CO-PRESENTED BY O'REILLY Ruby Central

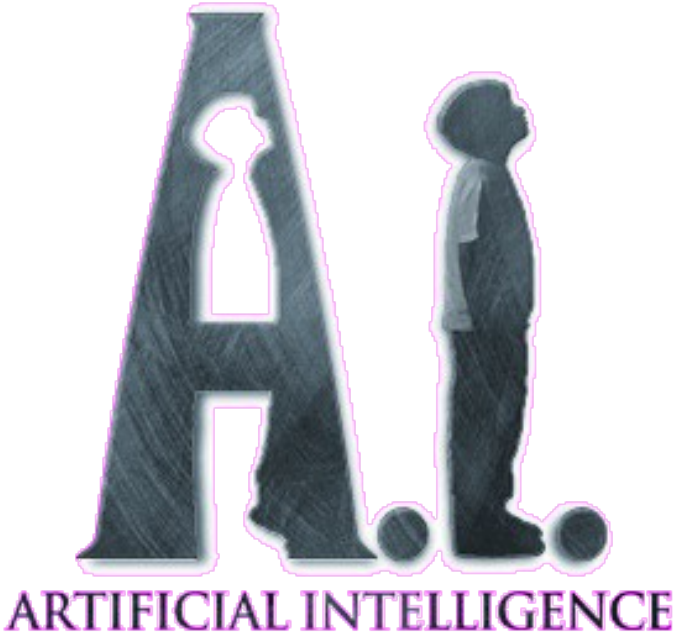
Santiago Bel
Sergio Espeja

Making Rails More (Artificially) Intelligent

Partially supported by **Ministerio de Educación y Ciencia** and the **Fondo Social Europeo** PTA-CTE/1370/2003

Artificial Intelligence

- "*The science and engineering of making intelligent machines*", John McCarthy
- Algorithms that solve ”intelligent” problems



What are we going to see?

- Bayesian Networks
  - Predicts if attendants will fall asleep
- Naive Bayes Classifier
  - Email classification for customer services.
- Genetic Algorithms
  - Optimize web page revenue from advertisements

Bayesian Networks

- Causal networks
  - Cause → Effect
- 2 components:
  - Acyclic graph
  - Probability distributions set
- Inference

Ruby and Bayesian Networks

- sbn, Simple Bayesian Networks
  - http://sbn.rubyforge.org/
  - Carl Youngblood, 2007
- bn4r, Bayesian Networks for Ruby
  - http://bn4r.rubyforge.org/
  - Sergio Espeja, 2006

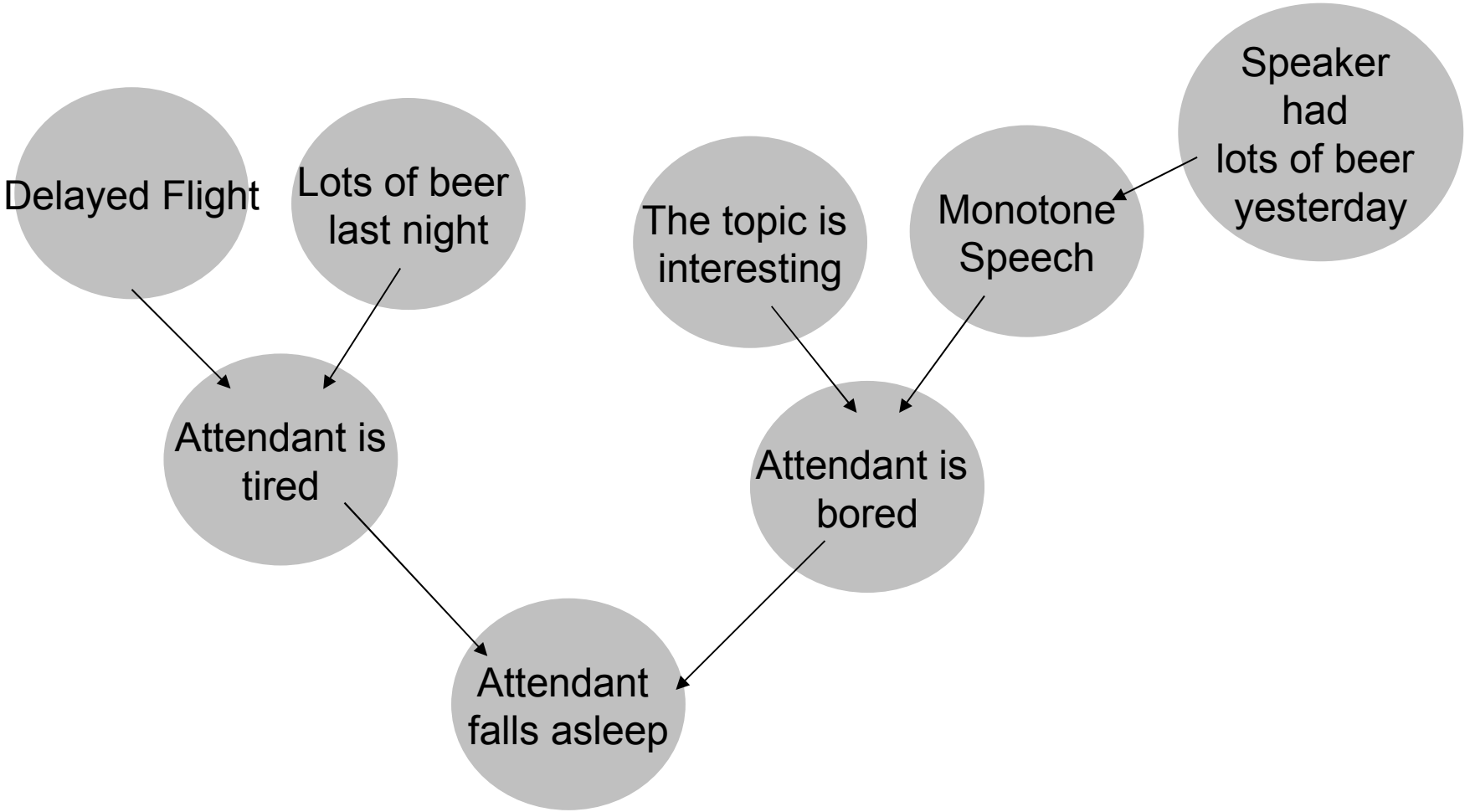
- Install the library

```
gem install bn4r
```

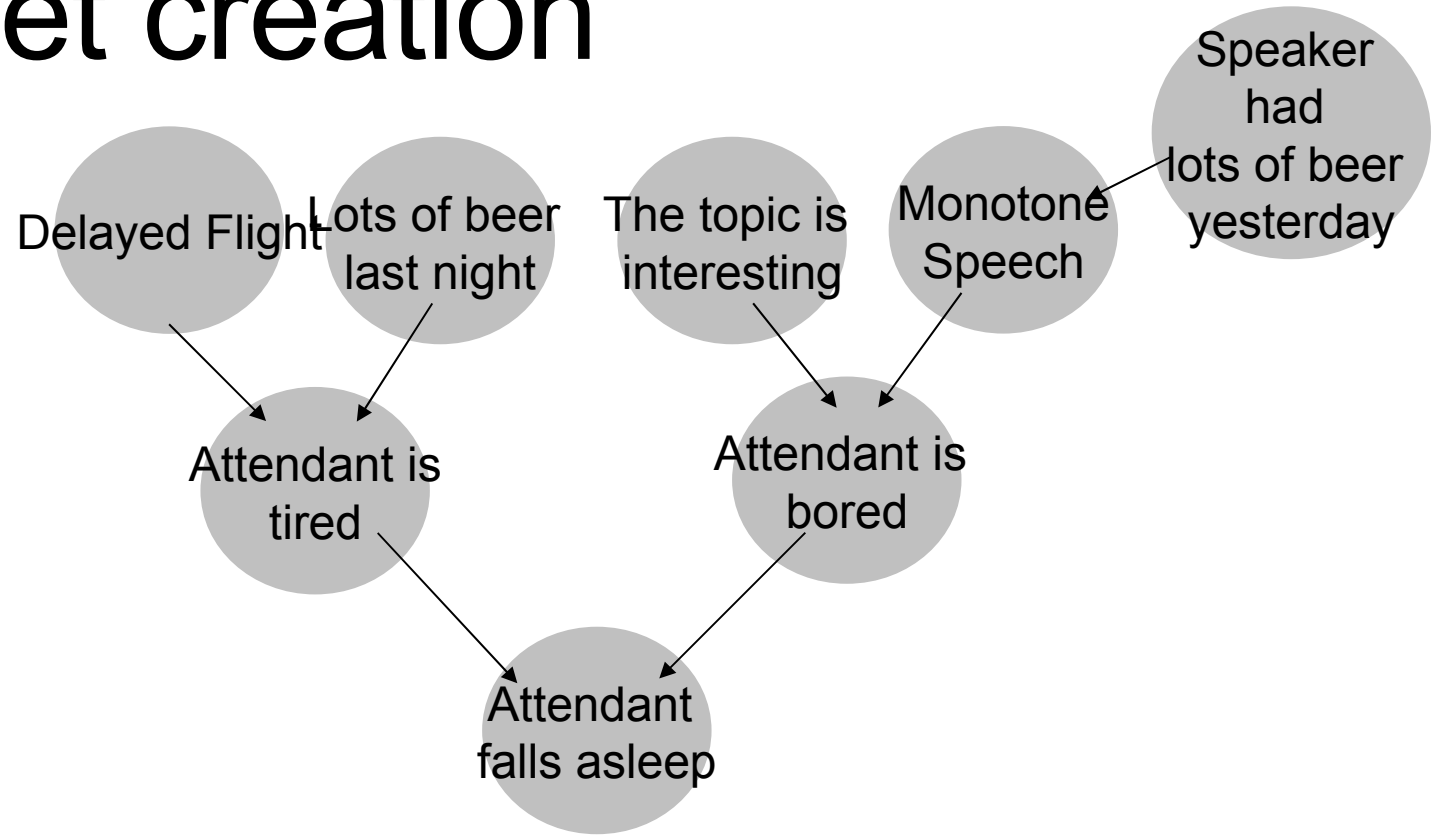
Bayes predictor: Are you going to fall asleep?

