Ruby master - Feature #4247

New features for Array#sample, Array#choice

01/07/2011 07:20 PM - oj (Yoji Ojima)

Status:	Assigned	
Priority:	Normal	
Assignee:	mame (Yusuke Endoh)	
Target version:		

Description

=begin
We are planning to add the following features of the random sampling to Array.

1. Weighted random sampling.
2. Sampling with replacement.
3. Iteration.

It is discussed in ruby-dev (Feature #3647 and #4147).

API will be:

Array#sample([size, [opt]])

- Random selection without replacement.
- Returns a new array when size is specified.
- opt: weight: proc or array random: Random instance

Array#choice([size, [opt]])

- Random selection with replacement.
- Returns a new array when size is specified.
- opt: same as above.

Array#each_sample([opt])

- Random selection iterator without replacement.
- Choose a random element and yield it.
- Returns an Enumerator if a block is not given.
- opt: same as above.

Array#each_choice([opt])

- Random selection iterator with replacement.
- Choose a random element and yield it.
- Returns an Enumerator if a block is not given.
- opt: same as above.

Comments?
=end

Related issues:

Related to Ruby master - Feature #3647: Array#sample(n, replace=false)	**Feedback**
Related to Ruby master - Feature #4147: Array#sample □□□□□□□□□□	**Feedback**

History

#1 - 01/07/2011 08:08 PM - mame (Yusuke Endoh)

Hi,

2011/1/7 Yoji Ojima redmine@ruby-lang.org:

> We are planning to add the following features of the random sampling to Array.
>
> 1. Weighted random sampling.

2. Sampling with replacement.
3. Iteration.

It is discussed in ruby-dev (Feature #3647 and #4147).

I'm adding complementary information.

There are two reasons why the name "choice" is selected:

- For backward compatibility. Once upon a time, trunk (before 1.9.0) had provided Array#choice. It was backported to 1.8.7. Then, in trunk, the name was changed to Array#sample (see the thread from [ruby-core:18036]). But 1.8.7 still provides Array#choice because it cannot remove Array#choice for compatibility reason.
  Note that 1.8.7's Array#choice does not receive any argument, so there is no compatibility problem.

- Mathematica provides RandomSample and RandomChoice for SRSWOR and SRSWR, respectively
  http://reference.wolfram.com/mathematica/ref/RandomSample.html
  http://reference.wolfram.com/mathematica/ref/RandomChoice.html

There are some algorithms [1] [2] for fast multiple sampling.

[1] Pavlos S. Efraimidis, Paul G. Spirakis
Weighted random sampling with a reservoir
Information Processing Letters
Volume 97, Issue 5 (16 March 2006)
(Ruby implementation is in [ruby-dev:42844])

[2] A. J. Walker
An Efficient Method for Generating Discrete Random Variables with General Distributions
ACM Transactions on Mathematical Software, 3 (1977), 253-256.

Matz roughly approved this suggestion. But he said that the method name of "each_sample" and "each_choice" are a bit awkward, and that he want to hear opinions of ruby-core folks.
Of course, we appreciate any comments about the feature itself rather than the name.

--
Yusuke Endoh mame@tsg.ne.jp

#2 - 01/07/2011 09:30 PM - Eregon (Benoit Daloze)

Hi,

On 7 January 2011 12:08, Yusuke ENDOH mame@tsg.ne.jp wrote:

> Hi,
>
> 2011/1/7 Yoji Ojima redmine@ruby-lang.org:
>
> > We are planning to add the following features of the random sampling to Array.
> >
> > 1. Weighted random sampling.
> > 2. Sampling with replacement.
> > 3. Iteration.
> >
> > It is discussed in ruby-dev (Feature #3647 and #4147).
>
> I'm adding complementary information.
>
> There are two reasons why the name "choice" is selected:
>
> - For backward compatibility. Once upon a time, trunk (before 1.9.0) had provided Array#choice. It was backported to 1.8.7. Then, in trunk, the name was changed to Array#sample (see the thread from [ruby-core:18036]). But 1.8.7 still provides Array#choice because it cannot remove Array#choice for compatibility reason.
>   Note that 1.8.7's Array#choice does not receive any argument, so there is no compatibility problem.

- Mathematica provides RandomSample and RandomChoice for SRSWOR and SRSWR, respectively
  http://reference.wolfram.com/mathematica/ref/RandomSample.html
  http://reference.wolfram.com/mathematica/ref/RandomChoice.html

There are some algorithms [1] [2] for fast multiple sampling.

[1] Pavlos S. Efraimidis, Paul G. Spirakis
Weighted random sampling with a reservoir
Information Processing Letters
Volume 97, Issue 5 (16 March 2006)
(Ruby implementation is in [ruby-dev:42844])

[2] A. J. Walker
An Efficient Method for Generating Discrete Random Variables with General Distributions
ACM Transactions on Mathematical Software, 3 (1977), 253-256.

Matz roughly approved this suggestion. But he said that the method name of "each_sample" and "each_choice" are a bit awkward, and that he want to hear opinions of ruby-core folks.
Of course, we appreciate any comments about the feature itself rather than the name.

--
Yusuke Endoh mame@tsg.ne.jp

Thanks for the name clarification.

I think 'choice' and 'each_choice' are weird.

Sample is "a subset of a population", and so it seems logical to have multiple elements in return.
But choice seems like 'singular', only meant for one element. Is it correct to say "Array#choice returns a choice of some random elements" ? It does not seems right to me.

To this idea, #choice should always return one element, and #sample could be the enumerator (the form which returns an Array would then be Array#sample.take(size)).
However, I guess that would break too much compatibility with current versions.
And I really like 1.9.2 name of #sample for 'a single random element' (even if it might be incorrect to statistics).

Did you consider having another option to {each_,}sample to allow replacement ?
Such as:
ary.sample 3, replace: true

I see the title of the feature is "Array#sample(n, replace=false)", so I guess the idea was there. How was it decided to instead go for another set of methods ?
(Just because of Mathematica's choice ? Compatibility seems against users' interest if it is to use a wrong method name)

Or to follow combinations/permutations:
ary.repeated_sample 3

In http://en.wikipedia.org/wiki/Simple_random_sample, the terms used are "simple random sampling with/without replacement", that comfort me to think choice is not the right .. choice (at least for multiple elements).

I am not a native English speaker, so I might be not accurate about this. If this is the case, please ignore what I said.

About the feature, I wonder if an Hash would be a good idea for the weight option.
It can be very similar to a Proc with #default_proc, except much faster for already stored values.

Off-topic:
I think it would be nice if some ruby-dev/ruby-core discussions could be merged.
Even if I cannot currently understand Japanese, I can at least read

their "coded" ideas.
I would see that as using the same topic, with both languages.

#3 - 01/07/2011 10:14 PM - mame (Yusuke Endoh)

=begin
Hi.

2011/1/7 Benoit Daloze eregontp@gmail.com:

> Did you consider having another option to {each_,}sample to allow replacement ?
> Such as:
> ary.sample 3, replace: true

The word "replace" gives impression of modification, for those who
are familiar with programming and not familiar with statistics.
Especially, Ruby provides Array#replace. For example, I imagine the
following behavior:

```
ary = [1, 2, 3]
p ary.sample(replace: true) #=> 1
p ary #=> [2, 3]
```

> Or to follow combinations/permutations:
> ary.repeated_sample 3

Hmm. Is each_repeated_sample OK?

> Off-topic:
> I think it would be nice if some ruby-dev/ruby-core discussions could be merged.
> Even if I cannot currently understand Japanese, I can at least read
> their "coded" ideas.
> I would see that as using the same topic, with both languages.

