2024 여름학기 연구참여 보고서

POSTECH IME Logistics Lab

2024.06.10 ~ 2024.08.02 (8주)

지도: 김병인 교수님, 김준 사수님, 김한솔 사수님

참여: 임성은 (POSTECH IME)

1. 국내 바지 항차 스케줄링 알고리즘 개발

1.1. 연구 과제

본 과제는 삼성중공업의 바지 항차 스케줄링 알고리즘을 개발하는 것으로, 바지선의 가용 면적, 가용 시간, 바지선의 초기위치와 종료위치, 가용시작시간과 가용종료시간, 안벽의 최대 접안 개수, 조수간만을 고려한 접안 가능 시간, 선하역 가능 시간, 옮기고자 하는 블록의 선적예정지, 선적요청시간, 하역예정지, 하역요청시간 등의 조건을 만족하는 바지선의 블록 항차 계획을 수립하는 것을 목표로 한다.

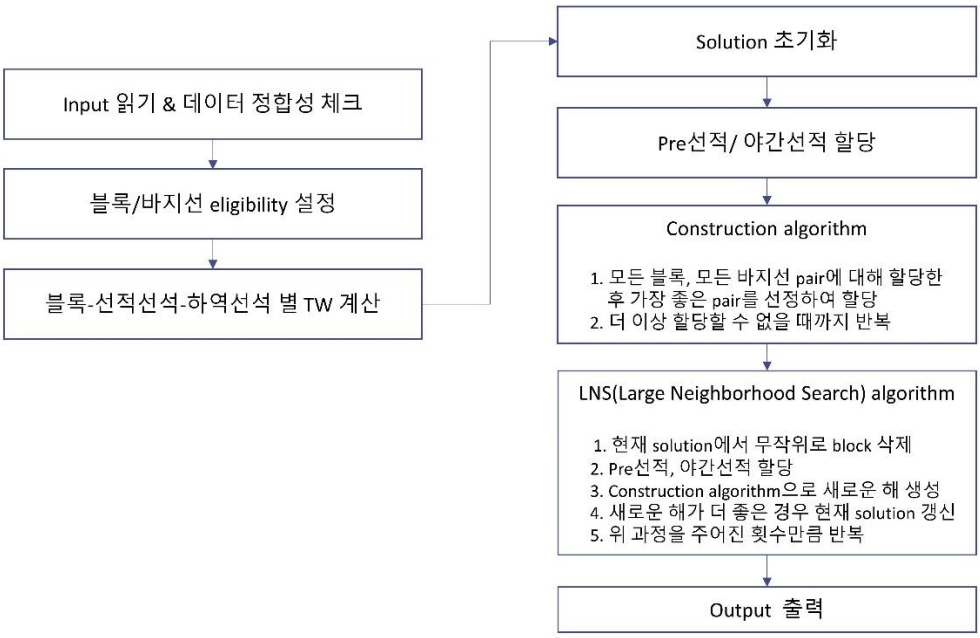
연구 참여 시작 시 본 과제는 LNS 알고리즘을 기반으로 한 스케줄링 알고리즘 개발 중이었으며, 알고리즘을 반영한 C++과 python 프로그램이 구현된 상태였다. 큰 문제를 풀기 위해서는 프로그램의 성능이 높아야 한다. 그러나 python 프로그램은 C++ 프로그램에 비해 약 25배 느리다는 문제가 있었다. 본 과제에서 Python 프로그램 가속화를 전담하여 해결하였으며, 모의데이터 생성, 데이터 정합성 확인을 부가적으로 담당하였다.

1.2. 연구 과정

본 과제에서 전담한 Python 프로그램 가속화를 이하 '본 연구'로 일컫는다. 본 연구는 문제 이해, 문제 이해를 바탕으로 한 C++ 및 Python 프로그램 코드 이해, Python 프로그램의 문제 지점 파악, 프로그램 가속화의 순서로 진행되었다.

1.2.1. 문제 및 코드 이해

본 연구는 프로그램 가속화를 목표로 하므로, 알고리즘의 자세한 설명을 생략한다. 알고리즘 개요는 [그림1]과 같다.



[그림1] 알고리즘 개요

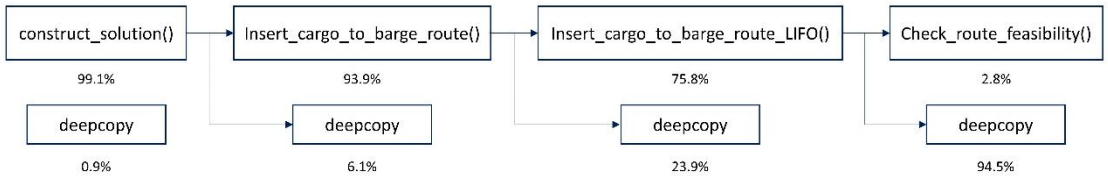
1.2.2. 프로그램 문제 지점 파악

Python의 cProfiler & line_profiler 모듈은 프로그램에 구현된 함수와 각 줄의 코드가 실행되는 시간을 출력한다. Hits는 실행횟수, Time은 총 실행시간, Per Hit은 평균 실행시간, % Time은 프로그램 총 실행시간에서 차지하는 비율을 의미한다. [표1]은 521줄, 529줄, 532줄에서 긴 시간이 소요된다는 것을 보여준다.

Line #	Hits	Time	Per Hit	% Time	Line Contents
511					def solve(self):
514	1	10667	10667	0	self.set_cargo_barge_eligibility()
515	1	24417	24417	0	self.adjust_cargo_time_windows_at_quays()
516	1	2065	2065	0	self.initialize_solution()
517	1	28012	28012	0	a_solution = deepcopy(self.solution)
518	1	18332	18332	0	a_solution.result_cargo_index = deepcopy(self.cargo_index)
519	1	193	193	0	self.uncheck_cargo(a_solution.result_cargo_index)
520	1	14714	14714	0	self.insert_pre_loaded_cargos(a_solution)
521	1	24518098	2.00E+07	8.1	**self.construct_solution(a_solution)**
523	1	7107	7107	0	a_solution.print_info()
525	31	158	5.1	0	for i in range(self.parameter.LNS_iteration):
529	30	2628252	87608.4	0.9	**cur_solution = deepcopy(a_solution)**
530	30	139097	4636.6	0	self.LNS_random_cargo_destory(cur_solution)
531	30	341512	11383.7	0.1	self.insert_pre_loaded_cargos(cur_solution)
532	30	2.74E+08	9.00E+06	90.8	**self.construct_solution(cur_solution)**
533	30	1328	44.3	0	if is_first_solution_better(cur_solution, a_solution) == True:
534	1	8169	8169	0	print("iteration:", i, " solution improved")
536	1	82736	82736	0	a_solution = deepcopy(cur_solution)
538	1	20841	20841	0	self.write_solution(a_solution)
540	1	4	4	0	return 0

[표 1] 코드 실행시간

프로그램이 실행되는 동안 가장 오래 소요되는 함수를 파악한 후, 해당 함수에서 실행시간의 가장 높은 비율을 차지하는 코드 또는 함수를 파악하는 식의 연쇄적 추적을 통해 프로그램이 느려지는 부분을 파악하였다. 추적 결과, [그림2]와 같이 deepcopy 메소드가 프로그램 실행시간의 상당 부분을 차지하였다.



