Action Space 수정을 위한 필요 사항 정리

2023/2/16

천동초교입구 신호 경우의 수

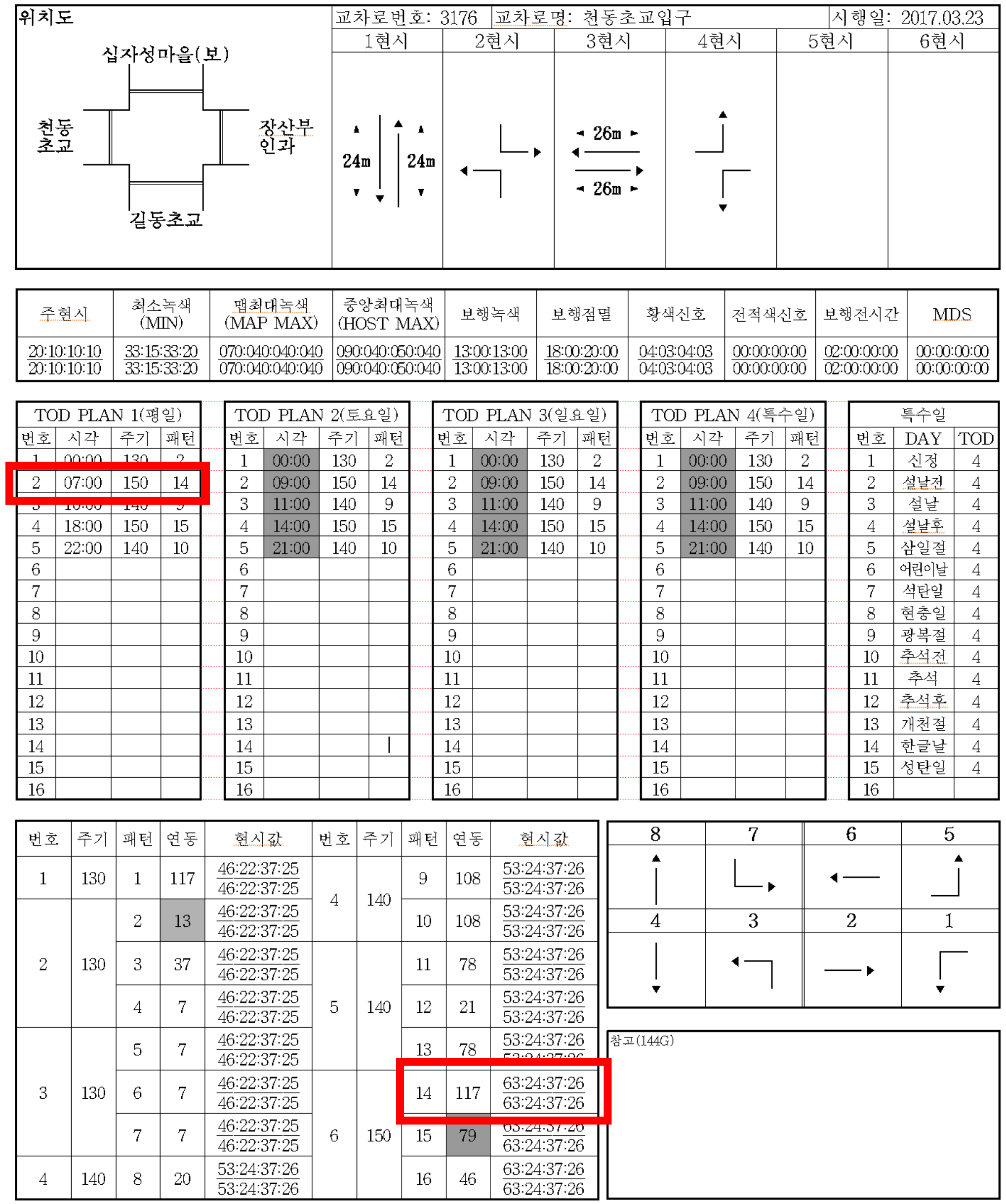
▶ 천동초교입구 신호 정보

- ❖ 최소 녹색 – 33:15:33:20
- ❖ 최대 녹색 – 70:40:40:40
- ❖ 현재 신호 – 63:24:37:26(주기 150)

▶ 최소, 최대 녹색, 현재 신호의 주기를 만족하는 경우의 수 → 3,564개

- DQN/Discrete Action
  - Output의 수가 exponential로 증가.

Source: Page 16, 신호 최적화 내용 정리 (2020/05/14)



위치도 | 교차로번호: 3176 | 교차로명: 천동초교입구 | 시행일: 2017.03.23

십자성마을(보)
천동초교
장산부인과
길동초교
1현시
2현시
3현시
4현시
5현시
6현시
24m
24m
26m
26m

주현시	최소녹색 (MIN)	맵최대녹색 (MAP MAX)	중앙최대녹색 (HOST MAX)	보행녹색	보행점멸	황색신호	전적색신호	보행전시간	MDS
20:10:10:10 / 20:10:10:10	33:15:33:20 / 33:15:33:20	070:040:040:040 / 070:040:040:040	090:040:050:040 / 090:040:050:040	13:00:13:00 / 13:00:13:00	18:00:20:00 / 18:00:20:00	04:03:04:03 / 04:03:04:03	00:00:00:00 / 00:00:00:00	02:00:00:00 / 02:00:00:00	00:00:00:00 / 00:00:00:00

TOD PLAN 1(평일)			
번호	시각	주기	패턴
1	00:00	130	2
2	07:00	150	14
3	10:00	140	9
4	18:00	150	15
5	22:00	140	10
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			

TOD PLAN 2(토요일)			
번호	시각	주기	패턴
1	00:00	130	2
2	09:00	150	14
3	11:00	140	9
4	14:00	150	15
5	21:00	140	10
6			
7			
8			
9			
10			
11			
12			
13			
14			|
15			
16			

TOD PLAN 3(일요일)			
번호	시각	주기	패턴
1	00:00	130	2
2	09:00	150	14
3	11:00	140	9
4	14:00	150	15
5	21:00	140	10
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			

TOD PLAN 4(특수일)			
번호	시각	주기	패턴
1	00:00	130	2
2	09:00	150	14
3	11:00	140	9
4	14:00	150	15
5	21:00	140	10
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			

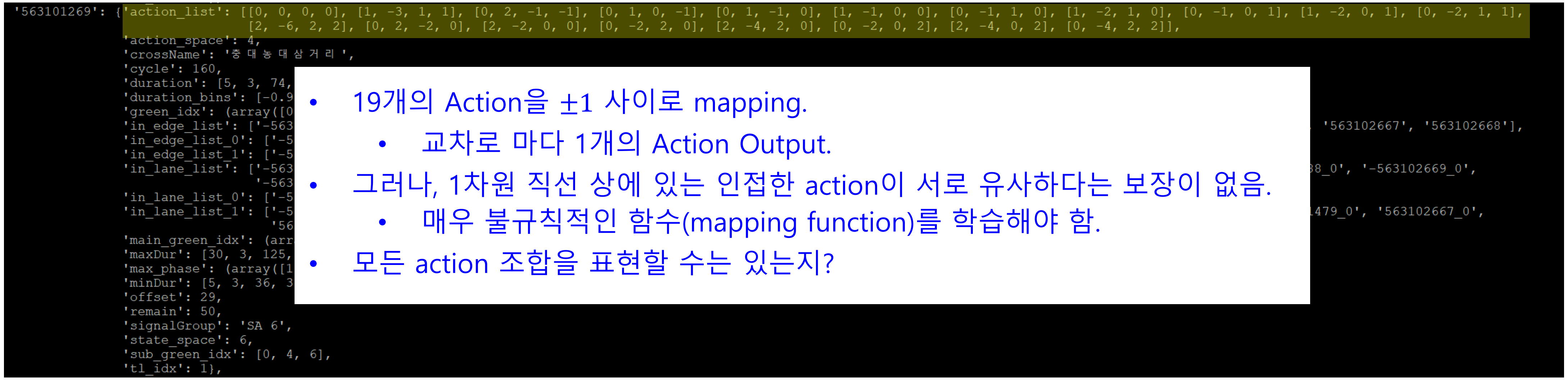
특수일		
번호	DAY	TOD
1	신정	4
2	설날전	4
3	설날	4
4	설날후	4
5	삼일절	4
6	어린이날	4
7	석탄일	4
8	현충일	4
9	광복절	4
10	추석전	4
11	추석	4
12	추석후	4
13	개천절	4
14	한글날	4
15	성탄일	4
16		

