In [1]:

```
%matplotlib notebook

import os, sys
sys.path.insert(1, os.path.join(sys.path[0], '..'))
import database
from snowballing.operations import reload, work_by_varname, load_work_map_all_years
from snowballing.strategies import Strategy, State
import custom_strategies
from functools import reduce
from matplotlib_venn import venn2, venn2_circles
from matplotlib import pyplot as plt
import pandas as pd

custom_strategies.LIMIT_YEAR = 2015
reload()
# !pip install matplotlib-venn
```

Contents [+] ↻ r

In [2]:

```python
from snowballing.operations import metakey, metakey_title

def busca(*libraries, seed=None, filter_function=None):
    filter_function = filter_function or (lambda x: x.category in ("snowball",))
    if seed is not None:
        iterable = ((1, work_by_varname(x)) for x in seed)
    else:
        iterable = load_work_map_all_years()
    seedset = set()
    visited = set()
    for _, work in iterable:
        for library in libraries:
            if int(getattr(work, library, 0)):
                visited.add(work)
                if filter_function(work):
                    seedset.add(work)
                break
    return seedset, filter_function, visited

def descreve_delta(strategy, state, name):
    target = state.find(name)
    previous_related = reduce(lambda x, y: x | y, (s.related for s in target.previous[0]),
    for work in previous_related:
        backward = set(strategy.ref[work]) & target.delta_visited
        if backward:
            print('backward', work.metakey)
            for ref in backward:
                print('-', ref.metakey, 'related' if ref in target.delta_related else '')
        forward = set(strategy.rev_ref[work]) & target.delta_visited
        if forward:
            print('forward', work.metakey)
            for ref in forward:
                print('-', ref.metakey, 'related' if ref in target.delta_related else '')

def separa_backward_forward(state):
    backward = set()
    forward = set()
    stack = [state]
    visited = {id(state)}
    while stack:
        current = stack.pop()
        if current.previous:
            if current.previous[1] == "backward":
                backward |= current.delta_related
            if current.previous[1] == "forward":
                forward |= current.delta_related

            antecessors = current.previous[0]
            for previous in antecessors:
                if id(previous) not in visited:
                    visited.add(id(previous))
                    stack.append(previous)
    return backward, forward

def encontraria(strategy, state):
    backward = set()
    forward = set()
    related = state.related - state.find("s0").related
    for work in state.related:
```

```
        backward |= (set(strategy.ref[work]) & related)
        forward |= (set(strategy.rev_ref[work]) & related)
    return backward, forward

#busca("scopus", seed=["wohlin2014a", "briand2000a"], filter_function=lambda x: True)
#busca("scopus", filter_function=lambda x: False)
```

1 Estratégias

1.1 Estratégia 1 - Busca em todas Digital Libraries (DL)

In [3]:

```
print ("Total de estudos primários usados como Seed Set: ", len ([x for _, x in load_work_m
```

```
Total de estudos primários usados como Seed Set:  22
```

In [4]:

```
print ("Lista de estudos primários usados como Seed Set:")
([x for _, x in load_work_map_all_years() if x.category == "snowball"])
```

Lista de estudos primários usados como Seed Set:

Out[4]:

[Exploring the use of the cynefin framework to inform software development a
pproach decisions,
 An elicitation instrument for operationalising GQM+ Strategies (GQM+ S-EI),
 Strategically balanced process adoption,
 Application of GQM+ Strategies{   extregistered} in the Japanese space
industry,
 Software engineering strategies: aligning software process improvement with
strategic goals,
 Defining and monitoring strategically aligned software improvement goals,
 Linking software development and business strategy through measurement,
 Integration of strategic management, process improvement and quantitative m
easurement for managing the competitiveness of software engineering organiza
tions,
 Utilizing GQM+ Strategies for an organization-wide earned value analysis,
 Utilizing GQM+ Strategies for business value analysis: An approach for eval
uating business goals,
 Software process improvement: Supporting the linking of the software and th
e business strategies,
 Entropy based software processes improvement,
 An approach to support the strategic alignment of software process improvem
ent programs,
 Strategic alignment of software process improvement programs using QFD,
 ProPAMet: a Metric for process and project alignment,
 A Low-overhead method for software process appraisal,
 Business-oriented process improvement: practices and experiences at Thales
Naval The Netherlands (TNNL),
 Measuring and improving software process in China,
 A business goal-based approach to achieving systems engineering capability
maturity,
 Applying and adjusting a software process improvement model in practice: th
e use of the IDEAL model in a small software enterprise,
 Managing process inconsistency using viewpoints,
 SPI:□I can't get no satisfaction□-directing process improvement to meet bus
iness needs]