Interesting :-)

--
Yusuke Endoh mame@tsg.ne.jp

=end

#4 - 01/07/2011 11:59 PM - Eregon (Benoit Daloze)

=begin
On 7 January 2011 14:14, Yusuke ENDOH mame@tsg.ne.jp wrote:

> 2011/1/7 Benoit Daloze eregontp@gmail.com:
>
> > Did you consider having another option to {each_,}sample to allow replacement ?
> > Such as:
> > ary.sample 3, replace: true
>
> The word "replace" gives impression of modification, for those who
> are familiar with programming and not familiar with statistics.
> Especially, Ruby provides Array#replace. For example, I imagine the
> following behavior:
>
> ```
> ary = [1, 2, 3]
> p ary.sample(replace: true) #=> 1
> p ary #=> [2, 3]
> ```

You are right, it is misleading.
Maybe, (I thought originally to that, but I changed seeing your answer
in [ruby-dev:42811])
ary.sample(n, replacement: true)
is clearer?

Or, to be concise and explicit:
ary.sample(n, repeat: true)

I like this one, what do you think?

Boolean flags in a Hash are not so cool though.
But just a Symbol flag (sample(n, :repeat, opts)) would complicate too
much the method signature.
And I think it is still better than having 2 methods.

> Or to follow combinations/permutations:
> ary.repeated_sample 3

Hmm. Is each_repeated_sample OK?

I think it is not too bad, but 'repeated_sample' give me the
impression there are a few samples, while it is a single one with
replacement.

> Off-topic:
> I think it would be nice if some ruby-dev/ruby-core discussions could be merged.
> Even if I cannot currently understand Japanese, I can at least read
> their "coded" ideas.
> I would see that as using the same topic, with both languages.

Interesting :-)

Sure, any concrete idea how that could be made possible?

=end

#5 - 01/08/2011 02:15 AM - tenderlovemaking (Aaron Patterson)

=begin
On Fri, Jan 07, 2011 at 08:08:03PM +0900, Yusuke ENDOH wrote:

> Hi,
>
> 2011/1/7 Yoji Ojima redmine@ruby-lang.org:
>
> > We are planning to add the following features of the random sampling to Array.
> >
> > 1. Weighted random sampling.
> > 2. Sampling with replacement.
> > 3. Iteration.
> >
> > It is discussed in ruby-dev (Feature #3647 and #4147).

[snip]

> Matz roughly approved this suggestion. But he said that the method name
> of "each_sample" and "each_choice" are a bit awkward, and that he want
> to hear opinions of ruby-core folks.
> Of course, we appreciate any comments about the feature itself rather
> than the name.

Why not provide "samples" that returns an Enumerator? Then you could
say:

list.samples.each { |x| .... }
list.samples.map { |x| .... }

etc.

--
Aaron Patterson
http://tenderlovemaking.com/

Attachment: (unnamed)
=end

#6 - 01/08/2011 02:47 AM - Eregon (Benoit Daloze)

=begin
On 7 January 2011 18:14, Aaron Patterson aaron@tenderlovemaking.com wrote:

> Why not provide "samples" that returns an Enumerator? Then you could
> say:
>
> list.samples.each { |x| .... }
> list.samples.map { |x| .... }
>
> etc.
>
> --
> Aaron Patterson
> http://tenderlovemaking.com/

I thought to that too (like String#lines), but as I said upper, a
sample is already supposed to be a set of random elements.
And so, you are iterating on random elements, of a growing sample.
Or you could see that as creating a new sample by adding one element
to the old sample.
That would make sense.

Anyway, we already have Array#sample returning a single element, so
Array#samples seems logical.
I prefer it to Array#each_sample too.

About returning an Enumerator, you could also yield like #each if a
block is given (again, like String#lines).

Is there a reason to change Array#sample instead of using the
enumerator form with take(n) to get an Array of n random elements ?
Is it so much more efficient ?

I find weird to have a method (#sample) returning a single element or
an Array depending on parameters.

So here is my proposition:
Array#sample remains unchanged.
Array#samples is like the proposed #each_sample, with the option
:repeat to use replacement.

=end

#7 - 01/11/2011 10:39 PM - mame (Yusuke Endoh)

=begin
Hi,

2011/1/8 Benoit Daloze eregontp@gmail.com:

> Is there a reason to change Array#sample instead of using the
> enumerator form with take(n) to get an Array of n random elements ?
>
> I find weird to have a method (#sample) returning a single element or
> an Array depending on parameters.