[그림2] 실행시간 비율

1.2.3. deepcopy 메소드의 문제점

deepcopy 메소드(이하 'deepcopy'라 한다.)는 Python에서 제공하는 표준 라이브러리인 copy 모듈에 포함된 메소드로, deepcopy 를 사용하여 객체의 깊은 복사를 수행할 수 있다. 본 프로그램에서는 원본 객체와 독립적인 메모리 공간을 가지는 복사된 객체를 생성해야 하는 경우가 많아 깊은 복사가 필요했고, 복잡한 객체를 한 번에 복사하기 위해 deepcopy를 사용하였다. 그러나

Python의 deepcopy 메소드는 다음과 같은 비효율성을 지닌다.

1) deepcopy는 모든 객체를 재귀적으로 복사하기 때문에, 객체 구조가 깊거나 복잡할 경우 메모리 사용량과 처리 시간이 크게 증가한다.

2) deepcopy는 mutable 객체와 immutable 객체에 따라 적절한 복사 메커니즘을 결정하는데, 이때 복사되는 각 요소의 객체 유형을 검사할 때 오버헤드를 발생시킨다.

3) deepcopy는 순환 참조(객체가 자기 자신을 참조하거나 두 개 이상의 객체가 서로를 참조하는 경우)를 처리하기 위한 추가적인 로직이 필요하므로, 이에 추가적인 오버헤드가 발생된다.

1.2.4. 깊은 복사 방법에 따른 실행시간 비교

효율적인 깊은 복사 방법을 탐색하기 위해 여러 방법의 실행시간을 비교하였다.

```
from copy import deepcopy

class old_class:
    def __init__(self):
        self.blah = 'blah'

class new_class(object):
    def __init__(self):
        self.blah = 'blah'

if __name__ == '__main__':
    import copy
    from time import time

    num_times = 100000
    L = [None, 'blah', 1, 543.4532,
         ['foo'], ('bar',), {'blah': 'blah'},
         old_class(), new_class()]

    t = time()
    for i in range(num_times):
        copy.copy(L)
    print('copy.copy:', time()-t)

    t = time()
    for i in range(num_times):
        copy.deepcopy(L)
    print('copy.deepcopy:', time()-t)
```

```
t = time()
for i in range(num_times):
    L[:]
print('list slicing [:]:', time()-t)

t = time()
for i in range(num_times):
    list(L)
print('list(L):', time()-t)

t = time()
for i in range(num_times):
    [i for i in L]
print('list expression(L):', time()-t)

t = time()
for i in range(num_times):
    a = []
    a.extend(L)
print('list extend:', time()-t)

t = time()
for i in range(num_times):
    a = []
    for y in L:
        a.append(y)
print('list append:', time()-t)

t = time()
for i in range(num_times):
    a = []
    a.extend(i for i in L)
print('generator expression extend:', time()-t)
```

실행 결과는 [표2]와 같다.

복사 방법	실행시간(ms)
copy.copy	0.025
copy.deepcopy	1.062
list slicing	0.007
list(L)	0.012
list expression(L)	0.024
list extend	0.010
list append	0.055
generator expression extend	0.040

[표2] 복사 실행시간 비교

해당 프로그램을 작동시킨 컴퓨터의 사양은 다음과 같다.

MacBook Pro (13-inch, M1, 2020), Apple M1 칩(8코어 CPU, 8코어 GPU, 16코어 Neural Engine, 16GB RAM

1.2.5. 개선 방안

위의 실험을 통해 deepcopy를 사용한 깊은 복사의 처리 속도가 비교적 느리다는 것을 확인하였다. 따라서 복잡한 객체를 복사할 시에는 deepcopy 메소드를 사용하는 대신 얕은 복사 copy와 list 슬라이싱을 복합적으로 사용하여 깊은 복사를 수행하는 사용자 정의 함수를 만들거나 deepcopy에 대한 매직 메소드를 구현하여 깊은 복사를 최적화할 수 있다. [그림3]과 [그림4]는 deepcopy를 사용하는 대신 깊은 복사가 필요한 클래스 내에 깊은 복사 기능을 하는 copy() 함수를 구현한 것이다.

```python
# 한 개의 barge의 schedule을 담는 class
class BargeSol:
    def __init__(self, index, start_time, start_quay_index, end_time, end_quay_index, unit_block_weight, area_limit, stopover):
        self.check = 0  # not used
        self.index = index
        self.start_time = start_time
        self.start_quay_index = start_quay_index
        self.end_time = end_time
        self.end_quay_index = end_quay_index  # if -1, any location can be possible for the end quay
        self.total_tp_time = 0
        self.total_wait_time = 0
        self.total_duration = 0
        self.unit_block_weight = unit_block_weight
        self.area_limit = area_limit
        self.stopover = stopover
        self.element_sequence = [] # 동적 변동하므로 numpy 어려움

    def copy(self): # LSE add: deepcopy 대체
        new_copy = copy.copy(self) #immutable objects
        new_copy.element_sequence = self.element_sequence[:] #mutable objects
        return new_copy
```

