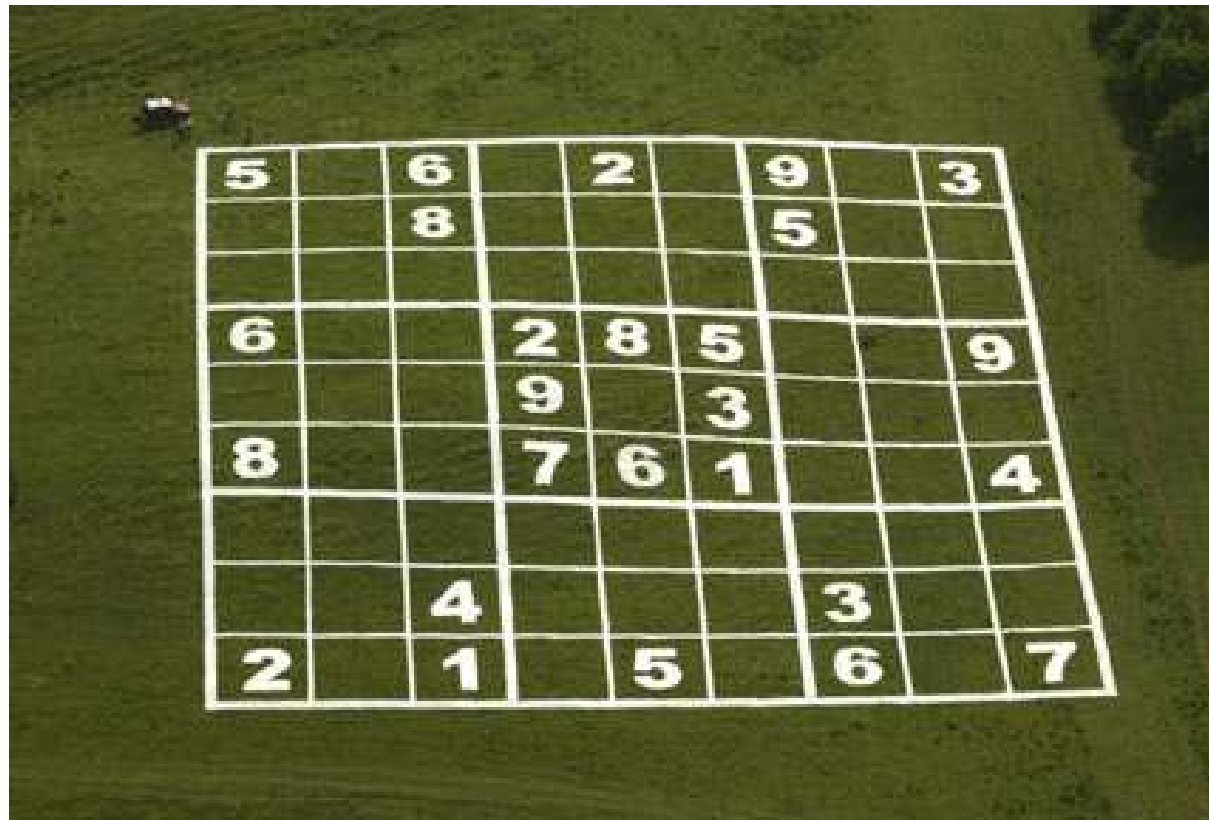
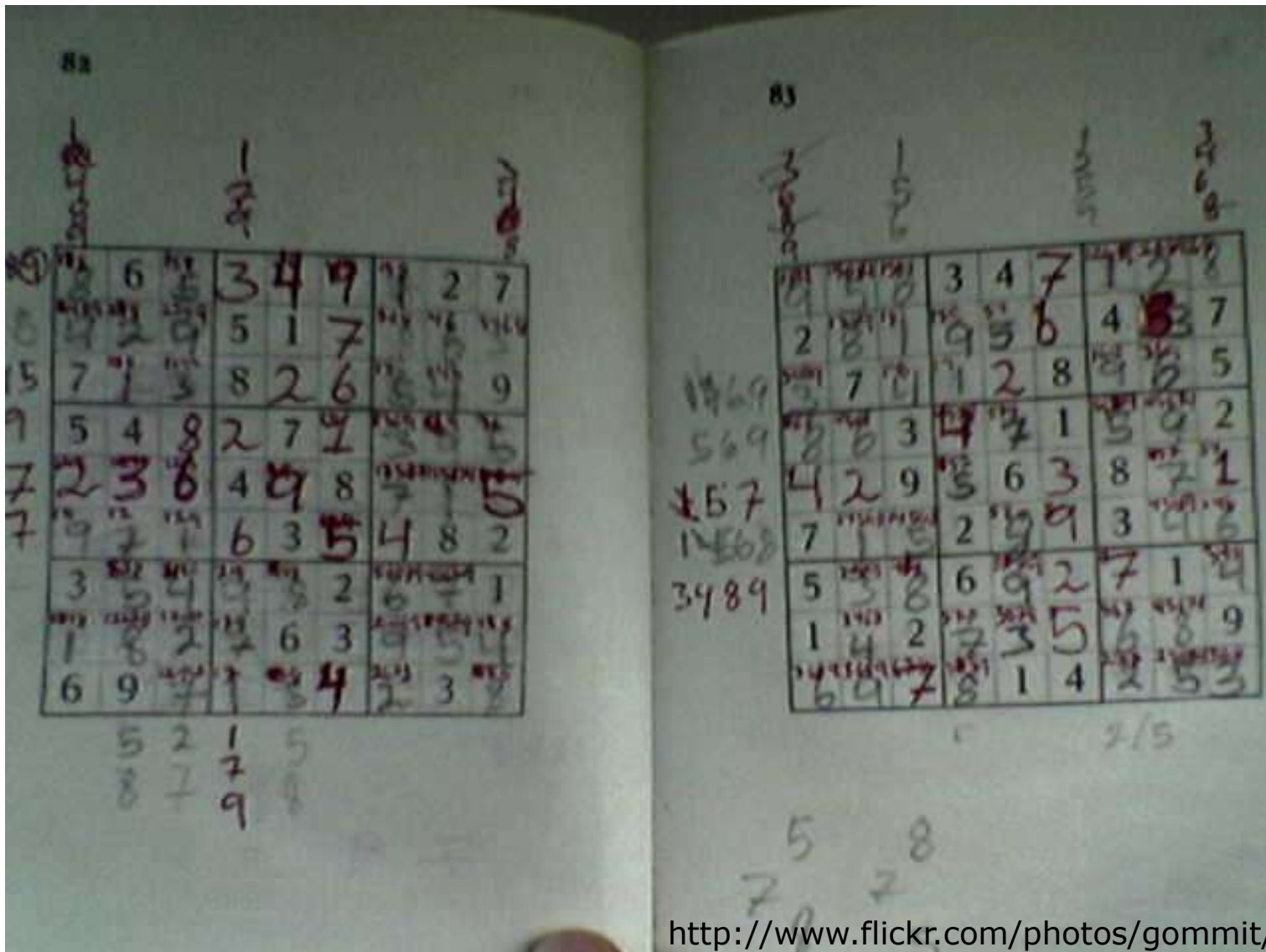