번호	주기	패턴	연동	현시값	번호	주기	패턴	연동	현시값
1	130	1	117	46:22:37:25 / 46:22:37:25	4	140	9	108	53:24:37:26 / 53:24:37:26
2	130	2	13	46:22:37:25 / 46:22:37:25			10	108	53:24:37:26 / 53:24:37:26
		3	37	46:22:37:25 / 46:22:37:25	5	140	11	78	53:24:37:26 / 53:24:37:26
		4	7	46:22:37:25 / 46:22:37:25			12	21	53:24:37:26 / 53:24:37:26
3	130	5	7	46:22:37:25 / 46:22:37:25			13	78	53:24:37:26 / 53:24:37:26
		6	7	46:22:37:25 / 46:22:37:25	6	150	14	117	63:24:37:26 / 63:24:37:26
		7	7	46:22:37:25 / 46:22:37:25			15	79	63:24:37:26 / 63:24:37:26
4	140	8	20	53:24:37:26 / 53:24:37:26			16	46	63:24:37:26 / 63:24:37:26

참고(144G)

녹색 신호 조정 offset 설명

- 현시 수와 주 현시(녹색 신호가 가장 긴 현시)를 입력으로 받아 제어 가능한 조합(action_list) 생성



```
'563101269': {'action_list': [[0, 0, 0, 0], [1, -3, 1, 1], [0, 2, -1, -1], [0, 1, 0, -1], [0, 1, -1, 0], [1, -1, 0, 0], [0, -1, 1, 0], [1, -2, 1, 0], [0, -1, 0, 1], [1, -2, 0, 1], [0, -2, 1, 1],
                              [2, -6, 2, 2], [0, 2, -2, 0], [2, -2, 0, 0], [0, -2, 2, 0], [2, -4, 2, 0], [0, -2, 0, 2], [2, -4, 0, 2], [0, -4, 2, 2]],
              'action_space': 4,
              'crossName': '충 대 농 대 삼 거 리 ',
              'cycle': 160,
              'duration': [5, 3, 74,
              'duration_bins': [-0.9
              'green_idx': (array([0
              'in_edge_list': ['-563                                                                          '563102667', '563102668'],
              'in_edge_list_0': ['-5
              'in_edge_list_1': ['-5
              'in_lane_list': ['-563                                                                          38_0', '-563102669_0',
                               '-563
              'in_lane_list_0': ['-5
              'in_lane_list_1': ['-5                                                                          1479_0', '563102667_0',
                                 '56
              'main_green_idx': (arr
              'maxDur': [30, 3, 125,
              'max_phase': (array([1
              'minDur': [5, 3, 36, 3
              'offset': 29,
              'remain': 50,
              'signalGroup': 'SA 6',
              'state_space': 6,
              'sub_green_idx': [0, 4, 6],
              'tl_idx': 1},
```

- 19개의 Action을 ±1 사이로 mapping.
  - 교차로 마다 1개의 Action Output.
- 그러나, 1차원 직선 상에 있는 인접한 action이 서로 유사하다는 보장이 없음.
  - 매우 불규칙적인 함수(mapping function)를 학습해야 함.
- 모든 action 조합을 표현할 수는 있는지?

- 충대농대삼거리 action_list 개수: 19개

- continuous action → discrete action 변환
  - -1~1을 19등분하여 매칭
  - np.digitize와 np.linspace 활용

```
np.linspace(-1, 1, $\boxed{19}$)
```

action_list 개수

```
array([-1.        , -0.88888889, -0.77777778, -0.66666667, -0.55555556,
       -0.44444444, -0.33333333, -0.22222222, -0.11111111,  0.        ,
        0.11111111,  0.22222222,  0.33333333,  0.44444444,  0.55555556,
        0.66666667,  0.77777778,  0.88888889,  1.        ])
```

continuous action

```
np.digitize(-0.983, bins=np.linspace(-1, 1, 19))
```

```
1
```

discrete action

continuous action

```
np.digitize(0.89, bins=np.linspace(-1, 1, 19))
```

```
18
```

discrete action

Source: Page 59, 신호 최적화 코드 리뷰 (2022/03/29)

Proposed I

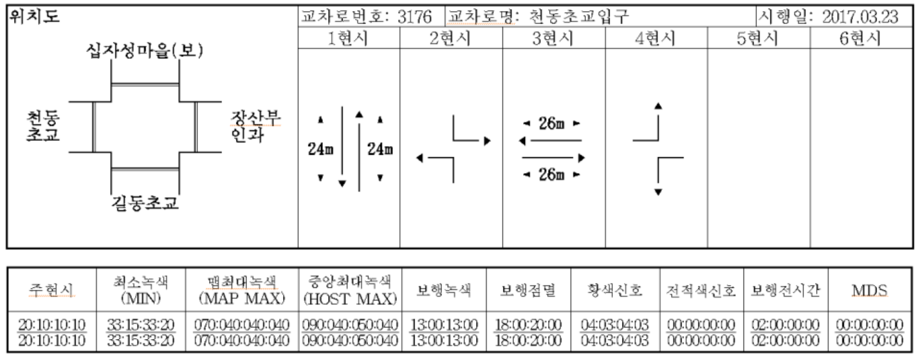
❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0
Action	$-1 \le a_1 \le +1$	$-1 \le a_2 \le +1$	$-1 \le a_3 \le +1$	$-1 \le a_4 \le +1$	$101 \le T \le 190$

$$T = (51.5 + a_1 \times 18.5) + (27.5 + a_2 \times 12.5) + (36.5 + a_3 \times 3.5) + (20.0 + a_4 \times 10.0)$$

- For each intersection, #Action outputs = #Phases
  - 신호 조합이 아닌, #Phases에 선형 비례로 증가.
- Action space를 4차원 vector로 표현
  - 모든 action 조합을 표현할 수 있음.
  - 비슷한 action vector는 실제로도 유사한 제어 신호임.
- 제약 조건
  - 최소/최대 녹색 시간 만족
  - 주기 (T=150)는 만족하지 않음.
    - Penalty 추가 (추후 고려)

천동초교입구



위치도	교차로번호: 3176	교차로명: 천동초교입구				시행일: 2017.03.23
	1현시	2현시	3현시	4현시	5현시	6현시

십자성마을(보)
천동초교
장산부인과
길동초교
24m
24m
26m
26m

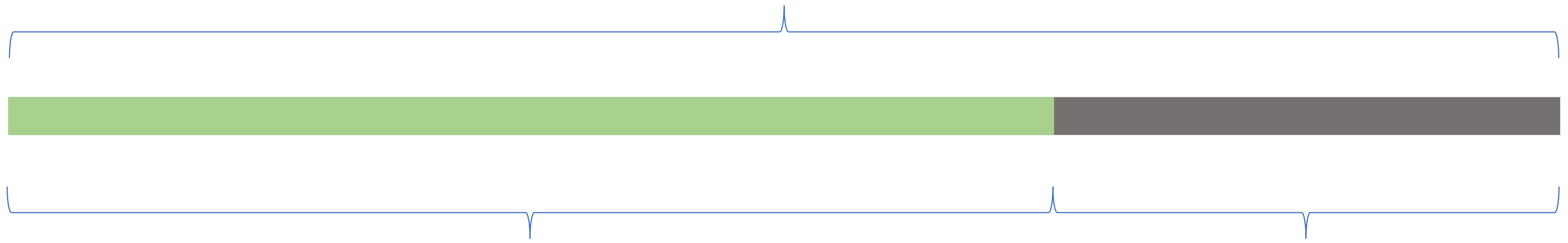
주현시	최소녹색 (MIN)	맵최대녹색 (MAP MAX)	중앙최대녹색 (HOST MAX)	보행녹색	보행점멸	황색신호	전적색신호	보행전시간	MDS
20:10:10:10 20:10:10:10	33:15:33:20 33:15:33:20	070:040:040:040 070:040:040:040	090:040:050:040 090:040:050:040	13:00:13:00 13:00:13:00	18:00:20:00 18:00:20:00	04:03:04:03 04:03:04:03	00:00:00:00 00:00:00:00	02:00:00:00 02:00:00:00	00:00:00:00 00:00:00:00