[그림3] 사용자 정의 깊은 복사 함수

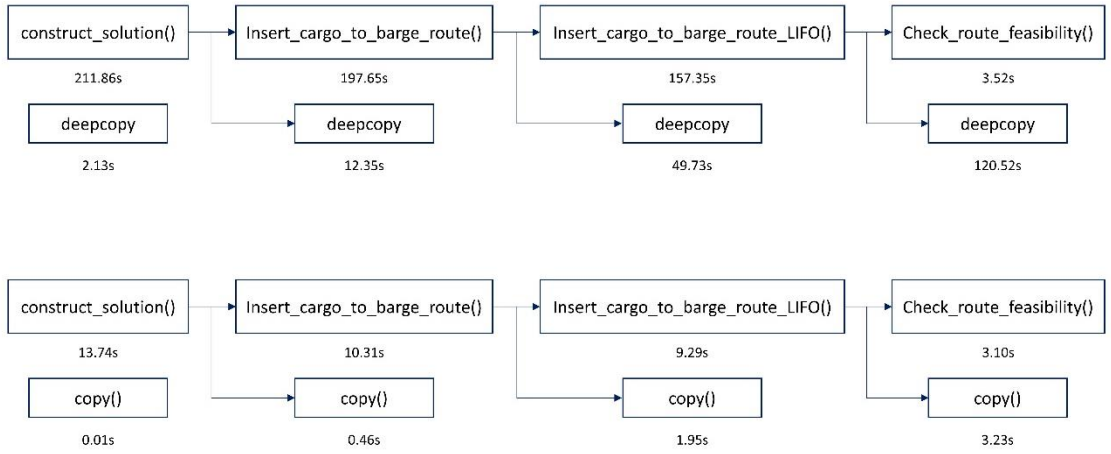
```python
# 한 개의 barge의 schedule을 담는 class
class CSolution:
    def __init__(self):
        self.barge_route_list = []  # BargeSol instance들을 담는다
        self.routed_cargo_num = 0  # 할당된 cargo 개수
        self.unrouted_cargo_num = 0  # 할당되지 못한 cargo 개수
        self.routed_barge_num = 0  # 할당된 barge 개수
        self.unrouted_barge_num = 0  # 할당되지 못한 barge 개수
        self.total_tp_time = 0  # solution 전체의 이동 시간 합
        self.total_wait_time = 0  # solution 전체의 대기 시간 합
        self.total_duration = 0  # solution 전체의 service 시간 합
        self.result_cargo_index = None  # cargo가 routing되었는지 여부가 check된 list 정보

    def copy(self):
        new_copy = copy.copy(self) #immutable objects
        new_copy.barge_route_list = [bs.copy() for bs in self.barge_route_list]
        new_copy.result_cargo_index = self.result_cargo_index[:] if self.result_cargo_index is not None else None
        return new_copy
```

[그림4] 사용자 정의 깊은 복사 함수

1.3. 연구 결과

깊은 복사를 최적화하는 함수를 구현하여 깊은 복사를 수행한 결과, [그림5]와 같이 실행시간이 짧아졌으며, [표3]과 같이 프로그램 성능이 약 10배 개선되었다. 따라서, 복잡하거나 크기가 큰 객체를 깊은 복사할 시에는 deepcopy 메소드를 그대로 사용하는 것이 아닌, 최적화된 함수를 구현하거나 매직 메소드를 활용하는 것을 권장하는 바이다.



[그림 5] 개선 결과

	C++	python(original)	python(acceleration)
Computation time (ms)	3095.00	69452.70	7464.60

[표3] 개선 결과

2. 의료영상장비 예약 알고리즘 개발

2.1. 연구 과제

본 과제는 포항세명기독병원의 의료영상장비 예약 알고리즘을 개발하는 것으로, 알고리즘이 CT 또는 MRI 촬영이 필요한 환자의 예약을 제안함으로써 영상 장비의 효율적인 가동을 도모하고, 많은 환자가 빠른 시일 내에 원하는 시간대에 촬영 받는 것을 목표로 한다.

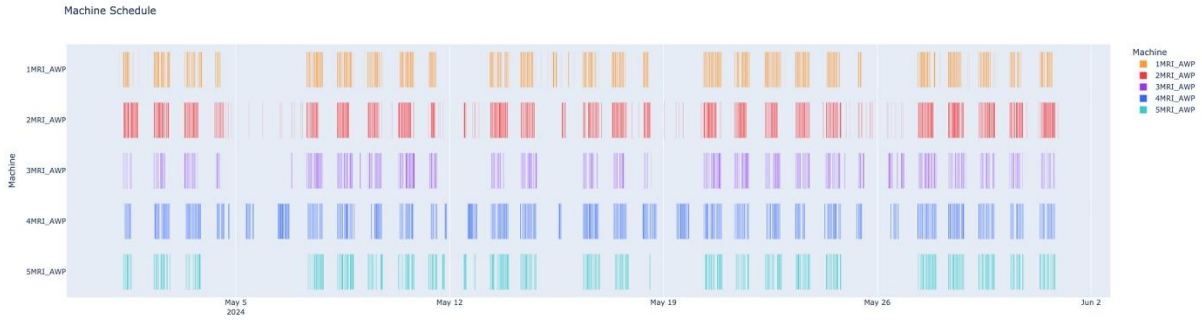
연구 참여 시작 시 본 과제는 병원의 니즈를 파악하는 초기 단계였으며, 알고리즘 개발 전 병원의 의료영상 촬영 및 예약 양상을 분석하는 것이 우선 필요했다. 따라서 본 과제에서는 병원 데이터를 분석하여 병원이 필요로 하는 인사이트를 제공하는 것을 담당하였다.

2.2. 연구 과정

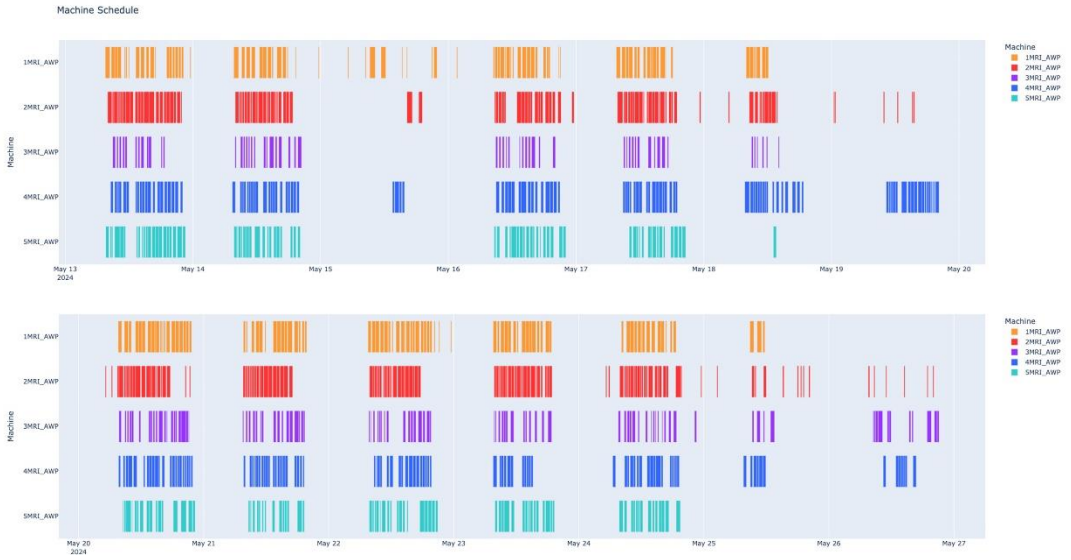
데이터 분석의 목적은 다음과 같다.

1) 간트 차트로 월간, 주간, 일간 예약 데이터를 시각화하여 예약 패턴을 한눈에 볼 수 있도록 한다.
2) 의료영상장비 촬영 시간의 기초 통계량을 계산한다.
3) 병원에서 현재 사용하는 슬롯(예상 촬영 시간)이 적절한지 판단한다.
4) 처방코드 별 적절한 슬롯 값을 얻는다.
5) 영상 장비 사용의 전반적인 흐름을 이해한다.
6) 현재 병원의 영상 장비 가동률을 계산한다.