Photo by **Jarkko Laine**
http://www.flickr.com/people/jarkko/

Bayes predictor: Are you going to fall asleep?



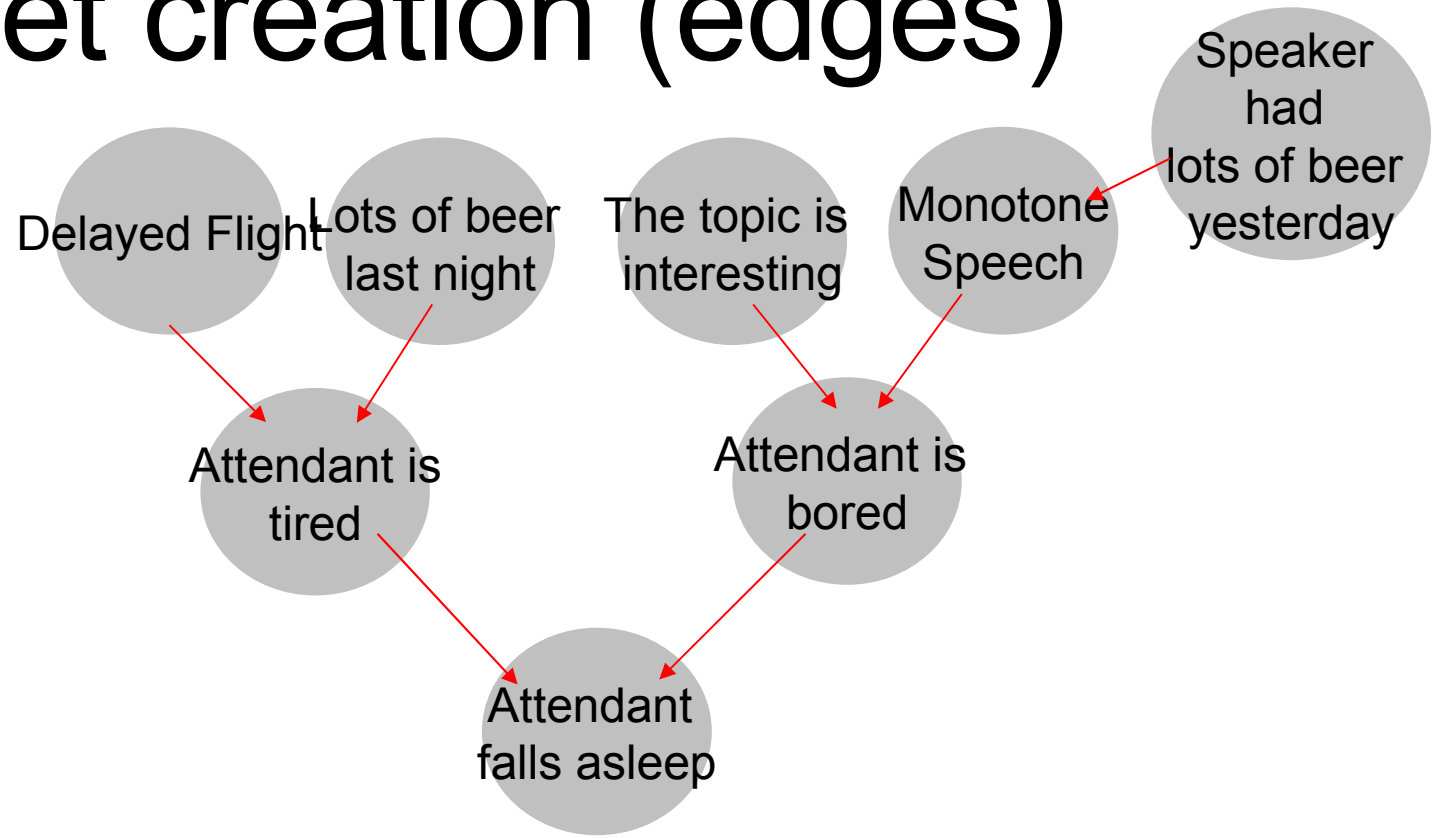
Bayes Net creation




```
# Create BayesNet
bn = BayesNet.new
```

```
# Create nodes for the Bayes Net (BayesNetNodes)
bn.add_vertex( delayed = BayesNetNode.new("DelayedFlight") )
bn.add_vertex( att_hangover = BayesNetNode.new("LotsOfBeerLastNight") )
…
bn.add_vertex( asleep = BayesNetNode.new("AttendantFallsAsleep") )
```

Bayes Net creation (edges)