In [5]:

```
reload()
TOTAL = [x for _, x in load_work_map_all_years() if x.category == "snowball"]
filter_function = lambda x: x.category in ("snowball", "forward", "backward")
```

In [6]:

```python
from snowballing.strategies import State

reload()
def busca_completa(libs, filter_function):
    union = None
    for dl in libs:
        strategy = Strategy(*busca(dl, filter_function=filter_function))
        strategy.initial.name = dl
        if union is None:
            union = strategy.initial.derive("union", name="s0")
        else:
            union.visited.update(strategy.initial.visited)
            union.related.update(strategy.initial.related)
            union.previous[0].append(strategy.initial)

    State.last_id = 0
    strategy.initial = union
    return strategy

strategy = busca_completa(["scopus", "ieee", "acm", "springer", "elcompendex", "webofscienc
#strategy.initial.find("acm")
strategy.initial
```

scopus
visited: 17
related: 8

union

ieee
visited: 26
related: 2

union

In [7]:
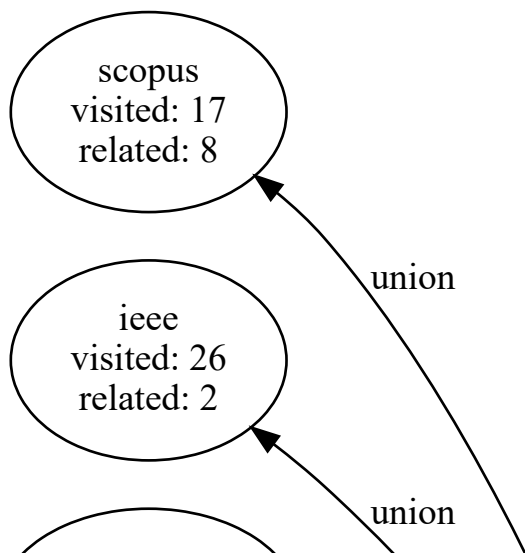
```python
array = []

name = "E1"
print ("Precision " + name)
EP = len (strategy.initial.related) / len (strategy.initial.visited)
print (EP)
print ("Recall " + name)
ER = len (strategy.initial.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E1
0.02826086956521739
Recall E1
0.5909090909090909
```

1.2 Estratégia 2 - Busca em todas Digital Libraries + Snowballing (Guideline - sfbu)
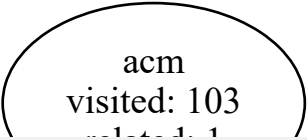
In [8]:

```python
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = busca_completa(["acm", "scopus", "springer", "elcompendex", "webofscience", "sci
#Strategy(*busca("acm","scopus", filter_function=filter_function))

state = strategy.sfbu()
#state = strategy.bbff()
#state = strategy.ffbb()
#state = strategy.s2ffbb2u()

state
```

acm
visited: 103

In [9]:

```python
def duplicados(strategy, state):
    encontrados = set()
    result = set()
    soma = 0
    related = state.related - state.find("s0").related
    for work in state.related:
        inter = encontrados & (set(strategy.ref[work]) & related)
        if inter:
            soma += len(inter)
            encontrados |= inter
            print(work @ metakey, "backward", inter @ metakey)
        inter = encontrados & (set(strategy.rev_ref[work]) & related)
        if inter:
            soma += len(inter)
            encontrados |= inter
            print(work @ metakey, "forward", inter @ metakey)
    return soma
```

In [10]:

```
sets = separa_backward_forward(state)
sets = encontraria(strategy, state)
v = venn2(sets, set_labels = ('Backward', 'Forward'))
c = venn2_circles(sets)
plt.title('Diagrama de Venn')
plt.show()
```

In [11]:

```
#Backward - Execução de backward em sets[0]
print("Backward", sets[0] @ metakey)
print("Forward", sets[1] @ metakey)
```

```
Backward ['trienekens2009a', 'becker2008b']
Forward ['petersen2015a', 'mandić2010a']
```