[그림6], [그림7]은 영상 장비별 촬영 데이터를 시각화한 Gantt Chart이다.



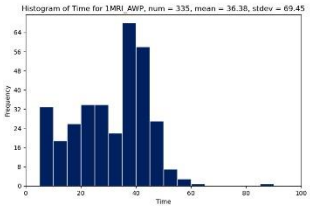
[그림6] 장비별 Gantt Chart

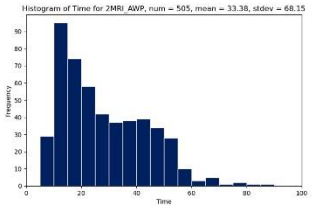


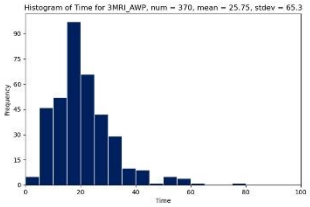
[그림 7] 장비별 Gantt Chart

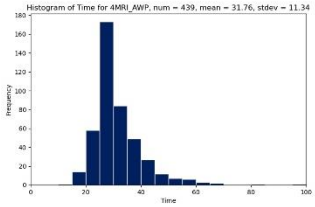
[그림8~12]는 촬영 시간의 기초 통계량 분석 결과이다. 장비별, 처방코드별, 요일별, 시간대별 촬영횟수와 촬영시간 평균, 촬영시간 표준편차, 히스토그램을 도출하였다.

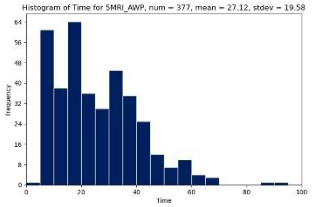
장비	촬영횟수	촬영시간 평균	촬영시간 표준편차
1MRI_AWP	335	36.38	69.45
2MRI_AWP	505	33.38	68.15
3MRI_AWP	370	25.75	65.3
4MRI_AWP	439	31.76	11.34
5MRI_AWP	377	27.12	19.58
total	2026	30.97	53.31











[그림 8] 장비별 통계량

처방코드	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMMR1F	121	21.82	9.82
RMMR1S	2	5.35	2.19
RMMR2S	3	24.27	8.16
RMMR3	1	4.3	0
RMMRPB	1	23	0
RMMRPB1	15	123.52	356.27
RMMRPK	3	21.53	7.3
RMMRPS1	2	17.15	2.9
RMMRPS2	2	15.6	3.96
RMMRPS3	1	20.7	0
RMMRPS31	1	31.6	0
RMNSMR	39	10.29	6.32
RMNSMRC	9	10.56	2.71
RMNSMRE_1	1	28.5	0
RMNSMRT	5	9.82	5.89
RMXHE2352_1	1	55.9	0
RMXHE235_1	1	42.5	0
RMXHF102E_1	1	21.4	0
RMXMAB	1	24	0
RMXMAN1	47	28.29	14.41
RMXMAN2	40	26.77	6.95
RMXMB	16	36.01	14.04
RMXMBA	103	25.92	32.75
RMXMBD_1	4	38.48	8.55
RMXMBEVW_1	4	45.02	13.74
RMXMBE_1	8	33.55	13.31
RMXMBLE_1	2	39.15	2.33
RMXMBRD_1	23	51.18	8.15
RMXMBST	233	15.7	27.77

처방코드	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMBST1	7	221.09	315.33
RMXMC	2	45.75	5.59
RMXMCBLPE_1	5	45.88	4.6
RMXMCE_1	1	43.9	0
RMXMCS	8	23.55	10.55
RMXMEL1	20	31.43	5.65
RMXMEL2	19	35.32	13.2
RMXMELE1_1	1	41.4	0
RMXMFM1	4	24.02	7.25
RMXMFM2	2	28.45	4.6
RMXMFMC2_1	1	45	0
RMXMFO	1	24.7	0
RMXMFO1	1	22.9	0
RMXMFOE2_1	1	41.5	0
RMXMFT1	15	30.91	7.6
RMXMFT2	23	28.93	7.81
RMXMH	17	25.19	7.13
RMXMHN1	33	35.86	12.06
RMXMHN2	27	35.49	12.02
RMXMHNE2_1	1	52.1	0
RMXMHS	2	51.05	8.13
RMXMHSE_1	1	45.7	0
RMXMHU2	2	97.1	98.43
RMXMHUE1_1	1	55.6	0
RMXMHUE2_1	1	36.5	0
RMXMK1	61	27.19	8.95
RMXMK2	64	27.28	11.18
RMXMKID_1	1	39.6	0
RMXMLIE_1	11	44.45	4.12
RMXMLS	94	12.91	12.03

처방코드	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMMRCP	15	32.45	4.11
RMXMNA	53	46.75	22.77
RMXMNAE_1	221	46.84	59.84
RMXMNE_1	2	89.9	42.43
RMXMORE_1	4	49.8	8.19
RMXMPADM_1	7	47.57	4.63
RMXMPAD_1	5	41.48	2.88
RMXMPE	1	26.8	0
RMXMPEE_1	3	35.73	4.2
RMXMPEOS	24	24.84	4.7
RMXMPRED_1	39	40.52	4.47
RMXMRA_1	1	38.3	0
RMXMS1	78	25.79	7.63
RMXMS2	101	26.75	10.63
RMXMSE2_1	1	54.6	0
RMXMTC	55	50.04	120.03
RMXMTCE_1	1	36.3	0
RMXMTI1	13	24.58	7.63
RMXMTI2	15	28.48	7.29
RMXMTIE_1	13	34.79	11.9
RMXMTL	139	21.5	9.18
RMXMTLE_1	6	35.15	4.52
RMXMTLS	2	38.05	30.48
RMXMTT	2	24.55	1.63
RMXMTTE_1	1	31.2	0
RMXMTTLS	25	57.43	170.12
RMXMTTLSE_1	2	41.15	12.37
RMXMW1	38	34.93	9.52
RMXMW2	42	34.94	8.24
total	2026	30.97	53.31