You have a misunderstanding. Array#sample already supports optional
argument to specify the count of elements sampled. This is not a new
behavior.

$ ruby -ve 'p [1, 2, 3].sample(2)'
ruby 1.9.2p0 (2010-08-18 revision 29036) [i686-linux]
[2, 1]

Unfortunately, this behavior is included in 1.9.2 which is already
released. It can no longer change.

> So here is my proposition:
> Array#sample remains unchanged.
> Array#samples is like the proposed #each_sample, with the option
> :repeat to use replacement.

Though I'm not against your proposition, it may be a bit confusing
because Array#sample may also return some elements.

Also, does anyone have an opinion about the keyword :repeat which
allows duplicated samples? Personally, I don't hate.

p [1, 2, 3].sample(5, repeat: true) #=> [2, 2, 3, 1, 3]

--
Yusuke Endoh mame@tsg.ne.jp

=end

#8 - 01/12/2011 03:14 AM - Eregon (Benoit Daloze)

=begin
On 11 January 2011 14:39, Yusuke ENDOH mame@tsg.ne.jp wrote:

> Hi,
>
> 2011/1/8 Benoit Daloze eregontp@gmail.com:
> You have a misunderstanding. Array#sample already supports optional
> argument to specify the count of elements sampled. This is not a new
> behavior.
>
> $ ruby -ve 'p [1, 2, 3].sample(2)'
> ruby 1.9.2p0 (2010-08-18 revision 29036) [i686-linux]
> [2, 1]
>
> Unfortunately, this behavior is included in 1.9.2 which is already
> released. It can no longer change.

Sorry about that, I just missed it.
I take notice to look ri whenever I speak of a method on #ruby-core.

> > So here is my proposition:
> > Array#sample remains unchanged.
> > Array#samples is like the proposed #each_sample, with the option
> > :repeat to use replacement.
>
> Though I'm not against your proposition, it may be a bit confusing
> because Array#sample may also return some elements.

Yes, I did not have that in mind, but I still prefer #samples to #each_sample.

So, this is close to the original proposition, except using "repeat:
true" instead of *choice and renaming #each_sample to #samples.

It might be confusing, but I think the general usage with chaining
enumerators will be nicer:
ary.samples.map { ... }
vs
ary.each_sample.map { ... }

I do not like using #each_slice in a chain for this reason.

But if people think it is too confusing, then let #each_sample be.

=end

#9 - 01/14/2011 03:49 PM - gunn (Arthur Gunn)

=begin
Like Benoit said, Array#choice does sound like it would return only one element, I very much like the proposal of:

p [1, 2, 3].sample(5, repeat: true) #=> [2, 2, 3, 1, 3]

I don't think the distinction between #sample and #samples is obvious enough, the naming suggests #sample would return one element, and #samples many, not that the latter is an iterator.

#each_sample is a little ugly, but much more self-explanatory I think.

How important is it to have something like #each_sample though? Personally, I would probably always do something like this:

[1, 2, 3, 4, 5].sample(3).each { |n| puts n }
=end

#10 - 03/25/2012 02:18 PM - mame (Yusuke Endoh)

- Description updated

- Status changed from Open to Assigned

- Assignee set to mame (Yusuke Endoh)

#11 - 03/25/2012 10:03 PM - trans (Thomas Sawyer)

Whatever happened to #pick and #pick! which picked one random element? The term #sample strongly suggests the return of a subset.

Anther consideration, maybe it would be more useful to use a random delegator.

enum.random.sample
enum.random.subset
enum.random.element
enum.random.permutations
...

There can be many more random functions.

#12 - 11/20/2012 02:31 AM - mame (Yusuke Endoh)

- Target version set to 2.6

My apologies, I forgot this ticket completely. I should have wrapped up this discussion.
This missed the deadline of 2.0.0. I'm setting this to next minor. Sorry.

--
Yusuke Endoh mame@tsg.ne.jp

#13 - 12/25/2017 06:14 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)