1.2.0.1 Resumo do Precision e Recall

In [12]:

```
E2P = len (strategy.initial.related) / len (strategy.initial.visited)
#print (len (strategy.initial.related))
#print (len (strategy.initial.visited))
#print (E2P)
print ("Precision na Busca E2: %.2f%%" % E2P)

#print ("Precision no Snowballing - E2:")
#print(len(state.related - state.find("s0").related))
#print(len(state.visited - state.find("s0").visited))

E2PS = len(state.related - state.find("s0").related) / len(state.visited - state.find("s0")

print ("\n""Precision no Snowballing E2: %.2f%%" % E2PS)

print("Precision no Snowballing E2 - Forward - 1 iteração: %.2f%%" % (len (state.find("s1")

print("Precision no Snowballing E2 - Backward - 1 iteração: %.2f%%" % (len (state.find("s2"

print("\n""Precision E2 (Busca + Snowballing): %.2f%%" % (len(state.related) / len(state.vi

#len (state.find("s1").related)

#len (state.find("s1").delta_related)
#len (state.find("s1").delta_visited)


name = "E2"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision na Busca E2: 0.03%

Precision no Snowballing E2: 0.01%
Precision no Snowballing E2 - Forward - 1 iteração: 0.01%
Precision no Snowballing E2 - Backward - 1 iteração: 0.01%

Precision E2 (Busca + Snowballing): 0.02%
Precision E2
0.016748768472906402
Recall E2
0.7727272727272727
```

In [13]:

```python
from collections import deque
def precision_recall(state, total, stop=""):
    # Precisao 0 quando nao visita nada
    array = [[
        "state", "precision", "recall", "operation",
        "related", "visited", "delta_related", "delta_visited",
        "accumulated_precision", "accumulated_recall",
    ]]
    stack = deque([state])
    visited = {id(state)}
    while stack:
        current = stack.pop()
        try:
            accumulated_precision = len(current.related) / len(current.visited)
            precision = len(current.delta_related) / len(current.delta_visited)
        except ZeroDivisionError:
            precision = 0.0

        array.append([
            current.name,
            precision,
            len(current.delta_related) / len(total),
            current.previous[1] if current.previous else "-",
            len(current.related),
            len(current.visited),
            len(current.delta_related),
            len(current.delta_visited),
            accumulated_precision,
            len(current.related) / len(total),
        ])
        if current.name == stop:
            break
        if current.previous:
            antecessors = current.previous[0]
            for previous in antecessors:
                if id(previous) not in visited:
                    visited.add(id(previous))
                    stack.appendleft(previous)
    return array
```

In [14]:

```
df = pd.DataFrame(list(reversed(precision_recall(state, TOTAL))))
df.columns = df.iloc[-1]
df = df.drop(df.index[-1])
df
```

Out[14]:

14	state	precision	recall	operation	related	visited	delta_related	delta_visited
0	ieee	0.0769231	0.0909091	-	2	26	2	26
1	sciencedirect	0.00549451	0.0454545	-	1	182	1	182
2	webofscience	0.5	0.227273	-	5	10	5	10
3	elcompendex	0.4375	0.318182	-	7	16	7	16
4	springer	0	0	-	0	128	0	128
5	scopus	0.470588	0.363636	-	8	17	8	17
6	acm	0.00970874	0.0454545	-	1	103	1	103
7	s0	0	0	union	13	460	0	0
8	s2	0.0077821	0.0909091	backward	15	717	2	257
9	s1	0.00947867	0.0909091	forward	15	671	2	211
10	s3	0	0	union	17	927	0	0
11	s5	0	0	backward	17	986	0	59
12	s4	0	0	forward	17	956	0	29
13	s6	0	0	union	17	1015	0	0

In [15]:

```python
fig = plt.figure()
df['precision'].plot(legend=True)
df['recall'].plot(legend=True)
ax = plt.gca()
ax.set_xticklabels(df["state"] + "\n" + df["operation"])
ax.set_title("By State");
```

In [16]:

```
fig = plt.figure()
df['accumulated_precision'].plot(legend=True)
df['accumulated_recall'].plot(legend=True)
ax = plt.gca()
ax.set_xticklabels(df["state"] + "\n" + df["operation"])
ax.set_title("Accumulated");
```