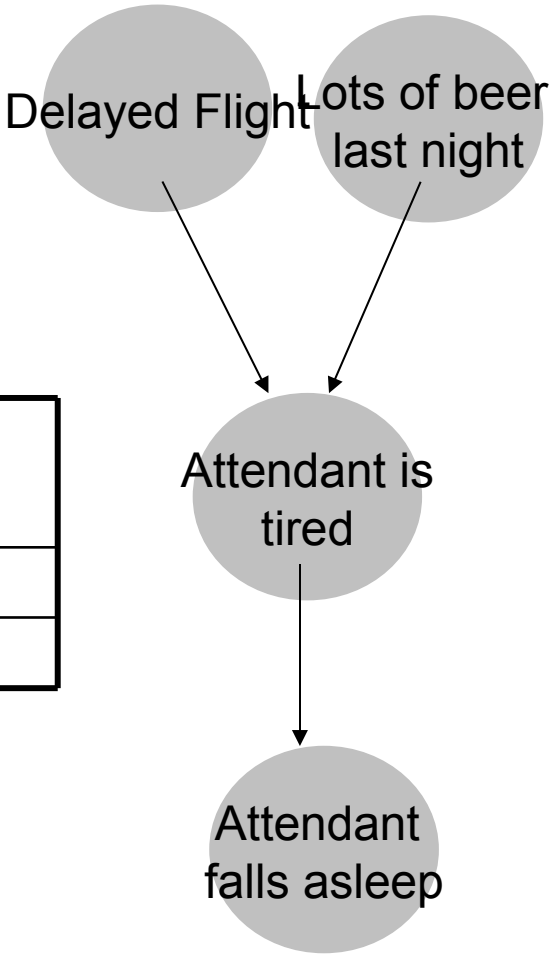

```
# Add relations ( edges ) between nodes in the BayesNet
bn.add_edge( delayed, tired )
bn.add_edge(att_hangover, tired )
…
bn.add_edge( tired, asleep )
bn.add_edge( bored, asleep )
```

Probability Distribution Tables

P(delay=true)	0.3
P(delay=false)	0.7

P(att_hangover=true)	0.2
P(att_hangover=false)	0.8

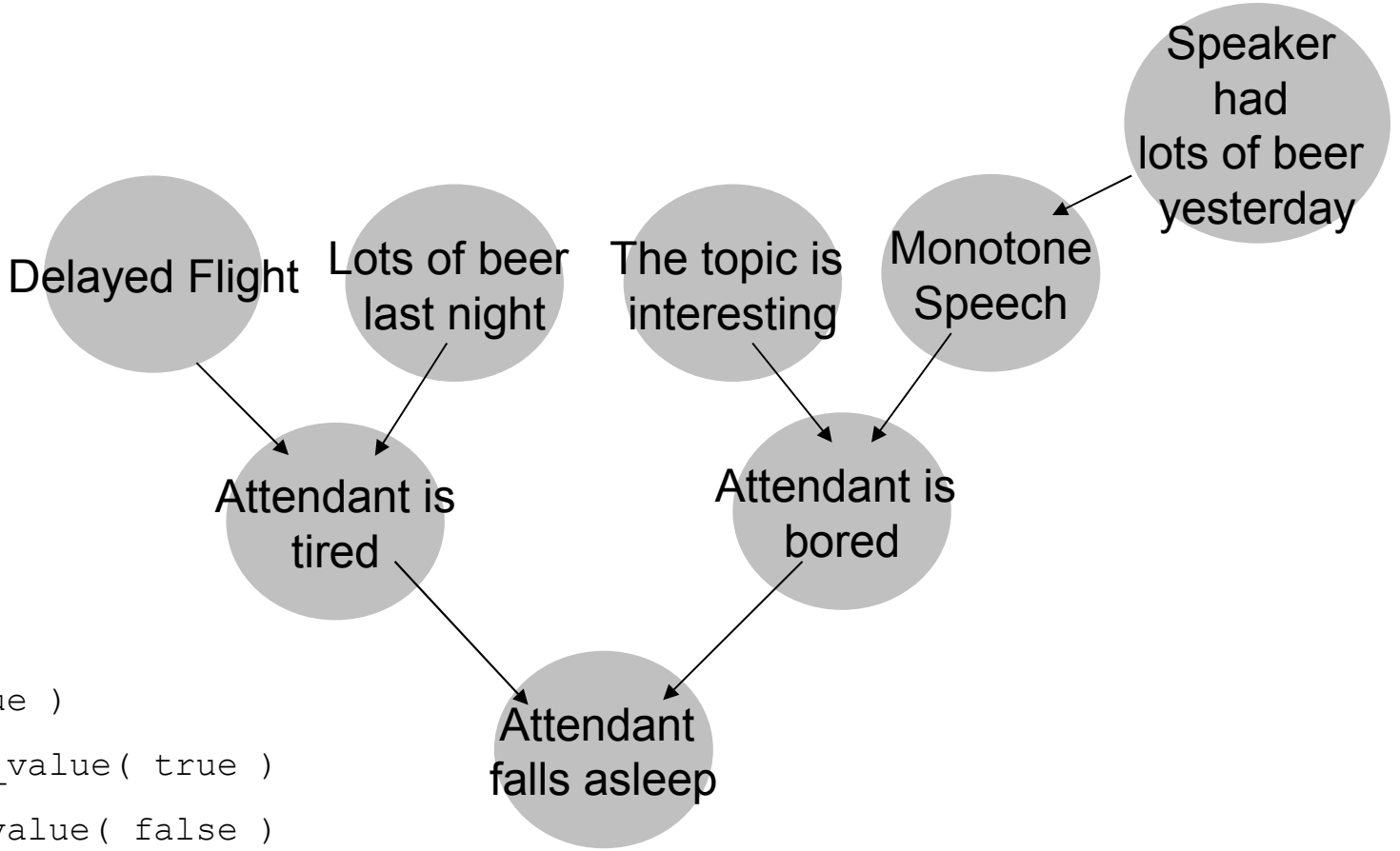
	delay & att_hangover	delay & !att_hangover	!delay & att_hangover	!delay & !att_hangover
P(tired=true)	0.95	0.7	0.8	0.05
P(tired=false)	0.05	0.3	0.2	0.95




```
# Assign probabilities to each node
delayed.set_probability_table( [], [0.3, 0.7] )
att_hangover.set_probability_table( [], [0.2, 0.8] )

tired.set_probability_table( [delay, att_hangover], [0.95,
0.05, 0.7, 0.3, 0.8, 0.2, 0.05,0.95] )
```