[그림 9] 처방코드별 통계량

처방코드	장비	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMH	1MRI_AWP	2	23.2	2.83
	2MRI_AWP	4	26.43	5.49
	4MRI_AWP	7	27.94	7.77
	5MRI_AWP	4	20.15	7.84
RMXMHN1	1MRI_AWP	6	34.22	14.03
	2MRI_AWP	6	43.25	13.62
	4MRI_AWP	18	33.97	10.5
	5MRI_AWP	3	35.67	15.1
RMXMHN2	1MRI_AWP	4	29.62	13.21
	2MRI_AWP	4	51.75	11.43
	4MRI_AWP	19	33.31	9.3
RMXMHNE2_1	2MRI_AWP	1	52.1	0
RMXMHS	2MRI_AWP	1	45.3	0
	5MRI_AWP	1	56.8	0
RMXMHSE_1	1MRI_AWP	1	45.7	0
RMXMHU2	4MRI_AWP	2	97.1	98.43
RMXMHUE1_1	4MRI_AWP	1	55.6	0
RMXMHUE2_1	4MRI_AWP	1	36.5	0
RMXMK1	1MRI_AWP	5	24.26	4.22
	2MRI_AWP	6	34.05	9.01
	3MRI_AWP	9	19.81	6.46
	4MRI_AWP	27	29.93	6.87
	5MRI_AWP	14	24.76	11.32
RMXMK2	1MRI_AWP	2	25.05	1.06
	2MRI_AWP	6	36.02	9.06
	3MRI_AWP	12	23.97	17.53
	4MRI_AWP	30	29.43	5.45
	5MRI_AWP	14	22.1	12.75

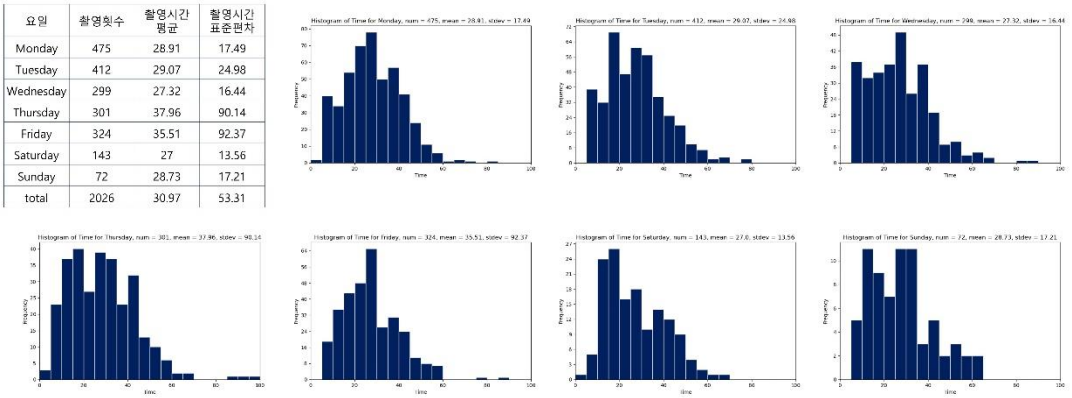
처방코드	장비	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMKID_1	1MRI_AWP	1	39.6	0
RMXMLIE_1	1MRI_AWP	11	44.45	4.12
RMXMLS	3MRI_AWP	3	14.2	8.9
	5MRI_AWP	91	12.87	12.15
RMXMMRCP	1MRI_AWP	13	32.26	4.39
	2MRI_AWP	2	33.65	1.48
RMXMNA	1MRI_AWP	15	41.13	4.46
	2MRI_AWP	16	45.74	4.54
	3MRI_AWP	4	38.05	5.24
	4MRI_AWP	3	45.33	21.03
	5MRI_AWP	15	56.07	40.79
RMXMNAE_1	1MRI_AWP	75	53.08	101.79
	2MRI_AWP	59	49.17	10.69
	3MRI_AWP	5	50.32	7.41
	4MRI_AWP	1	59.1	0
	5MRI_AWP	81	39.01	8.55
RMXMNE_1	5MRI_AWP	2	89.9	42.43
RMXMORE_1	5MRI_AWP	4	49.8	8.19
RMXMPADM_1	1MRI_AWP	7	47.57	4.63
RMXMPAD_1	1MRI_AWP	5	41.48	2.88
RMXMPE	2MRI_AWP	1	26.8	0
RMXMPEE_1	5MRI_AWP	3	35.73	4.2
RMXMPEOS	1MRI_AWP	3	27.27	3.51
	2MRI_AWP	3	22.5	2.72
	4MRI_AWP	10	28.06	3.48
	5MRI_AWP	8	20.79	3.57
RMXMPRED_1	1MRI_AWP	38	40.77	4.22
	2MRI_AWP	1	30.7	0
RMXMRA_1	4MRI_AWP	1	38.3	0

처방코드	장비	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMS1	1MRI_AWP	7	22.34	2.38
	2MRI_AWP	5	39.38	3.75
	3MRI_AWP	21	20.66	4.55
	4MRI_AWP	33	29.32	5.92
	5MRI_AWP	12	21.41	7.7
RMXMS2	1MRI_AWP	6	24.57	2.56
	2MRI_AWP	3	33.3	4.16
	3MRI_AWP	35	21.6	4.43
	4MRI_AWP	41	32.03	12.45
	5MRI_AWP	16	24.08	11.64
RMXMSE2_1	3MRI_AWP	1	54.6	0
RMXMTC	1MRI_AWP	1	40.8	0
	2MRI_AWP	2	41.4	3.11
	3MRI_AWP	46	51.72	131.38
	4MRI_AWP	4	45.03	10.14
	5MRI_AWP	2	34.7	4.95
RMXMTCE_1	3MRI_AWP	1	36.3	0
RMXMTI1	1MRI_AWP	5	22.32	2.5
	2MRI_AWP	4	29.93	11.39
	4MRI_AWP	3	24.6	4.5
	5MRI_AWP	1	14.4	0
RMXMTI2	1MRI_AWP	2	25.1	10.04
	2MRI_AWP	3	25.03	9.01
	4MRI_AWP	8	31.98	4.88
	5MRI_AWP	2	23.05	9.97
RMXMTIE_1	1MRI_AWP	3	36.73	7.43
	2MRI_AWP	1	60.4	0
	5MRI_AWP	9	31.3	10.08

처방코드	장비	촬영횟수	촬영시간 평균	촬영시간 표준편차
RMXMTL	1MRI_AWP	2	30.45	0.07
	2MRI_AWP	14	31.82	11.02
	3MRI_AWP	108	19.25	8.06
	4MRI_AWP	6	28.75	5.75
	5MRI_AWP	9	25.61	5.98
RMXMTLE_1	3MRI_AWP	5	34.76	4.94
	5MRI_AWP	1	37.1	0
RMXMTLS	3MRI_AWP	2	38.05	30.48
RMXMTT	3MRI_AWP	2	24.55	1.63
RMXMTTE_1	3MRI_AWP	1	31.2	0
RMXMTTLS	3MRI_AWP	23	59.42	177.53
	4MRI_AWP	2	34.6	7.5
RMXMTTLSE_1	2MRI_AWP	1	49.9	0
	3MRI_AWP	1	32.4	0
RMXMW1	1MRI_AWP	6	27.8	3.32
	2MRI_AWP	4	44.25	12.4
	4MRI_AWP	27	35.67	8.69
	5MRI_AWP	1	20.6	0
RMXMW2	1MRI_AWP	6	27.82	4.69
	2MRI_AWP	8	39.46	6.56
	4MRI_AWP	25	36.38	7.98
	5MRI_AWP	3	25.13	4.36
total	2026	30.97	53.31	