Proposed II

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

$$T = 150$$



$$Min = 33 + 15 + 33 + 20 = 101$$

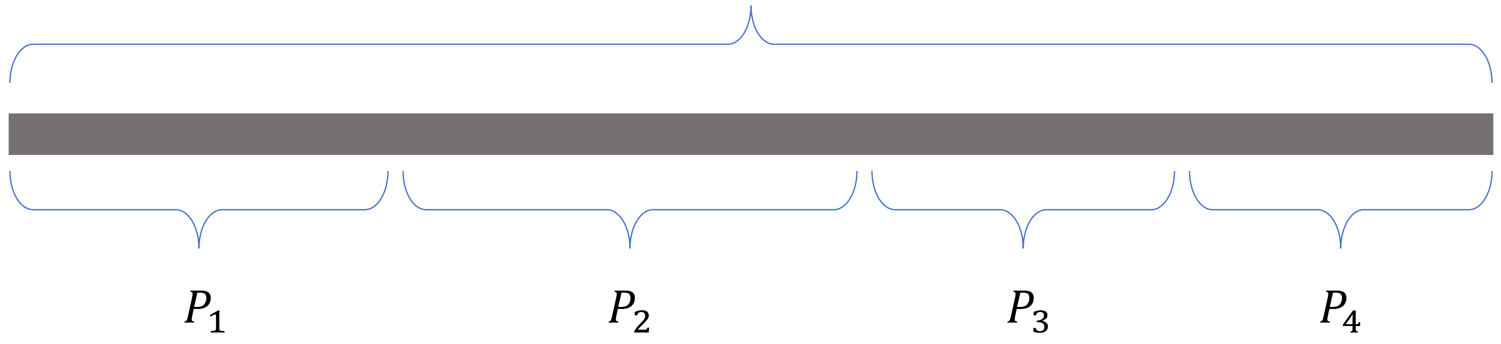
$$R = T - Min = 49$$

Proposed II

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

$$R = T - Min = 49$$

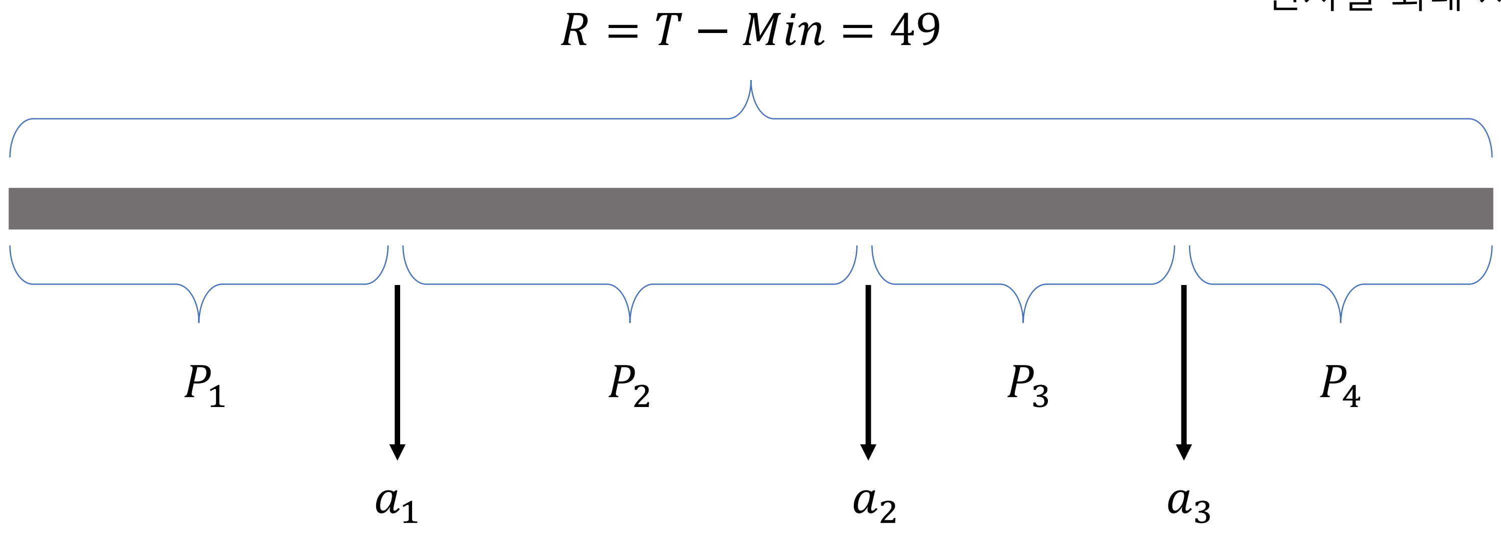


Proposed II

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

- 제약 조건
  - 최소, 주기 (T=150)는 만족
  - 현시별 최대 시간은 만족하지 않음.

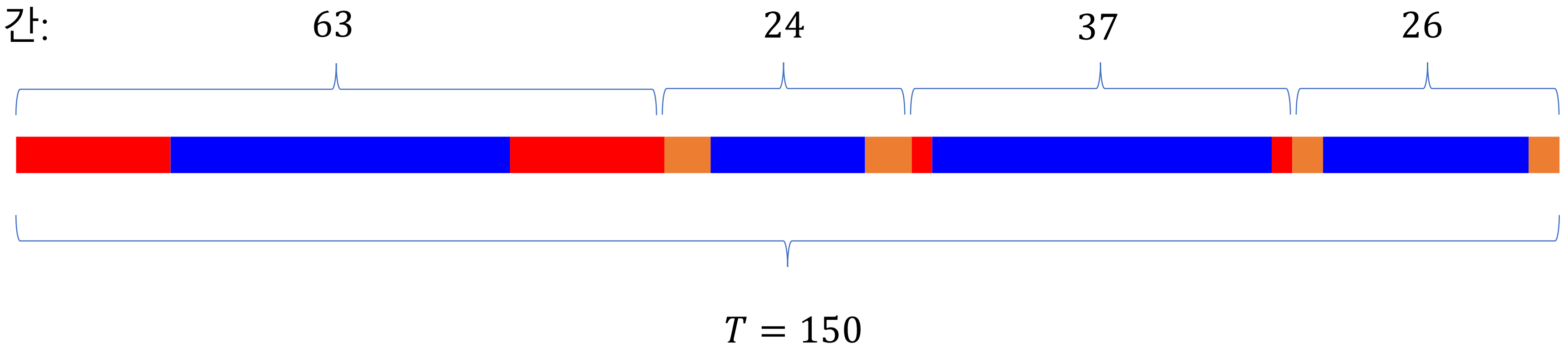


Proposed III

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

현재 현시 시간:

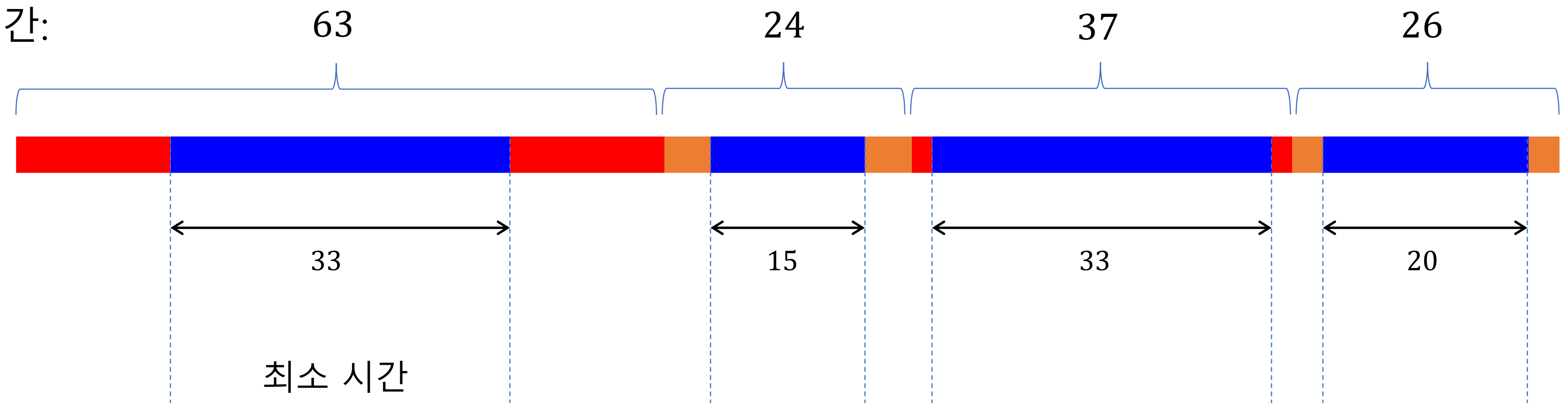


Proposed III

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

현재 현시 시간:

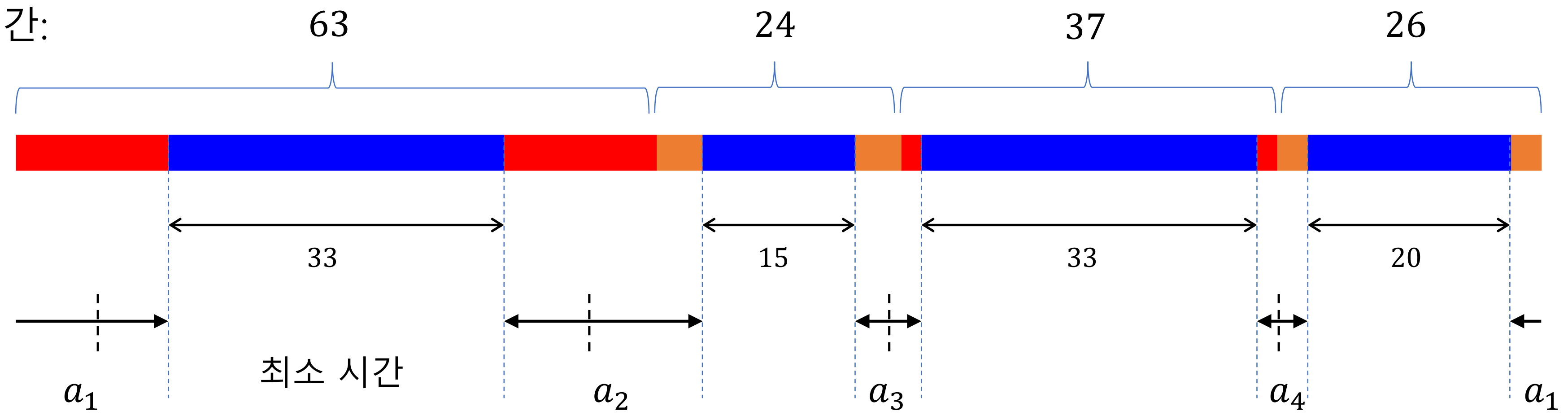


Proposed III

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

현재 현시 시간:

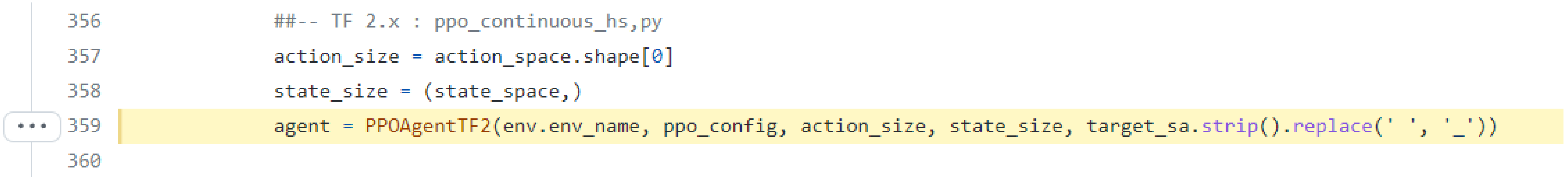


현재 신호 시간에서 최소 시간을 보장하고, 나머지 시간을 배분

수정해야할 부분

run.py

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/8ac45baeedda25f78d5343223ac2edee7aa44ee8/atsc-rl/multiagent_tf2/run.py#L359



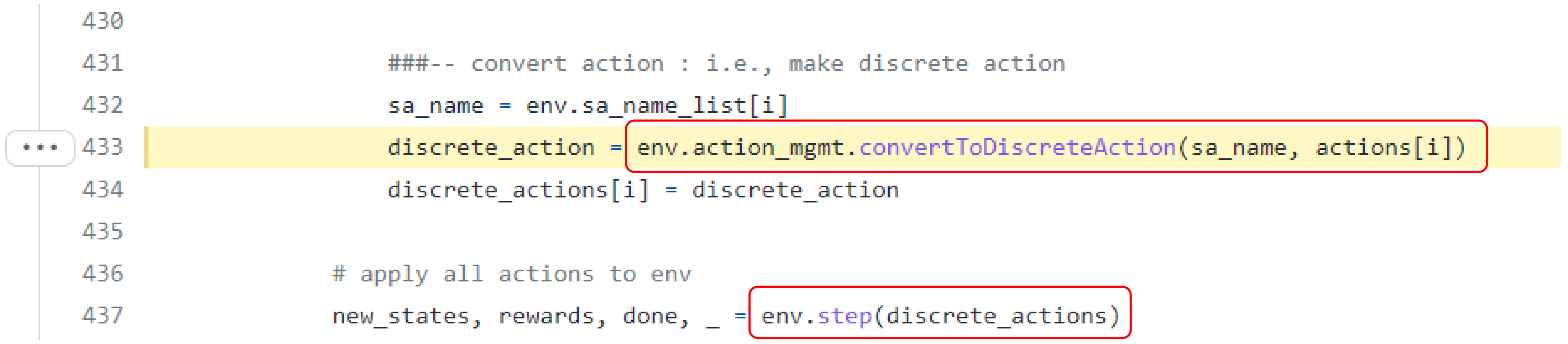
```
            ##-- TF 2.x : ppo_continuous_hs,py
            action_size = action_space.shape[0]
            state_size = (state_space,)
            agent = PPOAgentTF2(env.env_name, ppo_config, action_size, state_size, target_sa.strip().replace(' ', '_'))

```

- Proposed I으로 우선 진행
- action_size 결정을 위한 정보
  - Total number of phases in section.
  - 각 교차로마다의 현시 개수

수정해야할 부분

run.py

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/8ac45baeedda25f78d5343223ac2edee7aa44ee8/atsc-rl/multiagent_tf2/run.py#L433



```
                    ###-- convert action : i.e., make discrete action
                    sa_name = env.sa_name_list[i]
                    discrete_action = env.action_mgmt.convertToDiscreteAction(sa_name, actions[i])
                    discrete_actions[i] = discrete_action

                # apply all actions to env
                new_states, rewards, done, _ = env.step(discrete_actions)
```

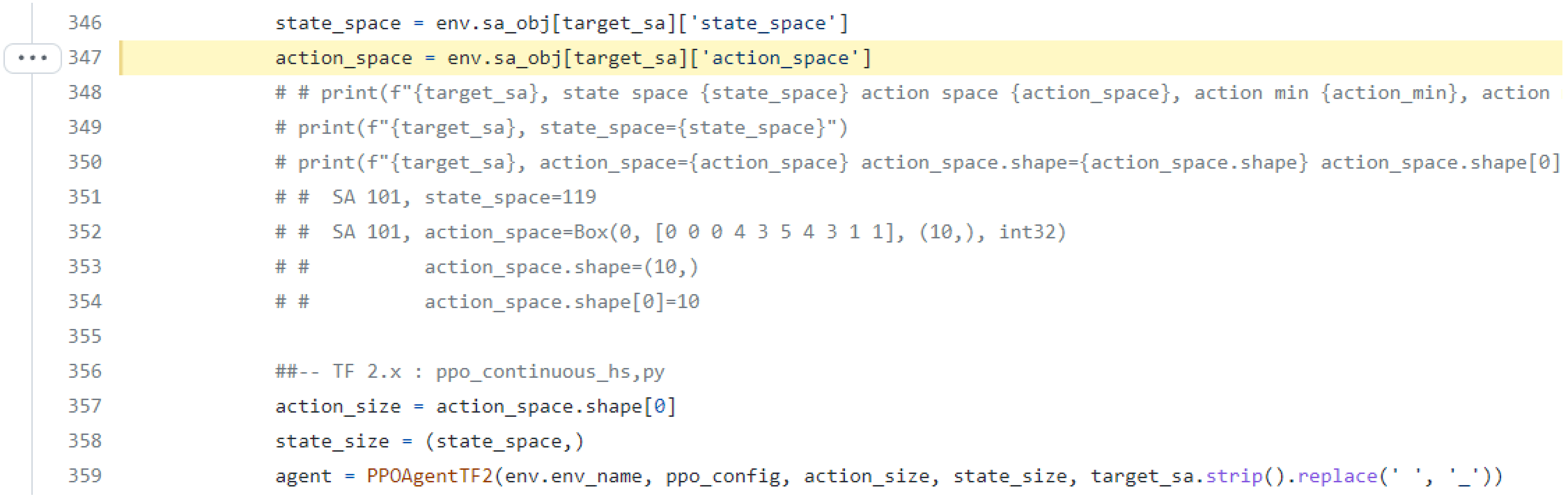
변경된 action이 적용될 수 있도록 함수 수정.

- convertToDiscreteAction()
- Step()

수정해야할 부분

run.py

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/8ac45baeedda25f78d5343223ac2edee7aa44ee8/atsc-rl/multiagent_tf2/run.py#L347



```
        state_space = env.sa_obj[target_sa]['state_space']
        action_space = env.sa_obj[target_sa]['action_space']
        # # print(f"{target_sa}, state space {state_space} action space {action_space}, action min {action_min}, action
        # print(f"{target_sa}, state_space={state_space}")
        # print(f"{target_sa}, action_space={action_space} action_space.shape={action_space.shape} action_space.shape[0]
        # #  SA 101, state_space=119
        # #  SA 101, action_space=Box(0, [0 0 0 4 3 5 4 3 1 1], (10,), int32)
        # #          action_space.shape=(10,)
        # #          action_space.shape[0]=10

        ##-- TF 2.x : ppo_continuous_hs,py
        action_size = action_space.shape[0]
        state_size = (state_space,)
        agent = PPOAgentTF2(env.env_name, ppo_config, action_size, state_size, target_sa.strip().replace(' ', '_'))
```

해당 값과 함수는 SaltSappoEnvV3 class에서 관리되고 있음.