Solve




```
delayed.set_value( true )
interesting_topic.set_value( true )
speaker_hangover.set_value( false )
is_attendant_sleeping = bn.enumeration_ask ( asleep, [delayed, interesting_topic,
                                                      speaker_hangover] )
```

P(is_attendant_sleeping=true| delayed, interesting_topic, speaker_hangover) = 0.1
P(is_attendant_sleeping=false| delayed, interesting_topic, speaker_hangover) = 0.9

Hey! You are breaking the probabilities!

Photo by **Jarkko Laine**
http://www.flickr.com/people/jarkko/

```
require "rubygems"
require "bn4r"

# Create BayesNet
bn = BayesNet.new

# Create nodes for the Bayes Net (BayesNetNodes)
bn.add_vertex( delayed = BayesNetNode.new("DelayedFlight") )
bn.add_vertex( att_hangover = BayesNetNode.new("LotsOfBeerLastNight") )
bn.add_vertex( tired = BayesNetNode.new("AttendantIsTired") )
bn.add_vertex( speaker_hangover = BayesNetNode.new("SpeakerHadLotsOfBeerYesterday") )
bn.add_vertex( monotone_speech = BayesNetNode.new("MonotoneSpeech") )
bn.add_vertex( interesting_topic = BayesNetNode.new("TheTopicIsInteresting") )
bn.add_vertex( bored = BayesNetNode.new("AttendantIsBored") )
bn.add_vertex( asleep = BayesNetNode.new("AttendantFallsAsleep") )

# Add relations ( edges ) between nodes in the BayesNet
bn.add_edge( delayed, tired )
bn.add_edge(att_hangover, tired )
bn.add_edge( tired, asleep )
bn.add_edge( speaker_hangover, monotone_speech )
bn.add_edge( monotone_speech , bored )
bn.add_edge( interesting_topic, bored )
bn.add_edge( bored, asleep )

# Assign probabilities to each node
delayed.set_probability_table( [], [0.3, 0.7] )
att_hangover.set_probability_table( [], [0.2, 0.8] )
tired.set_probability_table( [delayed, att_hangover], [0.95, 0.05, 0.7, 0.3, 0.8, 0.2, 0.05,0.95] )
speaker_hangover.set_probability_table( [], [0.5, 0.5] )
monotone_speech.set_probability_table( [speaker_hangover], [0.9, 0.1, 0.1, 0.9] )
interesting_topic.set_probability_table( [], [0.95, 0.05] )
bored.set_probability_table( [monotone_speech, interesting_topic], [0.6, 0.4, 0.95, 0.05, 0.05,
0.95, 0.8, 0.2] )
asleep.set_probability_table( [tired, bored], [0.9, 0.1, 0.3, 0.7, 0.4, 0.6, 0.05,0.95] )
```

```
# Assign observed values
delayed.set_value( true )
interesting_topic.set_value( true )
speaker_hangover.set_value( false )

# Ask the network
is_attendant_sleeping = bn.enumeration_ask ( monotone_speech, [ delayed, interesting_topic,
speaker_hangover] )

# Print probabilities
p is_attendant_sleeping
puts "true --> "  +
(is_attendant_sleeping[0]/(is_attendant_sleeping[0]+is_attendant_sleeping[1])).to_s
puts "false --> "  +
(is_attendant_sleeping[1]/(is_attendant_sleeping[0]+is_attendant_sleeping[1])).to_s
```

Naïve Bayes Classifier

- Applies the Bayes Rule with naïve independence assumptions
- Classes must be known
- Requires training

$$Posterior = \frac{Prior * Likelihood}{Evidence}.$$

Photo by **Aya Walraven Otake**
http://www.flickr.com/people/ayalan/

Ruby and Bayesian classifiers

- Ruby Classifier - Bayesian and LSI classification library
  - http://classifier.rubyforge.org/
  - Lucas Carlson, David Fayram II.
- Install the library

gem install classifier

Naïve Bayes example:

- Hosting provider customer services
- Depending on mail subject, send the email:
  - To Technical Department
  - To Commercial Department
- We need a training set with mail subjects already classified as “Technical” or “Commercial”