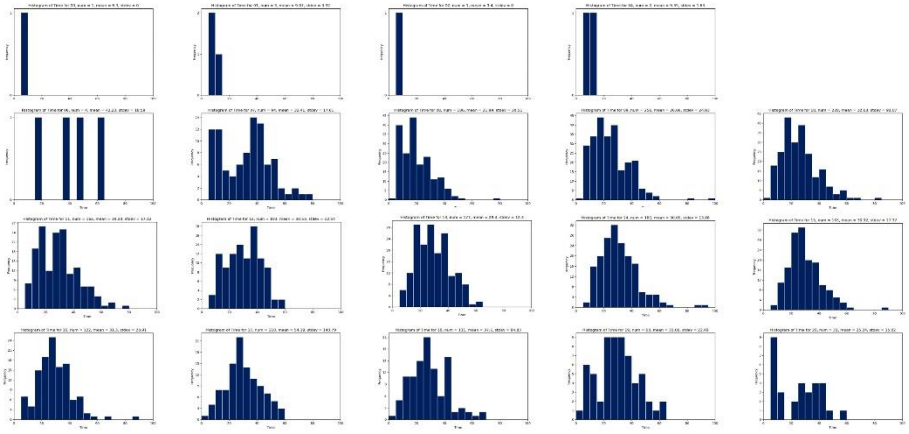
[그림 10] 처방코드-장비별 통계량

요일	촬영횟수	촬영시간 평균	촬영시간 표준편차
Monday	475	28.91	17.49
Tuesday	412	29.07	24.98
Wednesday	299	27.32	16.44
Thursday	301	37.96	90.14
Friday	324	35.51	92.37
Saturday	143	27	13.56
Sunday	72	28.73	17.21
total	2026	30.97	53.31



[그림 11] 요일별 통계량

촬영시작 시각	촬영횟수	촬영시간 평균	촬영시간 표준편차
0	1	6.3	0
1	3	9.87	3.32
2	1	9.4	0
4	2	9.35	1.63
5	0	0	0
6	4	42.23	18.59
7	94	32.41	17.61
8	186	21.99	30.51
9	256	26.66	24.93
10	228	32.63	93.07
11	162	30.23	17.12
12	103	30.53	12.54
13	177	29.4	12.3
14	180	30.65	13.88
15	163	30.92	17.37
16	122	31.3	23.91
17	110	54.19	143.79
18	111	37.7	84.83
19	63	31.66	22.49
20	31	25.24	15.02
21	15	20.61	18.58
22	9	22.04	16.01
23	5	9.72	1.69
total	2026	30.97	53.31



[그림 12] 시간대별 통계량

[그림13]은 슬롯에 대한 분석 결과이다. 각 예약에 대해 실제 촬영 시작 시각과 예약 시각, 실제 촬영 시간과 슬롯을 비교하였다. 촬영시간이 슬롯보다 짧으면 safe, 길면 over를 출력하고 초과한 시간을 함께 계산한다.

등록번호	start	시작시간 차	촬영시간	SLOT	촬영시간 <=SLOT	time_over
450995	early	46.22	47.62	60	safe	0
237330	early	26.85	0	60	safe	0
584104	early	15.6	44.72	60	safe	0
672588	early	5.95	12.32	30	safe	0
765838	early	14.85	37.82	30	over	7.82
103828	late	-14.15	43.45	60	safe	0
61186	late	-5.88	34.1	30	over	4.1
713931	late	-12.38	53.67	60	safe	0
727388	late	-6.05	44.57	60	safe	0
212209	early	3.07	43.42	60	safe	0
260723	early	10.23	44.87	60	safe	0
263402	early	25.37	46.85	60	safe	0
35534	early	4.8	26.07	60	safe	0
765888	early	0.7	46.1	60	safe	0
471257	early	6.17	43.22	60	safe	0
182490	early	7.88	42.45	60	safe	0
218917	early	25.43	47.23	60	safe	0
479327	early	25.75	39.32	60	safe	0
356454	early	11.9	42.83	60	safe	0
522697	late	-1.9	40.27	60	safe	0
65823	early	3.33	48.57	60	safe	0
145757	late	-10.38	33.85	30	over	3.85
754146	early	32.45	26.82	30	safe	0
559305	early	35.63	32.08	30	over	2.08
766124	early	50.53	24.55	30	safe	0
546708	late	-4.57	14.97	30	safe	0
360535	early	28.9	51.02	30	over	21.02
692314	early	16.08	31.32	30	over	1.32
456707	late	-3.55	30.25	30	over	0.25
543143	late	-21.18	35.18	30	over	5.18
...	...	...	...	...	...	...

early	1247	late	300
safe	1433	over	114

[그림13] 슬롯 분석

처방코드별 실제 촬영 시간과 슬롯을 비교하였다. 만족비율을 계산해 슬롯의 적절성을 판단한다[그림14].

처방코드	SLOT	촬영시간 평균	만족횟수	예약횟수	SLOT 만족비율
RHF2011	60	0	1	1	1
RMMR1F	30	17.84	97	97	1
RMMR1S	30	10.47	4	4	1
RMMR2S	30	17.93	3	3	1
RMNSMR	30	7.43	22	22	1
RMNSMRC	30	10.02	3	3	1
RMNSMRT	30	11	1	1	1
RMXHE2352_1	60	45.33	1	1	1
RMXHE235_1	60	32.15	4	4	1
RMXMABD_1	60	40.88	1	1	1
RMXMAN1	30	21.97	38	38	1
RMXMAN2	30	22.6	34	38	0.895
RMXMB	60	31.38	13	13	1
RMXMBA	60	30.11	5	5	1
RMXMBD_1	60	33.53	4	4	1
RMXMBEM_1	60	37.9	1	1	1
RMXMBEVW_1	60	50.12	1	1	1
RMXMBE_1	30	21.05	4	4	1
RMXMBRD_1	60	44.54	58	58	1
RMXMBST	30	13.04	39	39	1
RMXMBST1	60	50.12	2	2	1
RMXMCBLPE_1	60	44.1	4	4	1
RMXMCE_1	90	49.02	1	1	1
RMXMCS	30	24.9	5	5	1
RMXMEL1	30	29.25	5	12	0.417
RMXMEL2	30	29.09	7	10	0.7
RMXMF	60	48.45	1	1	1
RMXMFM1	30	25.32	2	2	1
RMXMFM2	30	29.57	1	1	1
...	...	...	...	...	...