Extreme Sudoku solving with Ruby



Matt Westcott

Why a sudoku solver?



The general plan

8		6		
		1 2 3 4 5 6 7 8 9	7	3
7			3	
	2			

First pass: Eliminating candidates

	2	3	8	1		7	4	
7					3	1		
	9				2	8		5
		9		4			8	7
4			2		8			3
1	6			3		2		
3		2	7				6	
		5	6					8
	7	6		5	1		9	

Second pass: Finding unique positions

		9	1					3
8				6		5		
		5			7	4		
5	3	4						
			8		6			
						7	3	9
		1	7			3		
		8	6	2				5
2					4	6		

Doing it in Ruby

```
solved_square.value = 6
grid.each_in_row(solved_square.row) do |square|
  square.eliminate(6) unless square == solved_square
end
```

```
grid.each_in_column(solved_square.column) do |square|
  square.eliminate(6) unless square == solved_square
end
```

Iterators

```
Class Grid
```

```
  def each_in_row(y)
    0.upto(9) do |x|
      yield @squares[y][x]
    end
  end
```

```
end
```

```
  def each_in_square(n)
    (n/3 .. n/3+2).each do |y|
      (n%3 .. n%3+2).each do |x|
        yield @squares[y][x]
      end
    end
  end
```

```
end
```

```
end
```

```
end
```

Lookahead

1					8	3		
	8			7	2	6	1	
			1	3	9	4		
				9		7		
7								2
		9		5	7			
		7		8				
	4	8	9	2			3	7
		2	7					4

Lookahead

```
def solve(grid)
  whittle_down!(grid) # do as much number-crossing-out
    as possible
  if grid.screwed_up?
    raise UnsolvableException
  elsif grid.solved?
    return grid
  else
    empty_square = grid.find{|square| square.empty?}
    empty_square.candidates.each { |candidate|
      backup_grid = grid.dup
      empty_square.fill_with(candidate)
      begin
```