Working on our classifier (I)

- Create the classifier

```
require 'rubygems'
require 'stemmer'
require 'classifier'

# Create the classifier
classifier = Classifier::Bayes.new('Technical', 'Commercial')
```

Training Sets (I)

- Commercial training set

commercial =
["I already payed this invoice.",
"Do you have any discount for 1 year contract?",
"Do you have discounts?",
"Do you have any affiliate schema?",
"This is my new VISA no",
"I cannot see my invoices",
"When is finishing my free period?",
"I have a friend that is interested in your services, will you make us any offer?",
"I didn't use your services last mont. I shouldn't be charged",
"Do you have any kind of warranty?",
"Is it possible to freeze my account and continue after holidays?",
"I don't want to continue with your services.",
"Great service! Can I upgrade my account?",
"Where are the differences between shared and private hosting?"]

Training Sets (II)

- Technician training set

technician =
["I have limit of quoata exceeded and I cannot see my mail.",
"Is your ftp server working correctly?",
"My mails get delayed too much when my php application sends them.",
"I cannot read my mail",
"I cannot connect to the ftp server.",
"I cannot upload to the ftp server",
"My rails application is not working in your server.",
"Can I use background processes in the shared hosting?",
"Can I install this gem?",
"My php app is not working",
"Can you setup a ssl cert for me please?",
"I cannot connect to my server.",
"How can I connect to subversion?"]

Working on our classifier (II)

- Train the classifier

```
# Train the classifier
technician.each { |technician| classifier.train_Technical technician }
commercial.each { |commercial| classifier.train_Commercial commercial }
```

Working on our classifier (III)

- Use It!

puts classifier.classify "How can I connect to my ftp account?“
-> Technical
puts classifier.classify "Is a private account better?“
-> Commercial
puts classifier.classify "Do you have available rails framework?“
-> Technical
puts classifier.classify "How much is your shared hosting for a whole year?"
-> Commercial
puts classifier.classify "I didn't get my discount in the last invoice. Is there something wrong?"
-> Commercial
puts classifier.classify "Connection Timeout in my ftp connections“
-> Technical
puts classifier.classify "Which is my email outgoing server?“
-> Technical

Genetic Algorithms

- Based on Darwin's evolution theory.
- Evolve to find exact or aproximate optimal solutions to problems