- 'gr, offset, gro, kc' 외에 option을 추가하여, 해당 기능을 지원.

Q & A

Action Space 코드 수정

2023/3/8

Proposed I

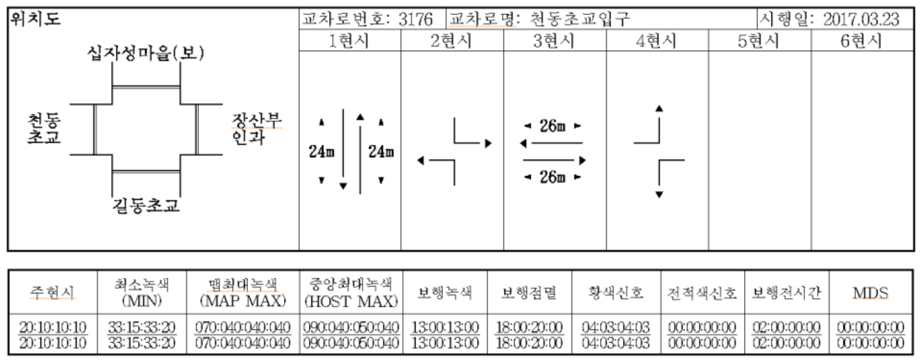
❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0
Action	$-1 \leq a_1 \leq +1$	$-1 \leq a_2 \leq +1$	$-1 \leq a_3 \leq +1$	$-1 \leq a_4 \leq +1$	$101 \leq T \leq 190$

$$T = (51.5 + a_1 \times 18.5) + (27.5 + a_2 \times 12.5) + (36.5 + a_3 \times 3.5) + (20.0 + a_4 \times 10.0)$$

- For each intersection, #Action outputs = #Phases
  - 신호 조합이 아닌, #Phases에 선형 비례로 증가.
- Action space를 4차원 vector로 표현
  - 모든 action 조합을 표현할 수 있음.
  - 비슷한 action vector는 실제로도 유사한 제어 신호임.
- 제약 조건
  - 최소/최대 녹색 시간 만족
  - 주기 (T=150)는 만족하지 않음.
    - Penalty 추가 (추후 고려)

천동초교입구



위치도

교차로번호: 3176 | 교차로명: 천동초교입구 | 시행일: 2017.03.23

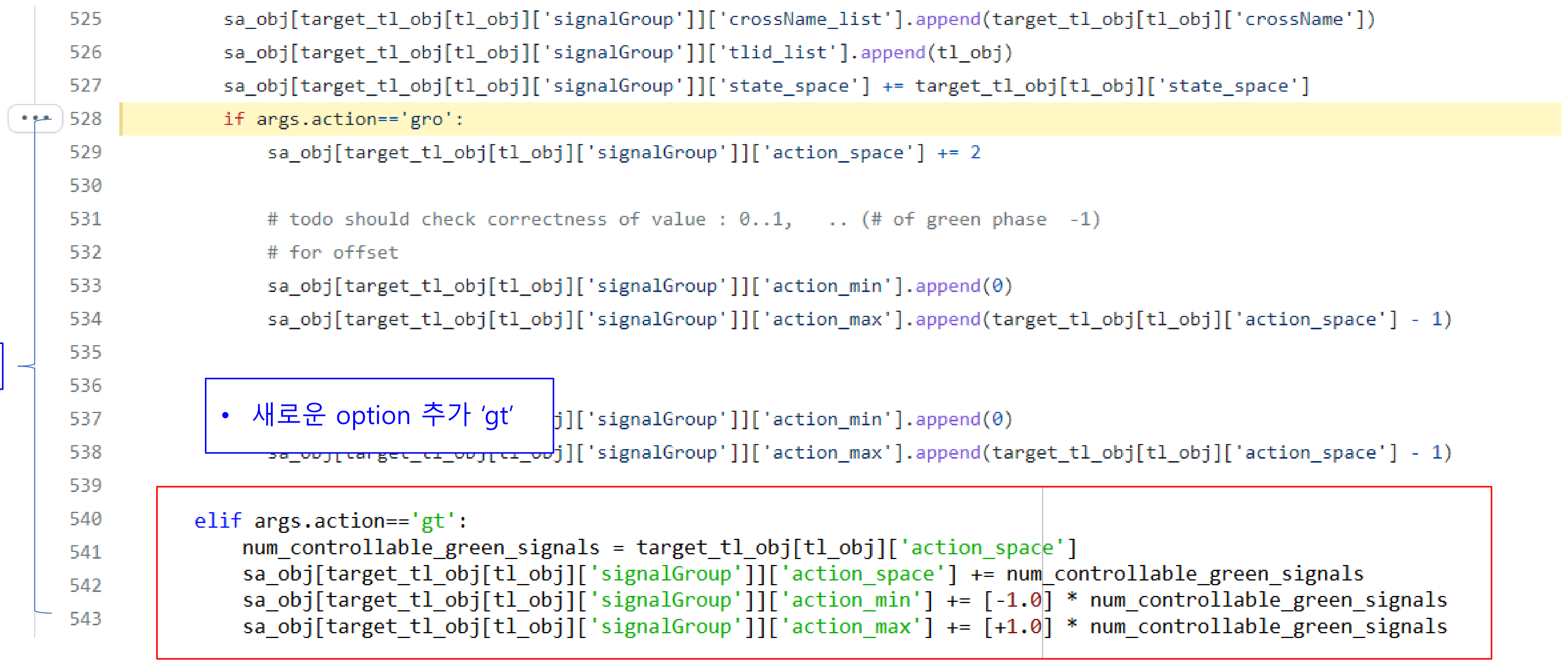
십자성마을(보)
천동초교
장산부인과
길동초교
1현시
2현시
3현시
4현시
5현시
6현시
24m
24m
26m
26m

주현시	최소녹색 (MIN)	맵최대녹색 (MAP MAX)	중앙최대녹색 (HOST MAX)	보행녹색	보행점멸	황색신호	전적색신호	보행전시간	MDS
20:10:10:10	33:15:33:20	070:040:040:040	090:040:050:040	13:00:13:00	18:00:20:00	04:03:04:03	00:00:00:00	02:00:00:00	00:00:00:00
20:10:10:10	33:15:33:20	070:040:040:040	090:040:050:040	13:00:13:00	18:00:20:00	04:03:04:03	00:00:00:00	02:00:00:00	00:00:00:00

SaltEnvUtil.py

getSaRelatedInfo(). sa_obj 생성 코드

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/master/atsc-rl/multiagent_tf2/env/SaltEnvUtil.py#L528



```
        sa_obj[target_tl_obj[tl_obj]['signalGroup']]['crossName_list'].append(target_tl_obj[tl_obj]['crossName'])
        sa_obj[target_tl_obj[tl_obj]['signalGroup']]['tlid_list'].append(tl_obj)
        sa_obj[target_tl_obj[tl_obj]['signalGroup']]['state_space'] += target_tl_obj[tl_obj]['state_space']
        if args.action=='gro':
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_space'] += 2

            # todo should check correctness of value : 0..1,   .. (# of green phase  -1)
            # for offset
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_min'].append(0)
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_max'].append(target_tl_obj[tl_obj]['action_space'] - 1)


                                                  j]['signalGroup']]['action_min'].append(0)
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_max'].append(target_tl_obj[tl_obj]['action_space'] - 1)

        elif args.action=='gt':
            num_controllable_green_signals = target_tl_obj[tl_obj]['action_space']
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_space'] += num_controllable_green_signals
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_min'] += [-1.0] * num_controllable_green_signals
            sa_obj[target_tl_obj[tl_obj]['signalGroup']]['action_max'] += [+1.0] * num_controllable_green_signals
```

