Ruby master - Feature #16828

Introduce find patterns

05/04/2020 04:13 AM - ktsj (Kazuki Tsujimoto)

Status:	Closed	
Priority:	Normal	
Assignee:	ktsj (Kazuki Tsujimoto)	
Target version:		

Description

I propose to add find pattern to pattern matching.

Specification

```
find_pattern: Constant(*var, pat, ..., *var)
            | Constant[*var, pat, ..., *var]
            | [*var, pat, ..., *var]
```

This pattern is similar to array pattern, except that it finds the first match values from the given object.

```
case ["a", 1, "b", "c", 2, "d", "e", "f", 3]
in [*pre, String => x, String => y, *post]
  p pre  #=> ["a", 1]
  p x    #=> "b"
  p y    #=> "c"
  p post #=> [2, "d", "e", "f", 3]
end
```

Note that, to avoid complexity, the proposed feature doesn't support backtracking. Thus, the following code raises a NoMatchingPatternError.

```
case [0, 1, 2]
in [*, a, *] if a == 1
end
```

Implementation

- https://github.com/k-tsj/ruby/tree/find-pattern-prototype

Associated revisions

Revision ddded115 - 06/14/2020 12:24 AM - ktsj (Kazuki Tsujimoto)

Introduce find pattern [Feature #16828]

History

#1 - 05/04/2020 09:03 AM - shyouhei (Shyouhei Urabe)

I am a noob in this area who don't understand what is good about it. Can you show us a bit more realistic example? The proposal kind of seems illustrative to me.

#2 - 05/04/2020 01:38 PM - shevegen (Robert A. Heiler)

Is this a bit like .scan(), but applied to the pattern matching style? (Not
sure if I understood this completely but it reminds me a bit of using .scan()
and MatchData on the result possibly.)

#3 - 05/27/2020 12:07 AM - matz (Yukihiro Matsumoto)

Suppose we have the following code:

```
json = {name: "Alice", children: [{name: "Bob", age: 6}, {name: "Chad", age: 4}]}
case json
in {name: "Alice", children: [{name: "Bob", age: age}]}
  p age
```

```
end
```

It doesn't match because array pattern does not search for the element. It only matches for an array with single element named "Bob". So to find an element in an array with multiple elements, you have to use find pattern like:

```
json = {name: "Alice", children: [{name: "Bob", age: 6}, {name: "Chad", age: 4}]}
case json
in {name: "Alice", children: [*, {name: "Bob", age: age}, *]}
  p age
end
```

Matz

#4 - 05/27/2020 12:09 AM - matz (Yukihiro Matsumoto)

Accepted. But we have to clarify the term "backtracking" here. It does backtrack but not with a guard, right?

Matz.

#5 - 05/27/2020 07:07 PM - Dan0042 (Daniel DeLorme)

From the implementation linked above:

```
case [0, 1, 2]
in [*, a, *]
  p a #=> 0; non-greedy match
end

case [0, 1, 2]
in [*, b, 2, *]
  p b #=> 1; backtracking
end
```

So yes, it supports "backtracking" in a certain fashion otherwise the second case wouldn't match.

Note that it only supports splat expressions at the beginning and end of the array:

```
case [0, 1, 2, 3, 4, 5, 6]
in [*, a, *, b, 5, *]
  p a, b
end
# syntax error, unexpected ',', expecting ']'
# in [*, a, *, b, 5, *]
```

#6 - 05/31/2020 09:59 AM - ktsj (Kazuki Tsujimoto)

- Assignee set to ktsj (Kazuki Tsujimoto)

- Status changed from Open to Assigned

> Accepted.

Thank you. I'll commit it.

> But we have to clarify the term "backtracking" here. It does backtrack but not with a guard, right?

Yes. This is exactly what I intend.

#7 - 05/31/2020 02:20 PM - sawa (Tsuyoshi Sawada)

- Description updated

#8 - 06/14/2020 12:28 AM - ktsj (Kazuki Tsujimoto)

- Status changed from Assigned to Closed

Applied in changeset git|ddded1157a90d21cb54b9f07de35ab9b4cc472e1.

Introduce find pattern [Feature #16828]