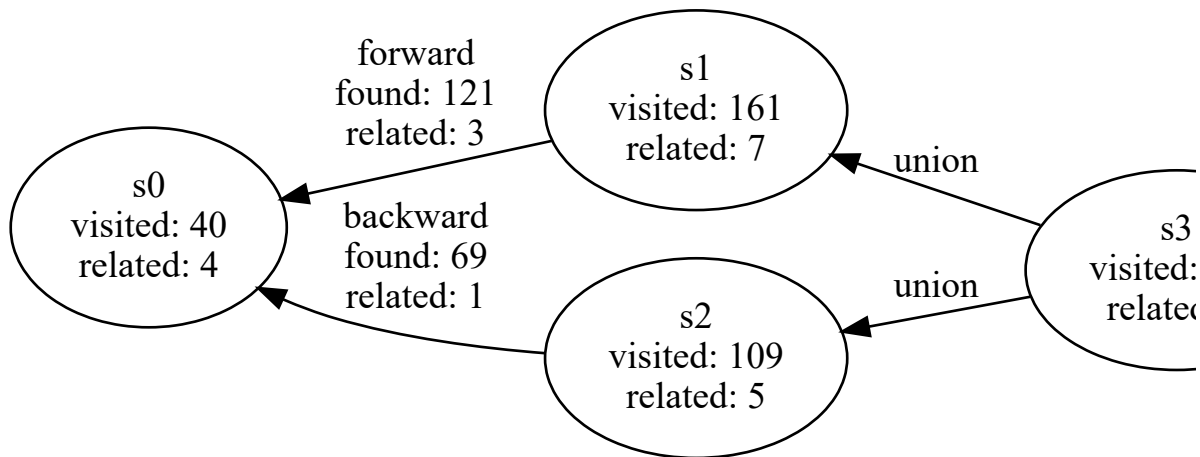## 1.3 Estratégia 3 - Busca Informal (Google Scholar) + Snowballing (Guideline - sfbu)

In [17]:

```
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = Strategy(*busca("gs", filter_function=filter_function))
state = strategy.sfbu()
state
```


In [18]:

```
name = "E3"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E3
0.02313624678663239
Recall E3
0.4090909090909091
```

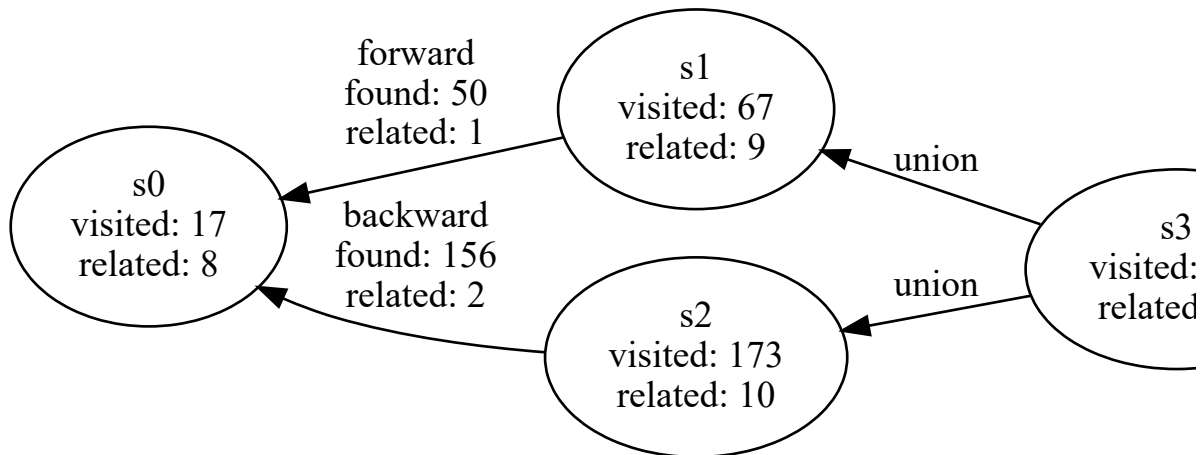## 1.4 Estratégia 4 - Busca em Scopus + Snowballing (Guideline - sfbu)

In [19]:

```
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = Strategy(*busca("scopus", filter_function=filter_function))

#from copy import copy
#strategy.initial.visited = copy(strategy.initial.related)
state = strategy.sfbu()
state
```


In [20]:

```
name = "E4"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E4
0.029345372460496615
Recall E4
0.5909090909090909
```