Action space 계산

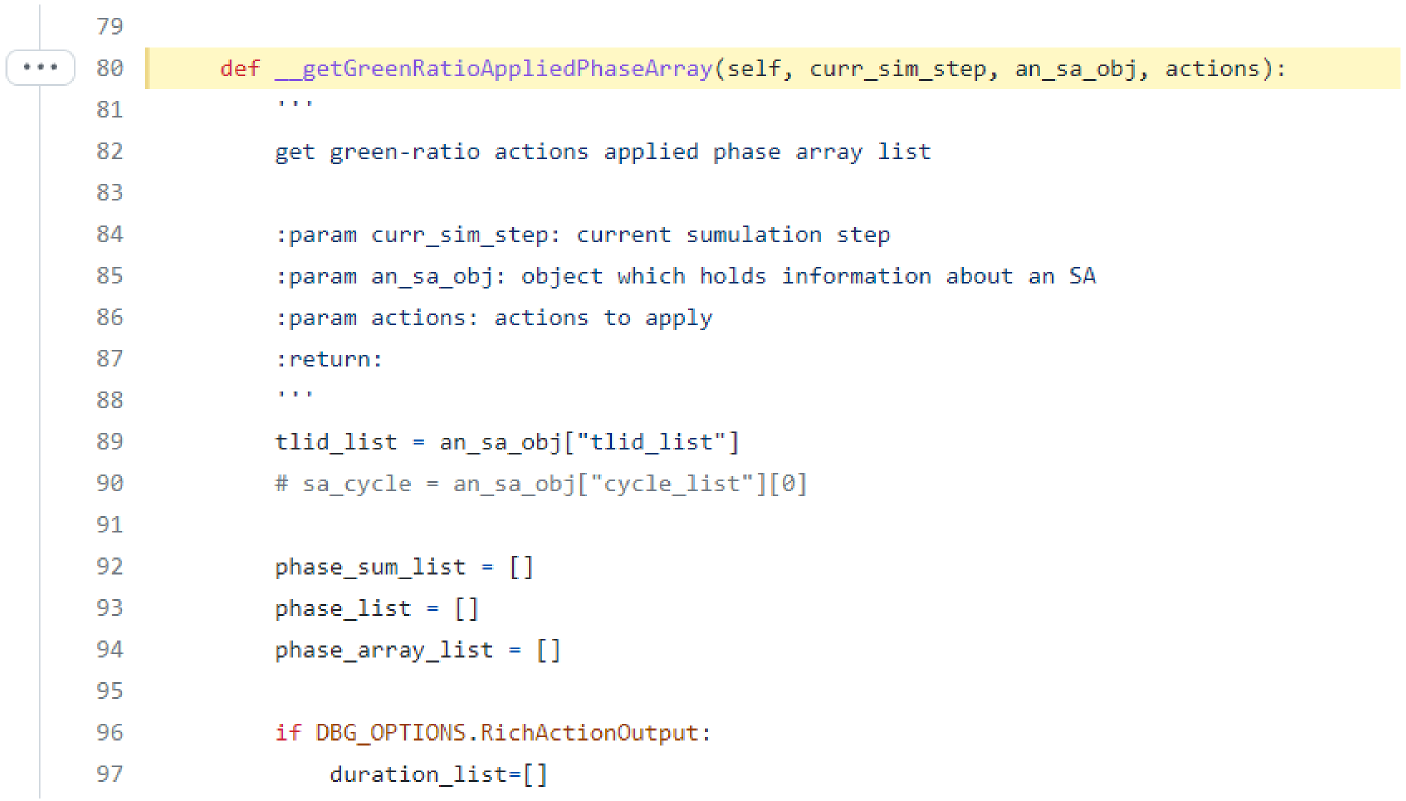
- 새로운 option 추가 'gt'

SappoActionMgmt.py #0

ETRI

__getGreenRatioAppliedPhaseArray()를 바탕으로 수정

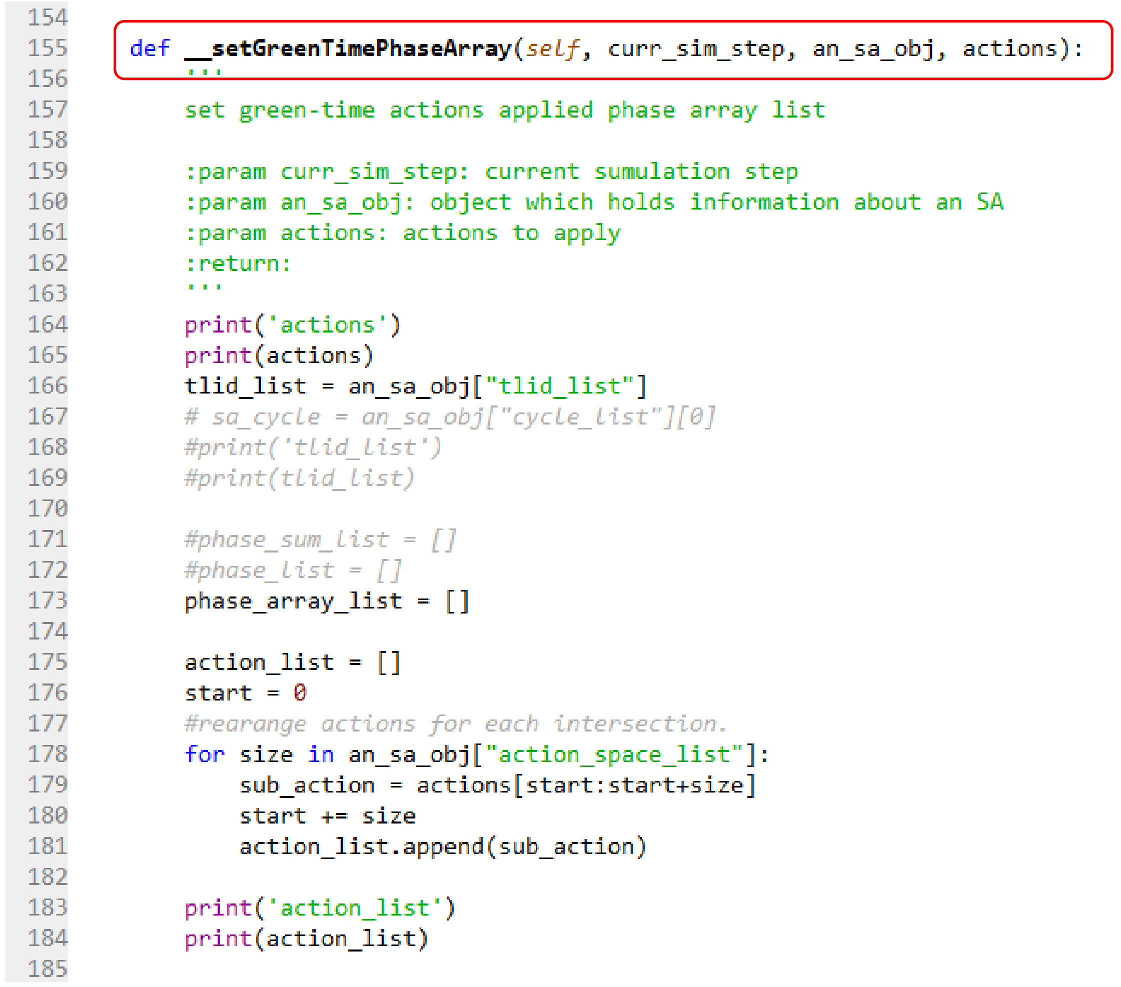
https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/a9038816e570983a604ce14a272ba373d033acc8/atsc-rl/multiagent_tf2/env/SappoActionMgmt.py#L80



```python

    def __getGreenRatioAppliedPhaseArray(self, curr_sim_step, an_sa_obj, actions):
        '''
        get green-ratio actions applied phase array list

        :param curr_sim_step: current sumulation step
        :param an_sa_obj: object which holds information about an SA
        :param actions: actions to apply
        :return:
        '''
        tlid_list = an_sa_obj["tlid_list"]
        # sa_cycle = an_sa_obj["cycle_list"][0]

        phase_sum_list = []
        phase_list = []
        phase_array_list = []

        if DBG_OPTIONS.RichActionOutput:
            duration_list=[]
```