처방코드	장비	SLOT	촬영시간 평균	만족횟수	예약횟수	SLOT 만족비율
RHF2011	2MRI_AWP	60	0	1	1	1
RMMR1F	1MRI_AWP	30	21.18	2	2	1
	2MRI_AWP	30	19.31	2	2	1
	3MRI_AWP	30	10.37	18	18	1
	4MRI_AWP	30	20.05	71	71	1
	5MRI_AWP	30	9.84	4	4	1
RMMR1S	3MRI_AWP	30	4.35	1	1	1
	4MRI_AWP	30	12.51	3	3	1
RMMR2S	3MRI_AWP	30	6.33	1	1	1
	4MRI_AWP	30	23.73	2	2	1
RMNSMR	3MRI_AWP	30	7.43	22	22	1
RMNSMRC	3MRI_AWP	30	10.02	3	3	1
RMNSMRT	3MRI_AWP	30	11	1	1	1
RMXHE2352_1	2MRI_AWP	60	45.33	1	1	1
RMXHE235_1	1MRI_AWP	60	30.2	1	1	1
	5MRI_AWP	60	32.79	3	3	1
RMXMABD_1	1MRI_AWP	60	40.88	1	1	1
RMXMAN1	2MRI_AWP	30	26.83	4	4	1
	3MRI_AWP	30	20.17	5	5	1
	4MRI_AWP	30	24.48	15	15	1
	5MRI_AWP	30	18.53	14	14	1
RMXMAN2	1MRI_AWP	30	26.33	1	1	1
	2MRI_AWP	30	30.53	2	4	0.5
	3MRI_AWP	30	15.77	1	1	1
	4MRI_AWP	30	25.56	12	13	0.923
	5MRI_AWP	60	19.06	18	19	0.947
RMXMB	1MRI_AWP	60	33.85	10	10	1
	2MRI_AWP	60	26.26	2	2	1
	5MRI_AWP	60	16.93	1	1	1
...	...	...	...	...	...	...

장비	SLOT	촬영시간 평균	만족횟수	예약횟수	SLOT 만족비율
1MRI_AWP	60	37.89	261	263	0.992
2MRI_AWP	60	36.72	220	251	0.876
3MRI_AWP	30	19.69	361	364	0.992
4MRI_AWP	30	27.03	321	389	0.825
5MRI_AWP	30	25.41	270	280	0.964

[그림14] 슬롯 적절성 판단

[그림15]는 가동률 분석 결과이다. 가동률은 병원의 의료 영상 장비 운영 시간으로 실제 촬영시간의 합을 나누어 계산하였다.

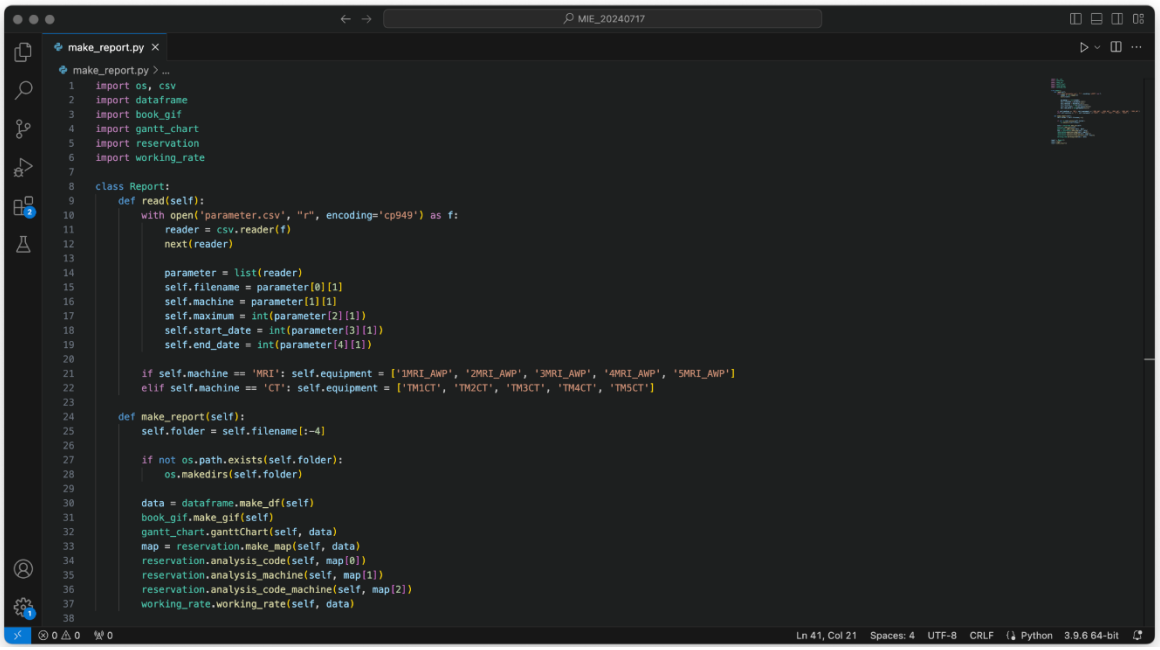
date	1MRI_AWP	2MRI_AWP	3MRI_AWP	4MRI_AWP	5MRI_AWP
20240501	0.349	0.552	0.281	0.708	0.491
20240502	0.54	0.723	0.425	0.71	0.571
20240503	0.497	0.737	0.482	0.816	0.416
20240504	0.765	0.409	0.453	0.525	0
20240505	0	0.035	0	0.634	0
20240506	0	0.079	0.832	0.65	0
20240507	0.733	0.869	0.581	0.721	0.667
20240508	0.637	0.856	0.545	0.661	0.622
20240509	0.675	0.764	0.489	0.71	0.4
20240510	0.742	0.696	0.485	0.683	0.564
20240511	0.765	0.332	0.478	0.4	0.439
20240512	0	0.129	0	0.65	0.293
20240513	0.547	0.817	0.431	0.715	0.636
20240514	0.44	0.914	0.449	0.712	0.54
20240515	0.154	0.233	0	0.89	0
20240516	0.603	0.611	0.367	0.646	0.627
20240517	0.698	0.556	0.472	0.775	0.654
20240518	0.867	0.848	0.267	0.671	1
20240519	0	0.148	0	0.726	0
20240520	0.765	0.583	0.56	0.674	0.45
20240521	0.643	0.934	0.523	0.7	0.511
20240522	0.623	0.934	0.471	0.696	0.596
20240523	0.751	0.898	0.444	0.669	0.561
20240524	0.696	0.531	0.356	0.552	0.617
20240525	0.685	0.211	0.461	0.761	0
20240526	0	0.094	0.374	0.521	0
20240527	0.496	0.825	0.514	0.696	0.696
20240528	0.528	0.746	0.456	0.774	0.685
20240529	0.47	0.791	0.406	0.651	0.697
20240530	0.557	0.666	0.36	0.56	0.559
20240531	0.653	0.613	0.375	0.589	0.569