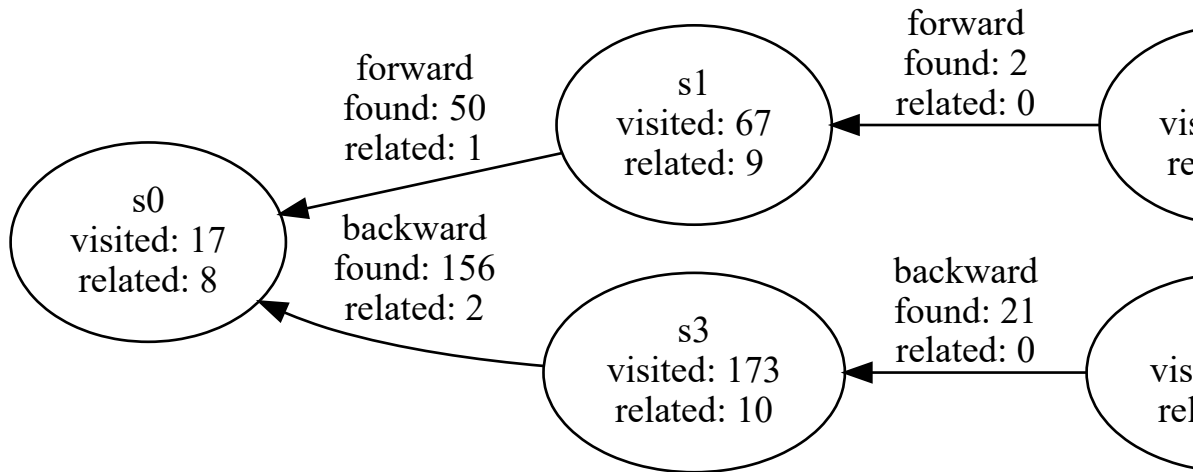## 1.5 Estratégia 5 - Busca em Scopus + Snowballing (Short Paper - s2ffbb2u)

In [21]:

```
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = Strategy(*busca("scopus", filter_function=filter_function))
state = strategy.s2bbff2u()
state
```

forward
found: 2
related: 0

forward
found: 50
related: 1

s1
visited: 67
related: 9

vis
re

s0
visited: 17
related: 8

backward
found: 156
related: 2

backward
found: 21
related: 0

s3
visited: 173
related: 10

vis
rel

In [22]:

```
name = "E5"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E5
0.044715447154471545
Recall E5
0.5
```

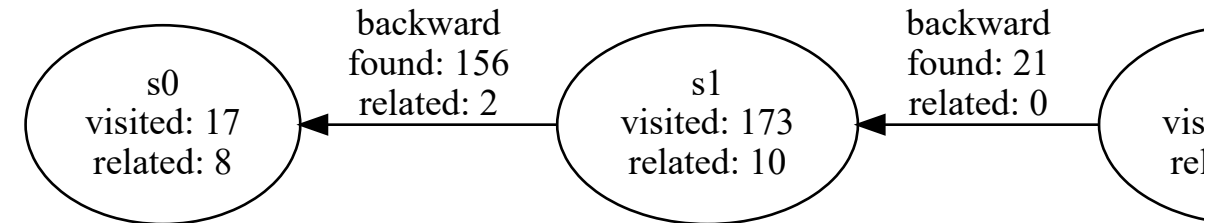## 1.6 Estratégia 6 - Busca em Scopus + Snowballing (JF - bbff)

In [23]:

```python
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = Strategy(*busca("scopus", filter_function=filter_function))
state = strategy.bbff()
state
```


In [24]:

```python
name = "E6"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E6
0.03513513513513514
Recall E6
0.5909090909090909
```

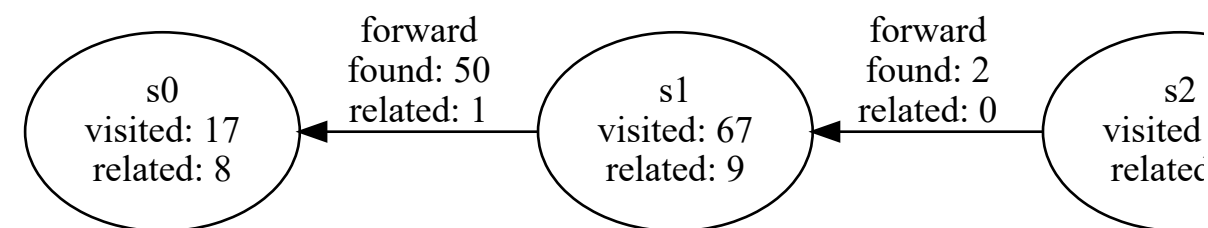## 1.7 Estratégia 7 - Busca em Scopus + Snowballing (JF - ffbb)

In [25]:

```python
reload()
filter_function = lambda x: x.category in ("snowball", "forward", "backward")

strategy = Strategy(*busca("scopus", filter_function=filter_function))
state = strategy.ffbb()
state
```

In [26]:

```
name = "E7"
print ("Precision " + name)
EP = len (state.related) / len (state.visited)
print (EP)
print ("Recall " + name)
ER = len (state.related) / len (TOTAL)
print (ER)

array.append((name, EP, ER))
```

```
Precision E7
0.037037037037037035
Recall E7
0.5454545454545454
```