SappoActionMgmt.py #1

__setGreenTimePhaseArray()

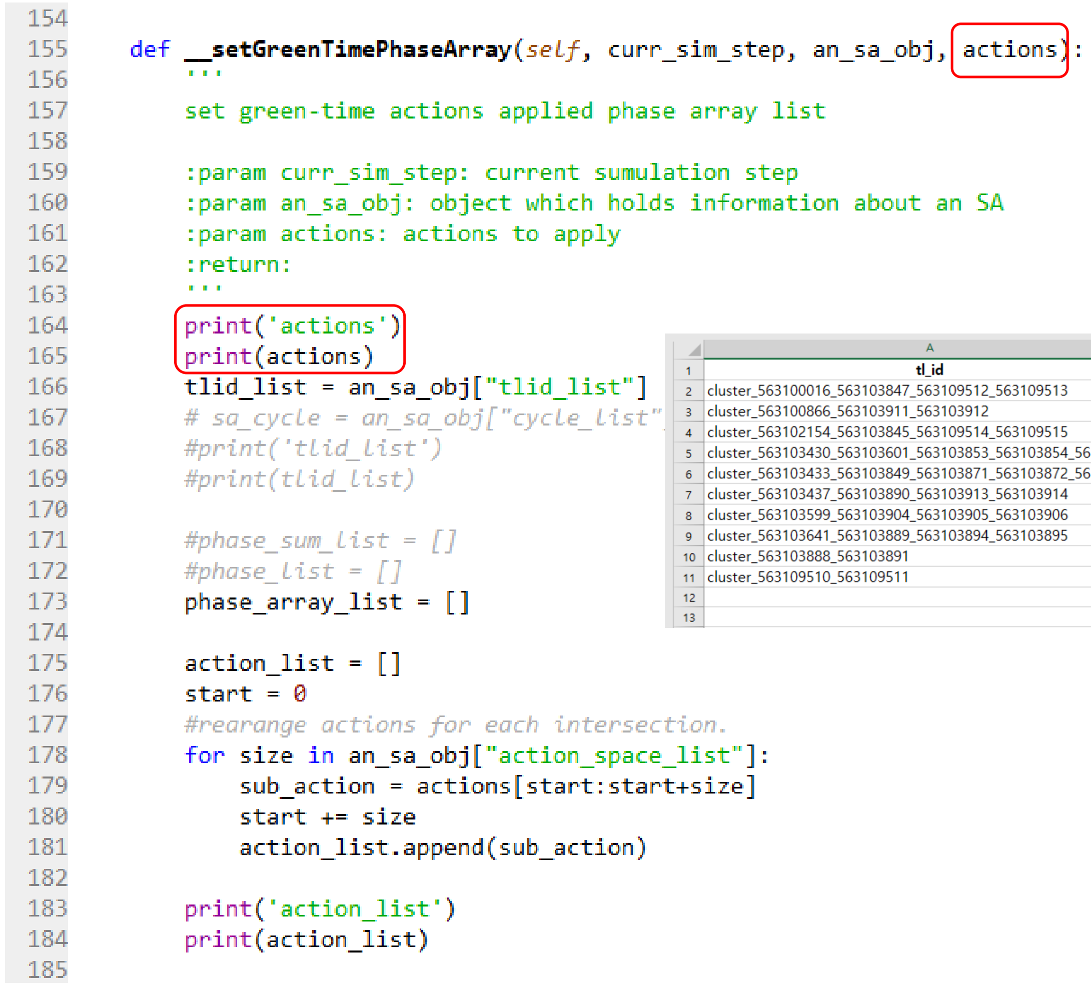


```
def __setGreenTimePhaseArray(self, curr_sim_step, an_sa_obj, actions):
    '''
    set green-time actions applied phase array list

    :param curr_sim_step: current sumulation step
    :param an_sa_obj: object which holds information about an SA
    :param actions: actions to apply
    :return:
    '''
    print('actions')
    print(actions)
    tlid_list = an_sa_obj["tlid_list"]
    # sa_cycle = an_sa_obj["cycle_list"][0]
    #print('tlid_list')
    #print(tlid_list)

    #phase_sum_list = []
    #phase_list = []
    phase_array_list = []

    action_list = []
    start = 0
    #rearange actions for each intersection.
    for size in an_sa_obj["action_space_list"]:
        sub_action = actions[start:start+size]
        start += size
        action_list.append(sub_action)

    print('action_list')
    print(action_list)

```

- 함수 추가

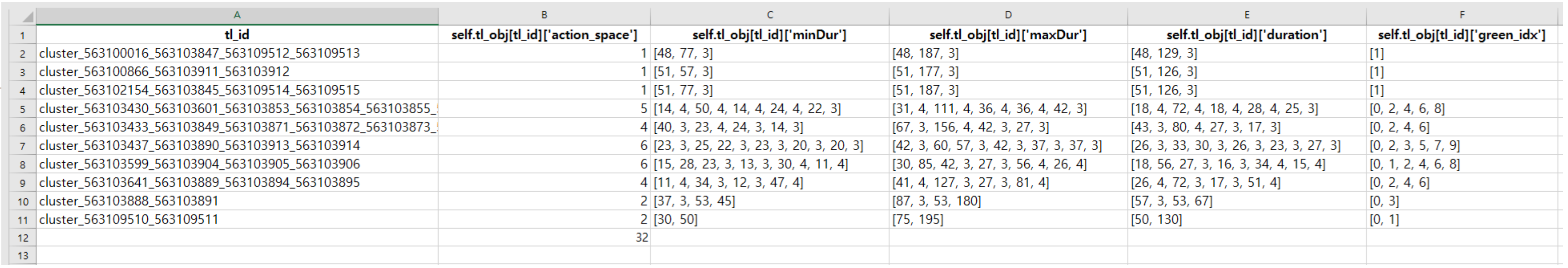
SappoActionMgmt.py #2

__setGreenTimePhaseArray()



```python
    def __setGreenTimePhaseArray(self, curr_sim_step, an_sa_obj, actions):
        '''
        set green-time actions applied phase array list

        :param curr_sim_step: current sumulation step
        :param an_sa_obj: object which holds information about an SA
        :param actions: actions to apply
        :return:
        '''
        print('actions')
        print(actions)
        tlid_list = an_sa_obj["tlid_list"]
        # sa_cycle = an_sa_obj["cycle_list"
        #print('tlid_list')
        #print(tlid_list)

        #phase_sum_list = []
        #phase_list = []
        phase_array_list = []

        action_list = []
        start = 0
        #rearange actions for each intersection.
        for size in an_sa_obj["action_space_list"]:
            sub_action = actions[start:start+size]
            start += size
            action_list.append(sub_action)

        print('action_list')
        print(action_list)
```

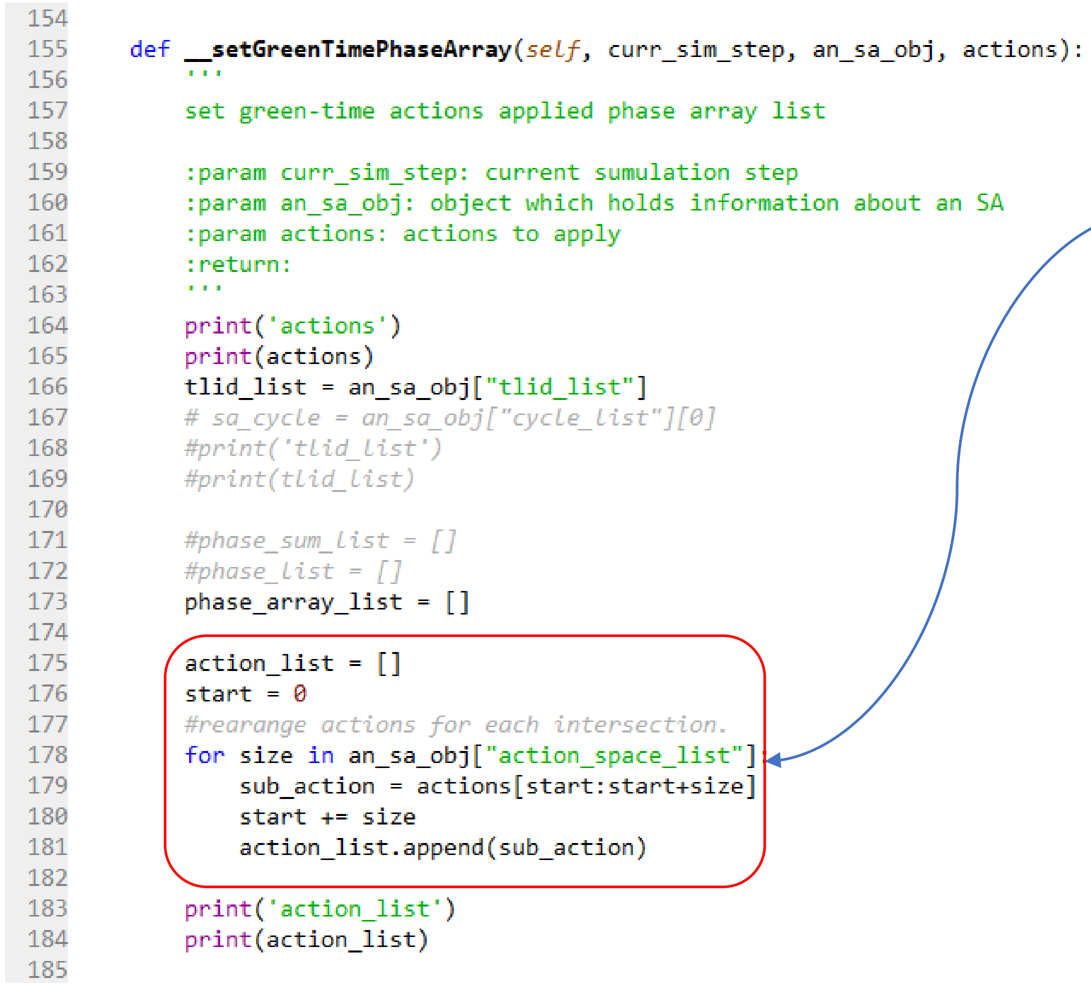
- Dim. Of actions = 32
  - SA101의 경우, 10개의 교차로가 있으며,
  - 총 32개의 조절가능한 녹색 신호가 있음.