date	1MRI_AWP	2MRI_AWP	3MRI_AWP	4MRI_AWP	5MRI_AWP
20240601	0.609	0.651	0.398	0.378	0.36
20240602	1	0.183	0	0.716	0
20240603	0.839	0.908	0.511	0.834	0.636
20240604	0.775	0.831	0.458	0.712	0.556
20240605	0.638	0.834	0.496	0.669	0.537
20240606	0.094	1	2.031	0.503	0
20240607	0.678	0.66	0.44	0.691	0.516
20240608	0.761	0.147	0.307	0.531	0
20240609	0	0.142	1	0.496	0
20240610	0.35	0.709	0.4	0.72	0.676
20240611	0.513	0.787	0.417	0.677	0.642
20240612	0.629	0.744	0.404	0.689	0.601
20240613	0.568	0.751	0.386	0.721	0.535
20240614	0.571	0.711	0.399	0.818	0.686
20240615	0.216	0.218	0.201	0.546	0.594
20240616	1	0.021	0	0.596	0
20240617	0.582	0.82	0.478	0.652	0.62
20240618	0.58	0.7	0.464	0.703	0.596
20240619	0.566	0.789	0.392	0.735	0.501
20240620	0.668	0.661	0.291	0.696	0.497
20240621	0.692	0.779	0.49	0.698	0.462
20240622	0.318	0.215	0.278	0.799	0.647
20240623	0	0.075	0.058	0.494	0
20240624	0.702	0.719	0.375	0.724	0.74
20240625	0.536	0.601	0.36	0.591	0.59
20240626	0.56	0.845	0.267	0.697	0.739
20240627	0.637	0.529	0.374	0.672	0.611
20240628	0.553	0.827	0.389	0.719	0.621
20240629	0.559	0.323	0.438	0.537	0.607
20240630	0	0.412	0	0.734	0

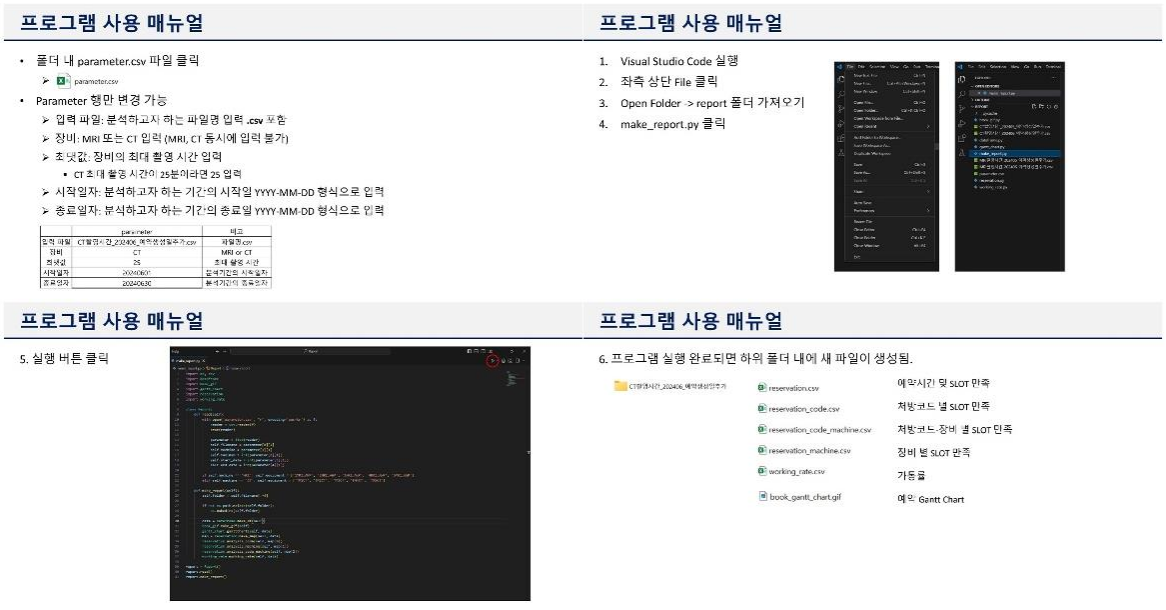
[그림 15] 가동률 분석

2.3. 연구 결과

병원 관계자와의 미팅을 통해 데이터 분석 결과가 병원의 상황을 수치로서 명료하게 나타내고, 병원이 필요한 사항을 명확히할 수 있음을 확인하였다. 따라서 병원 관계자가 원하는 데이터에 대한 분석 결과를 확인할 수 있도록 데이터 분석 자동화 프로그램을 제작하였다[그림16]. 프로그램 사용 매뉴얼을 함께 제작하여 프로그램 사용을 용이하게 했다[그림17].



[그림16] 데이터 분석 자동화 프로그램



[그림17] 프로그램 사용 매뉴얼

해당 성과는 향후 의료영상장비 예약 알고리즘의 문제를 정의하고, 알고리즘의 최적화 방향을 제시하는 역할을 할 것으로 기대된다.

3. 후기 및 느낀 점

두 개의 과제를 진행하면서 기업 관계자와 직접 소통하며 문제를 해결하는 과정을 가까이서 경험할 수 있어 값진 시간이었습니다. 현실의 문제를 풀기 위해서는 알고리즘을 개발하고 프로그램을 구현하는 것도 중요하지만, 이해관계자와 소통하고 그들의 요구를 파악하는 것 또한 어렵고 중요한 과정이라는 것을 깨달았습니다. 연구 과정에서는 단순히 주어진 일을 하는 것보다 항상 '왜'라는 질문을 던지며 능동적으로 연구하고, 비판적으로 사고해야 더 효과적인 솔루션을 제시하고 스스로의 성장을 도모할 수 있다는 것을 배웠습니다.

두 달 간의 연구참여를 통해 Python 실력을 기를 수 있었으며, 물류 분야에 대한 흥미도 높아졌습니다. 졸업 후 경쟁력 있는 연구자가 되기 위해서는 기본기를 다지는 것이 중요하다는 것을 느껴, 하나부터 열까지 원론적으로 배울 수 있는 학부 수업의 소중함을 인식하기도 하였습니다. 따라서 향후에는 이산 최적화와 데이터 구조, 객체지향 프로그래밍, 수학과 전공수업을 수강하며 기본기를 갖추고자 합니다.

두 달 동안 제가 성장할 수 있도록 도와주신 김병인 교수님과 김준 사수님, 김한솔 사수님께 감사드립니다.