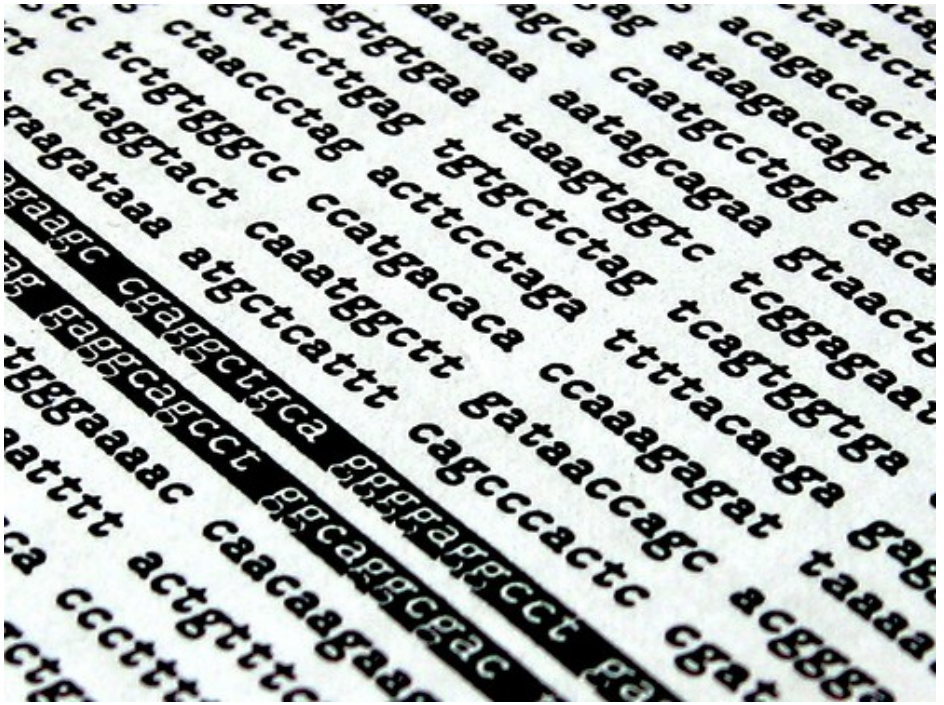


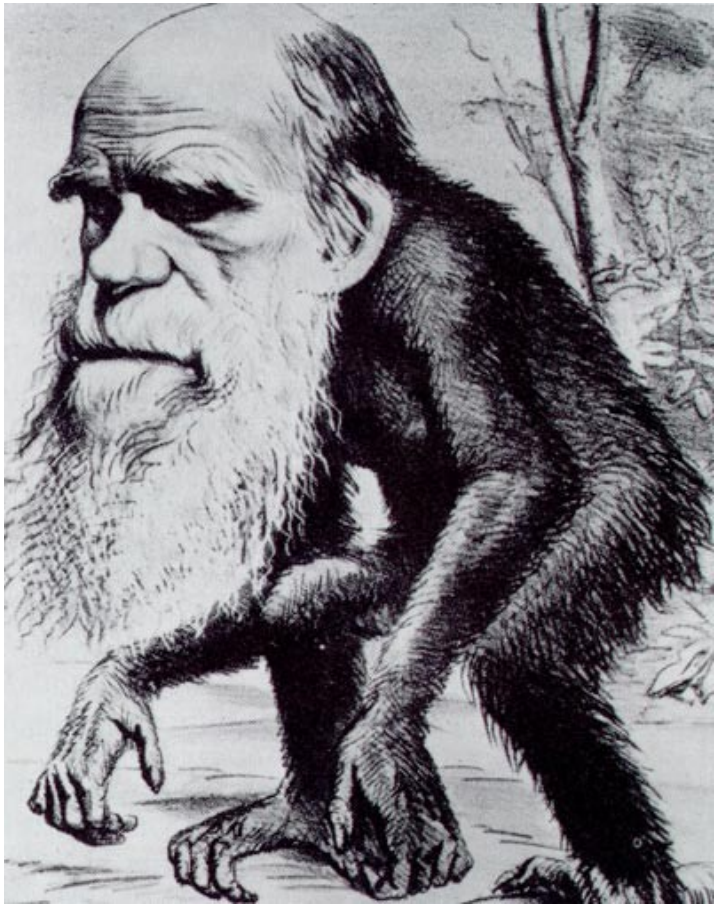
Photo by **John Mu**
http://www.flickr.com/people/jm123467890/

Useful GA example

- Optimize advertisement revenue in your website
- It's done automatically and without human supervision
- Find optimal ad combinations

Darwin's Evolution Theory…

- There is a population (set of individuals)
- Only best-fit individuals survive
- Individuals combine between them to produce new generations
- Natural mutations introduce new peculiarities in the population



GA Requirements

- Individual Data Representation

How do we manage the real data for each individual?

- Fitness function:

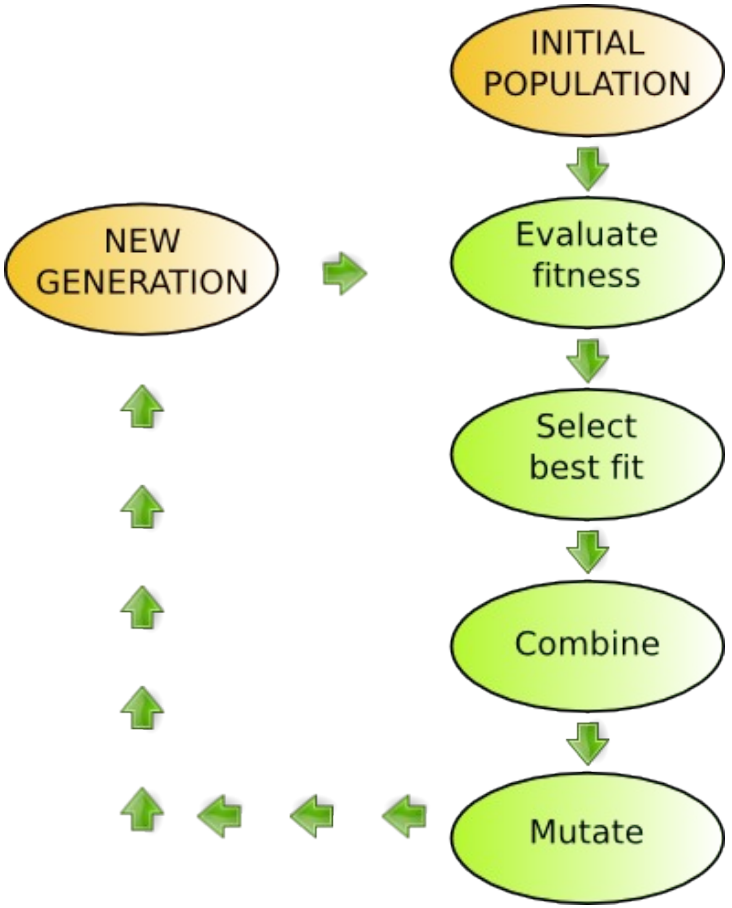
When is an individual better than another one?

- Recombine method

How to create new "child" individuals?

- Mutate method

How to introduce new peculiarities into the population?



