Interval Scheduling Demo

February 9, 2022

Interval Request Scheduling – Greedy Algorithm

[3]:
```
import random

# random pair (s, f) where s and f are integers between
# 0 and 99, inclusive
def random_request():
    return tuple(sorted(random.sample(range(0, 100), 2)))

# make n random requests and return them all
def make_requests(n):
    #requests = []
    #for i in range(n):
    #    requests.append(random_request())
    #return requests
    return [random_request() for i in range(n)]
```

[2]:
```
random_request()
```

[2]:
```
(12, 72)
```

random_request()

[4]:
```
make_requests(10)
```

[4]:
```
[(25, 71),
 (27, 75),
 (33, 94),
 (31, 66),
 (22, 98),
 (24, 76),
 (15, 54),
 (89, 91),
 (0, 99),
 (4, 31)]
```

[12]:
```
def plot_requests(requests):
    # sort the requests
    # plot one per line
```

```
    sorted_requests = sorted(requests, key=lambda x : x[1])

    for r in sorted_requests:
        # r is one tuple (start, end)
        print(" "*(r[0]) + "-"*(r[1]-r[0]))
```

[13]:
```
R = make_requests(10)
print(R)
plot_requests(R)
```

```
[(21, 70), (46, 73), (3, 99), (58, 77), (68, 71), (31, 65), (65, 82), (24, 76),
(20, 65), (29, 80)]
                                 ----------------------------------
                    ---------------------------------------------
                     ------------------------------------------------
                                                                 ---
                                              ---------------------------
                        ------------------------------------------------
                                                        -------------------
                              ---------------------------------------------------
-----------------
   ------------------------------------------------------------------------------
-------------------
```

[41]:
```
def greedy_solution(requests):
    sorted_requests = sorted(requests, key=lambda x: x[1])
    # O(n * log(n))
    solution = []

    # remove the thing at index 0 from sorted_requests
    # AND append to solution, in one line
    solution.append(sorted_requests.pop(0))

    while len(sorted_requests) > 0:
        request = sorted_requests.pop(0)

        # if the start time of request is > the end
        # time of the latest meeting I've accepted:
        if request[0] >= solution[-1][1]:
            solution.append(request)

    return solution
```

[47]:
```
requests = make_requests(100)
```

[48]:
```
greedy_sol = greedy_solution(requests)
```

[49]:
```
len(greedy_sol)
```

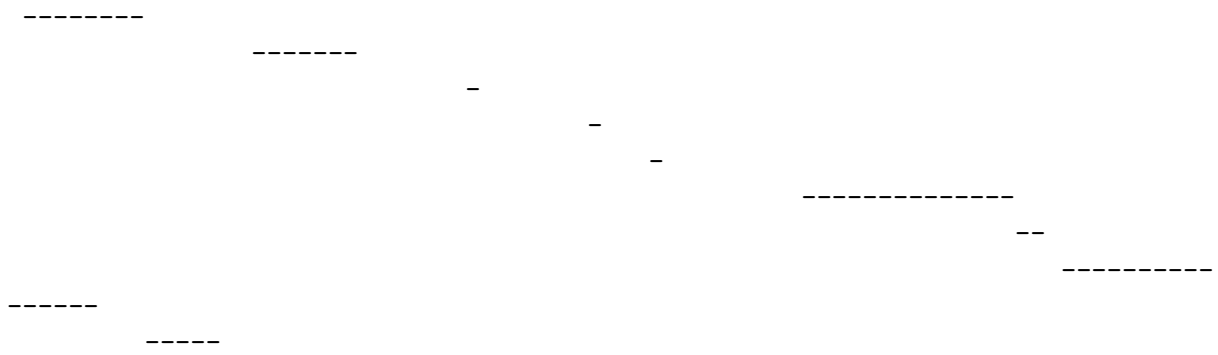
[49]: 10

[50]:
```
print(greedy_sol)
```

```
[(2, 10), (17, 24), (31, 32), (39, 40), (43, 44), (53, 67), (67, 69), (70, 80),
(81, 87), (90, 95)]
```

[ ]:

[51]:
```
plot_requests(greedy_sol)
```



[52]:
```
len(greedy_sol)
```

[52]: 10

[ ]: