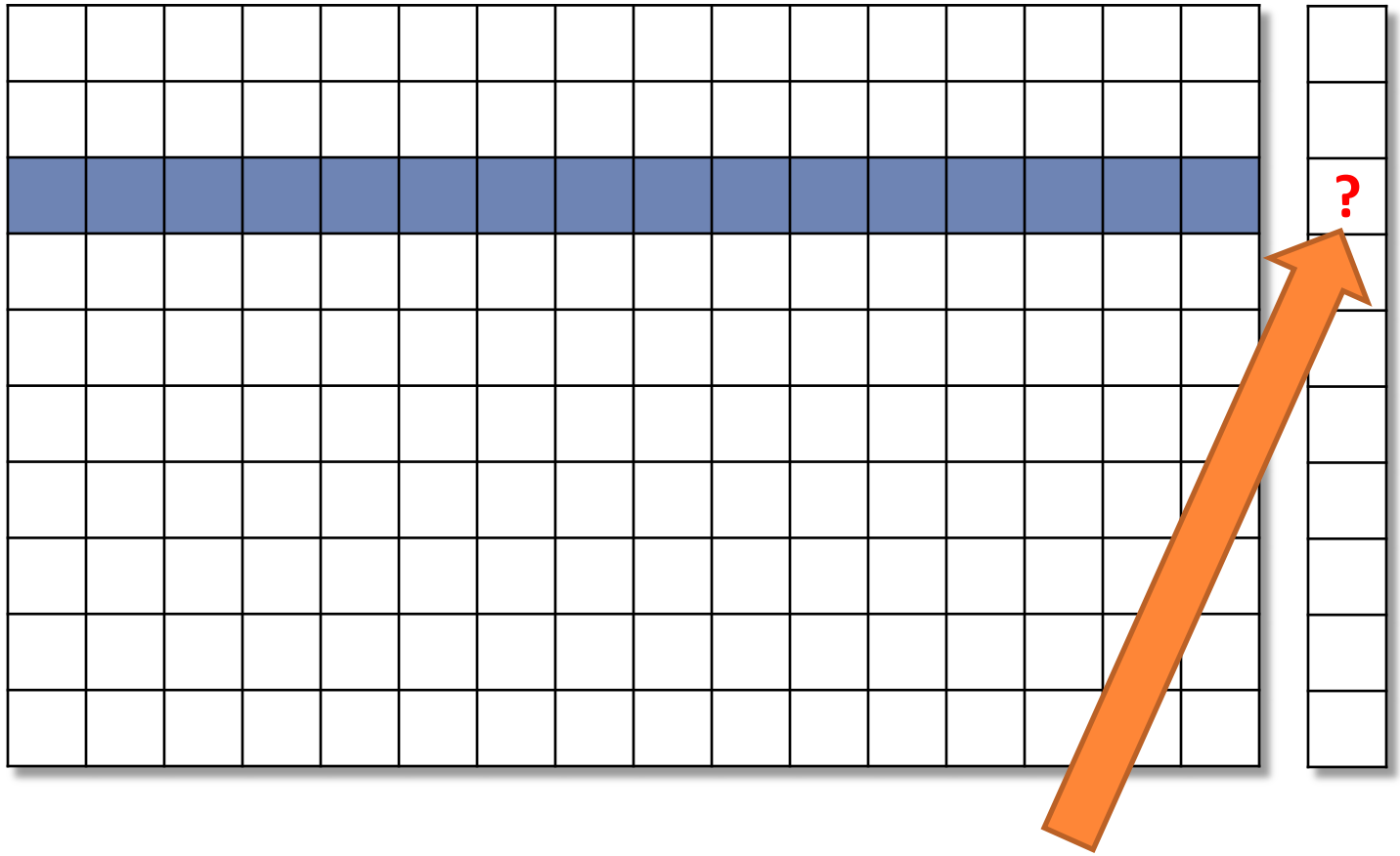
```
Entropy[text] = 15 - length(text)
```

4. Result is

```
Entropy = min(Entropy[text], Entropy)
```

Result…

…for this row

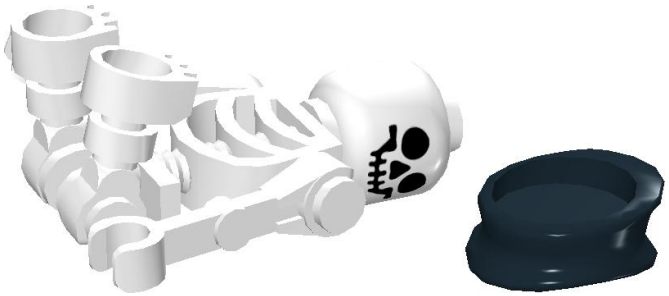


Entropy = min(Entropy[1] .. Entropy[16], Entropy[zeroes], Entropy[text])

That's all, folks…

That's all, folks…

I am looking forward to the discussion!



What will be next?
Have you heard about
a **jump map** ?

Zdeněk Breitenbacher
AVG Technologies,
zdenek.breitenbacher@avg.com