2 Análise das Estratégias

In [27]:

```
%matplotlib notebook
import pandas as pd
```

In [28]:

```
df = pd.DataFrame (array,columns = ['Estratégia','Precision','Recall'])
df.index = df['Estratégia']
df
```
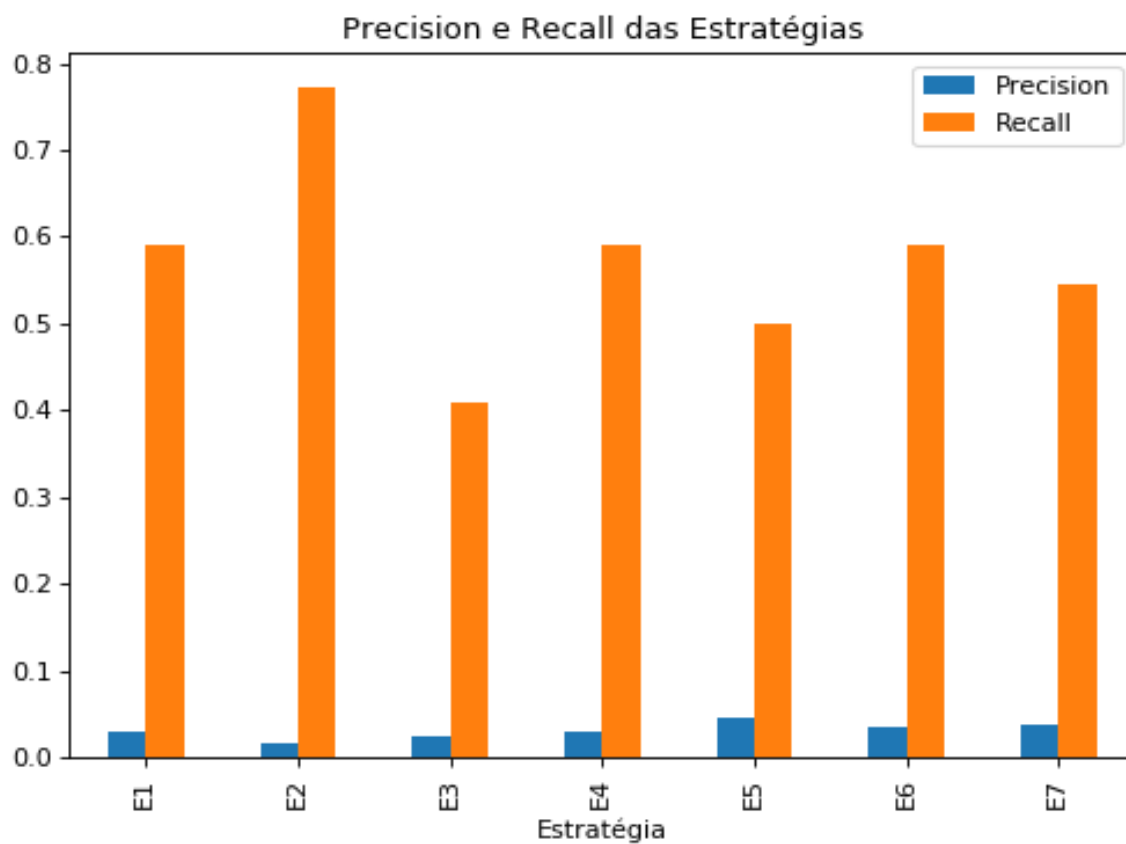
Out[28]:

	Estratégia	Precision	Recall
Estratégia			
E1	E1	0.028261	0.590909
E2	E2	0.016749	0.772727
E3	E3	0.023136	0.409091
E4	E4	0.029345	0.590909
E5	E5	0.044715	0.500000
E6	E6	0.035135	0.590909
E7	E7	0.037037	0.545455

In [29]:

```python
import matplotlib.pyplot as plt
ax = df.plot.bar(title ="Precision e Recall das Estratégias")
plt.tight_layout()
```


In [30]:

```python
df
```

Out[30]:

	Estratégia	Precision	Recall
Estratégia			
E1	E1	0.028261	0.590909
E2	E2	0.016749	0.772727
E3	E3	0.023136	0.409091
E4	E4	0.029345	0.590909
E5	E5	0.044715	0.500000
E6	E6	0.035135	0.590909
E7	E7	0.037037	0.545455