Lookahead

```
else
  empty_square = grid.find{|square| square.empty?}
  empty_square.candidates.each { |candidate|
    backup_grid = grid.dup
    empty_square.fill_with(candidate)
    begin
      solution = solve(grid)
      return solution
    rescue UnsolvableException
      grid = backup_grid
    end
  }
  raise UnsolvableException
end
end
end
```

http://www.timesonline.co.uk/article/0,,182

No 864

RATING: FIENDISH

	6		1			8		7
				5				
3			9					
	4							
	8		5					
			3					
5		2						

KILLER No 259 Tim

18			12					
	20		13					
						21		
13								
8			20					
4		7				19		
				12				
23								

Classic Su Doku © Puzz

Fill the grid so that every 3x3 box con

Play interactive Su

Su

Send Chat Attach Address Fonts

To:

Cc:

Subject:

Bcc:

Account:

Solution:

9	6	5	1	4	2	8	3	7
4	1	8	7	5	3	6	9	2
3	2	7	9	6	8	1	5	4
7	4	1	6	2	9	3	8	5
6	8	9	5	3	4	7	2	1
2	5	3	8	7	1	9	4	6
8	3	6	4	1	5	2	7	9
1	9	4	2	8	7	5	6	3
5	7	2	3	9	6	4	1	8

```

Last login: Sat Jul 22 23:10:36 on ttty2
Welcome to Darwin!
woodstock:~ matthew$ cd Development/ruby/sudoku/
woodstock:~/Development/ruby/sudoku matthew$ ruby sudoku.rb
9 6 5 1 4 2 8 3 7
4 1 8 7 5 3 6 9 2
3 2 7 9 6 8 1 5 4
7 4 1 2 6 2 9 3 8 5
6 8 9 5 3 4 7 2 1
2 5 3 8 7 1 9 4 6
8 3 6 4 1 5 2 7 9
1 9 4 2 8 7 5 6 3
5 7 2 3 9 6 4 1 8

woodstock:~/Development/ruby/sudoku matthew$

```

Screen scraping

```
require 'net/http'
# step 1: fetch redirector page
h = Net::HTTP.new('www.timesonline.co.uk', 80)
response = h.get('/sudoku')
find_meta = response.body.scan(/<META HTTP-
    EQUIV=Refresh CONTENT='\s10;
    URL="http://\swww\s.timesonline\s.co\s.uk([\s"]+
    \s">/)
homepage_url = find_meta[0][0]

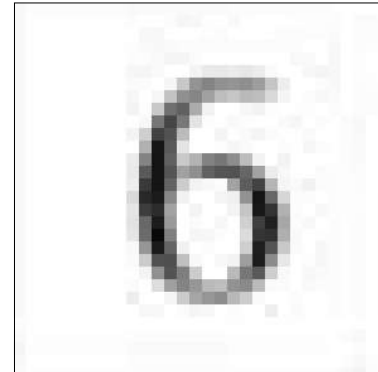
# step 2: look for puzzle link on homepage
response = h.get(homepage_url)
link_text = today.strftime("Su Doku: %B %d, %Y");
find_puzzle_url = response.body.scan(/<a href='\s(
    [\s"]+)\s' class='\ssectionhead\s'>#{link_text}
    </a>/)
puzzle_url = find_puzzle_url[0][0]
```

JPEG artifacts 'Я' us



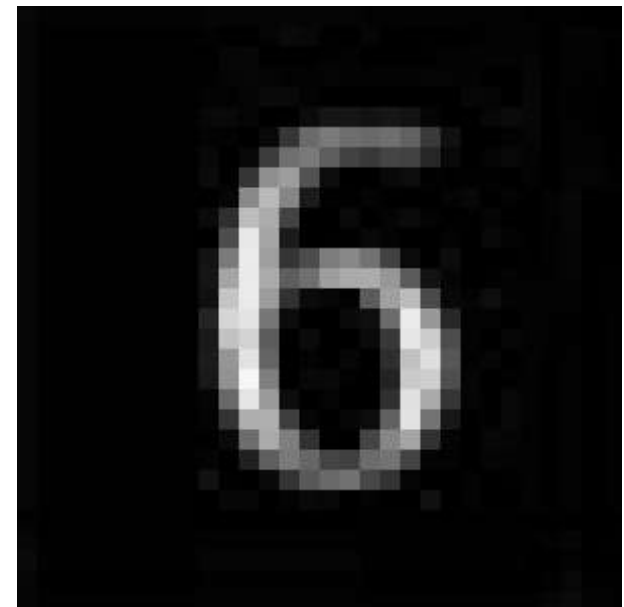
OCR processing

```
require 'RMagick'  
include Magick  
img = img.negate  
img.resize!  
  (100,100,LanczosFilter, 4)  
mass = 0; com_x = 0; com_y = 0  
(0...100).each do |y|  
  (0...100).each do |x|  
    pix = img.pixel_color(x,y).  
      intensity  
    mass += pix  
    com_x += x * pix  
    com_y += y * pix
```