Ruby and Genetic Algorithms

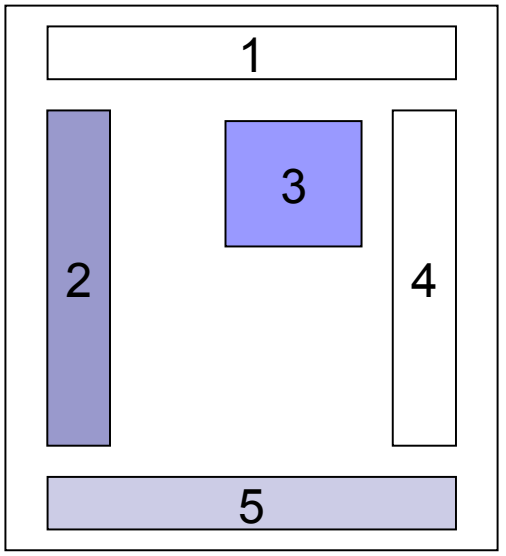
- Gga4r, General Genetic Algorithms for Ruby
  - http://gga4r.rubyforge.org/
  - Sergio Espeja, 2007

- Install the library

```
gem install gga4r
```

Individual representation

- Every ad position is represented in a vector
- Each position can have different ad versions



1	2	3	4	5
X	ad9	ad3	X	ad7

- **Objective:**

  Get the best ad combination -> the best vector

Individual Data Representation

- AdsVector Class.

```
class AdsVector < Array
  attr_accessor :clicks, :prints

  def initialize(value)
    super(value)
    clear
  end

  def clear
    @clicks = 0
    @prints = 0
  end
end
```

- Array that stores the ad combination
- It keeps control of prints and clicks

Fitness function

- In this case an ad combination is better when reaches more CTR (Click Through Ratio).

```
class AdsVector < Array
  attr_accessor :clicks, :prints

  def fitness
          @clicks.to_f/@prints.to_f
  end
end
```

Recombine method

- A random position in the vector is decided. Then, the 2 combined vectors are mixed into 2 new “child” vectors.

```
def recombine(c2)
  cross_point = (rand * c2.size).to_i
  c1_a, c1_b = self.separate(cross_point)
  c2_a, c2_b = c2.separate(cross_point)
  [AdsVector.new(c1_a + c2_b),
  AdsVector.new(c2_a + c1_b)]
end
```

Parents

1	2	3	4	5
X	ad9	ad3	X	ad7

1	2	3	4	5
ad1	ad2	ad3	ad4	ad5

Children

1	2	3	4	5
X	ad9	ad3	ad4	ad5

1	2	3	4	5
ad1	ad2	ad3	X	ad7

Mutate method

- A random ad is placed in a random position sometimes getting new individual configurations.

1	2	3	4	5
X	ad9	ad3	ad2	ad7

1	2	3	4	5
X	ad9	ad3	ad1	ad7

```
ADS = %w{ NONE AD1 AD2 AD3 AD4 AD5 AD6 AD7 AD8 AD9 }

def mutate
  mutate_point = (rand * self.size).to_i
  self[mutate_point] = ADS[(rand * ADS.size).to_i]
end
```

GA Initialization and running

- Create initial population (mainly randomly)
- GA Object creation with the initial population.
- Running: Evolve and check for the best individuals.

Let it play: Create the initial population

```
NUM_POSITIONS = 5
ADS = %w{ NONE AD1 AD2 AD3 AD4 AD5 AD6 AD7 AD8 AD9 }
MAX_POPULATION = 20

def create_initial_population
  num_initial_pop = MAX_POPULATION
  initial_population = []

  num_initial_pop.times do
    individual = AdsVector.new NUM_POSITIONS
    individual.each {|pos, i| individual[i] = ADS[(rand * ADS.size).to_i]}
    initial_population << individual
  end

  return initial_population
end
```

Let it play: Create the GA Object

```
@ads_ga = GeneticAlgorithm.new( create_initial_population,
                                :max_population => MAX_POPULATION )
```

Let it play: Evolve and check for the best individual

```
@ads_ga.evolve
one_best_fit = @ads_ga.best_fit.first
```

What's missing?

- In order to evaluate the fitness function it is required to count ad's prints and clicks.

```
@total_prints = 0

def on_print(individual_id)
  @ads_ga.generations[-1][individual_id].prints += 1
  @total_prints += 1
end

def on_click(individual_id)
  @ads_ga.generations[-1][individual_id].clicks += 1
end
```

Simulation

```
# Simulates a search for the best ad,
# In this simulations an ad combination has more probabilities
# of being clicked when has more coincidences v[i] = Ad_i
# [Ad1, Ad2, NONE, NONE, NONE] is better than [Ad2, Ad1, NONE, NONE, NONE]
def search_best_ads_simulation
  1000.times do |t|
    (@ads_ga.generations[-1].size).times do |individual_id|
      50.times do
        on_print(individual_id)
        prob = 0.0
        @ads_ga.generations[-1][individual_id].each_with_index  do |ad, pos|
          prob += 1.0/NUM_POSITIONS.to_f if "AD#{pos+1}" == ad
        end
        on_click(individual_id) if rand < prob
    end
  end
    puts "Evolution " + @ads_ga.generations.size.to_s
    one_best_fit = @ads_ga.best_fit.first
    puts "Best fitness = #{one_best_fit.fitness}"
    p one_best_fit
    puts "-"*50
    @ads_ga.evolve
    @ads_ga.generations[-1].each { |individual| individual.clear }
end
end
```

Conclusions…

- AI in web applications is more than spam detection.
- Don’t be afraid of AI (at least by now…)
- It’s very easy to use.
- Think about it. AI can improve your web applications.

Questions?

Thanks!

Code and more info at bee.com.es