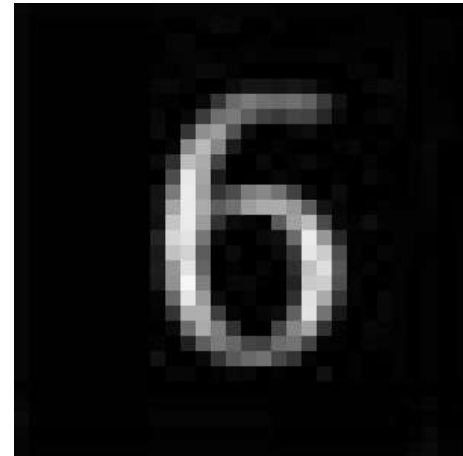
OCR processing

```
require 'RMagick'  
include Magick  
img = img.negate  
img.resize!  
  (100,100,LanczosFilter, 4)  
mass = 0; com_x = 0; com_y = 0  
(0...100).each do |y|  
  (0...100).each do |x|  
    pix = img.pixel_color(x,y).  
      intensity  
    mass += pix  
    com_x += x * pix  
    com_y += y * pix
```



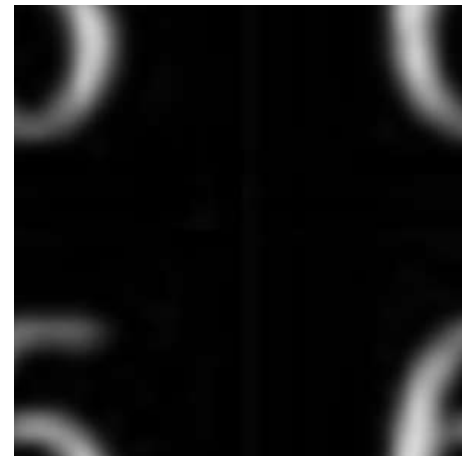
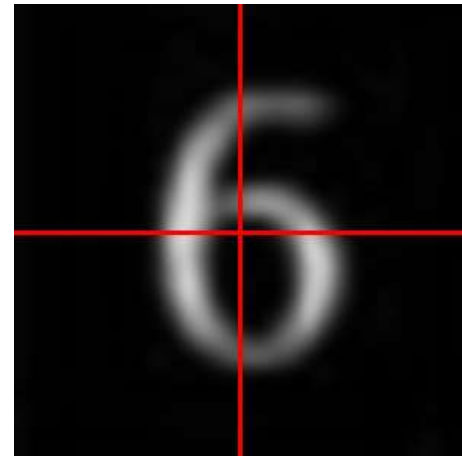
OCR processing

```
require 'RMagick'  
include Magick  
img = img.negate  
img.resize!  
  (100,100,LanczosFilter, 4)  
mass = 0; com_x = 0; com_y = 0  
(0...100).each do |y|  
  (0...100).each do |x|  
    pix = img.pixel_color(x,y).  
      intensity  
    mass += pix  
    com_x += x * pix  
    com_y += y * pix
```



OCR processing

```
mass = 0; com_x = 0; com_y = 0
(0...100).each do |y|
  (0...100).each do |x|
    pix = img.pixel_color(x,y).
      intensity
    mass += pix
    com_x += x * pix
    com_y += y * pix
  end
end
com_x /= mass
com_y /= mass
img = img.roll(-com_x, -com_y)
```



OCR processing

```
def difference(img1, img2)
  total_diff = 0;
  (0...100).each do |y|
    (0...100).each do |x|
      total_diff += (
        img1.pixel_color(x,y).intensity -
        img2.pixel_color(x,y).intensity).abs
    end
  end
  total_diff
end
```



Success...



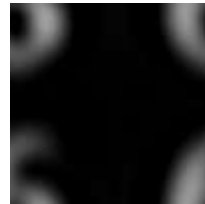
Digit: 0
Error: 58575



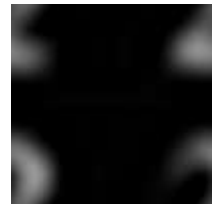
Digit: 5
Error: 28413



Digit: 1
Error: 43923



Digit: 6
Error: 2397 (92% certainty)



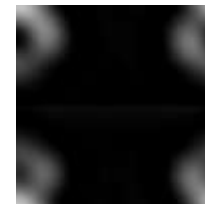
Digit: 2
Error: 42246



Digit: 7
Error: 37606



Digit: 3
Error: 33617



Digit: 8
Error: 32810



Digit: 4
Error: 39996



Digit: 9
Error: 38262



<http://www.flickr.com/photos/ingopics/>