	A	B	C	D	E	F
1	tl_id	self.tl_obj[tl_id]['action_space']	self.tl_obj[tl_id]['minDur']	self.tl_obj[tl_id]['maxDur']	self.tl_obj[tl_id]['duration']	self.tl_obj[tl_id]['green_idx']
2	cluster_563100016_563103847_563109512_563109513	1	[48, 77, 3]	[48, 187, 3]	[48, 129, 3]	[1]
3	cluster_563100866_563103911_563103912	1	[51, 57, 3]	[51, 177, 3]	[51, 126, 3]	[1]
4	cluster_563102154_563103845_563109514_563109515	1	[51, 77, 3]	[51, 187, 3]	[51, 126, 3]	[1]
5	cluster_563103430_563103601_563103853_563103854_563103855_	5	[14, 4, 50, 4, 14, 4, 24, 4, 22, 3]	[31, 4, 111, 4, 36, 4, 36, 4, 42, 3]	[18, 4, 72, 4, 18, 4, 28, 4, 25, 3]	[0, 2, 4, 6, 8]
6	cluster_563103433_563103849_563103871_563103872_563103873_	4	[40, 3, 23, 4, 24, 3, 14, 3]	[67, 3, 156, 4, 42, 3, 27, 3]	[43, 3, 80, 4, 27, 3, 17, 3]	[0, 2, 4, 6]
7	cluster_563103437_563103890_563103913_563103914	6	[23, 3, 25, 22, 3, 23, 3, 20, 3, 20, 3]	[42, 3, 60, 57, 3, 42, 3, 37, 3, 37, 3]	[26, 3, 33, 30, 3, 26, 3, 23, 3, 27, 3]	[0, 2, 3, 5, 7, 9]
8	cluster_563103599_563103904_563103905_563103906	6	[15, 28, 23, 3, 13, 3, 30, 4, 11, 4]	[30, 85, 42, 3, 27, 3, 56, 4, 26, 4]	[18, 56, 27, 3, 16, 3, 34, 4, 15, 4]	[0, 1, 2, 4, 6, 8]
9	cluster_563103641_563103889_563103894_563103895	4	[11, 4, 34, 3, 12, 3, 47, 4]	[41, 4, 127, 3, 27, 3, 81, 4]	[26, 4, 72, 3, 17, 3, 51, 4]	[0, 2, 4, 6]
10	cluster_563103888_563103891	2	[37, 3, 53, 45]	[87, 3, 53, 180]	[57, 3, 53, 67]	[0, 3]
11	cluster_563109510_563109511	2	[30, 50]	[75, 195]	[50, 130]	[0, 1]
12		32				
13						

SappoActionMgmt.py #3

__setGreenTimePhaseArray()



```python
    def __setGreenTimePhaseArray(self, curr_sim_step, an_sa_obj, actions):
        '''
        set green-time actions applied phase array list

        :param curr_sim_step: current sumulation step
        :param an_sa_obj: object which holds information about an SA
        :param actions: actions to apply
        :return:
        '''
        print('actions')
        print(actions)
        tlid_list = an_sa_obj["tlid_list"]
        # sa_cycle = an_sa_obj["cycle_list"][0]
        #print('tlid_list')
        #print(tlid_list)

        #phase_sum_list = []
        #phase_list = []
        phase_array_list = []

        action_list = []
        start = 0
        #rearange actions for each intersection.
        for size in an_sa_obj["action_space_list"]:
            sub_action = actions[start:start+size]
            start += size
            action_list.append(sub_action)

        print('action_list')
        print(action_list)

```

- action_list
  - 각 교차로별로 action을 재정리

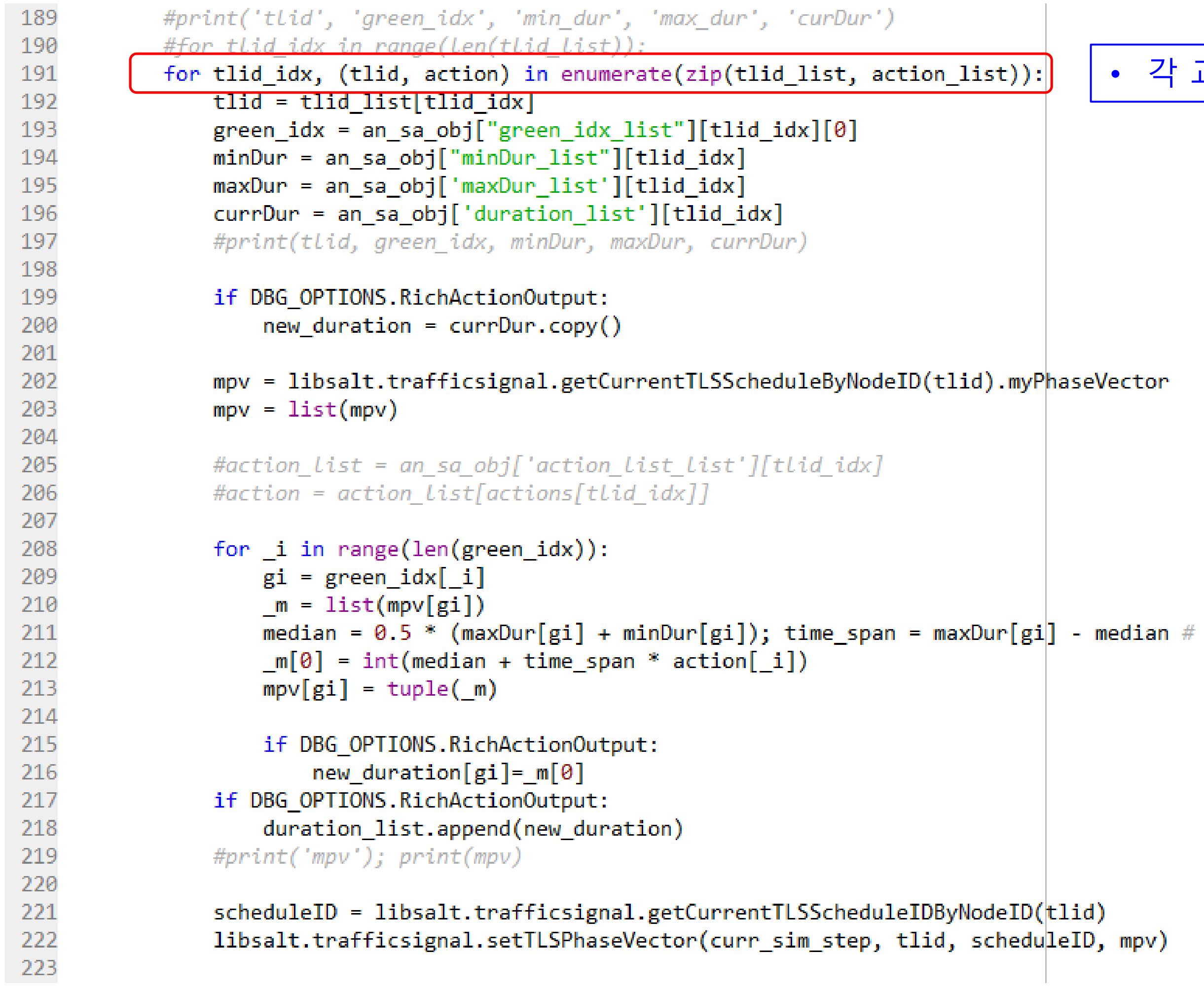
```
[[0.333],
 [-0.356],
 [0.052],
 [0.317, 0.176, 0.261, -0.446, 0.331],
 [0.044, -0.474, -0.017, 0.644],
 [-0.801, 0.259, 0.490, 0.023, -0.142, 0.0668],
 [-0.525, -0.620, 0.325, -0.432, -0.239, 0.270],
 [0.155, -0.026, -0.427, 0.43],
 [0.658, 0.178],
 [-0.475, 0.099]]
```

10개 교차로

교차 별 신호 시간 조절을 위한 action

SappoActionMgmt.py #4

__setGreenTimePhaseArray()

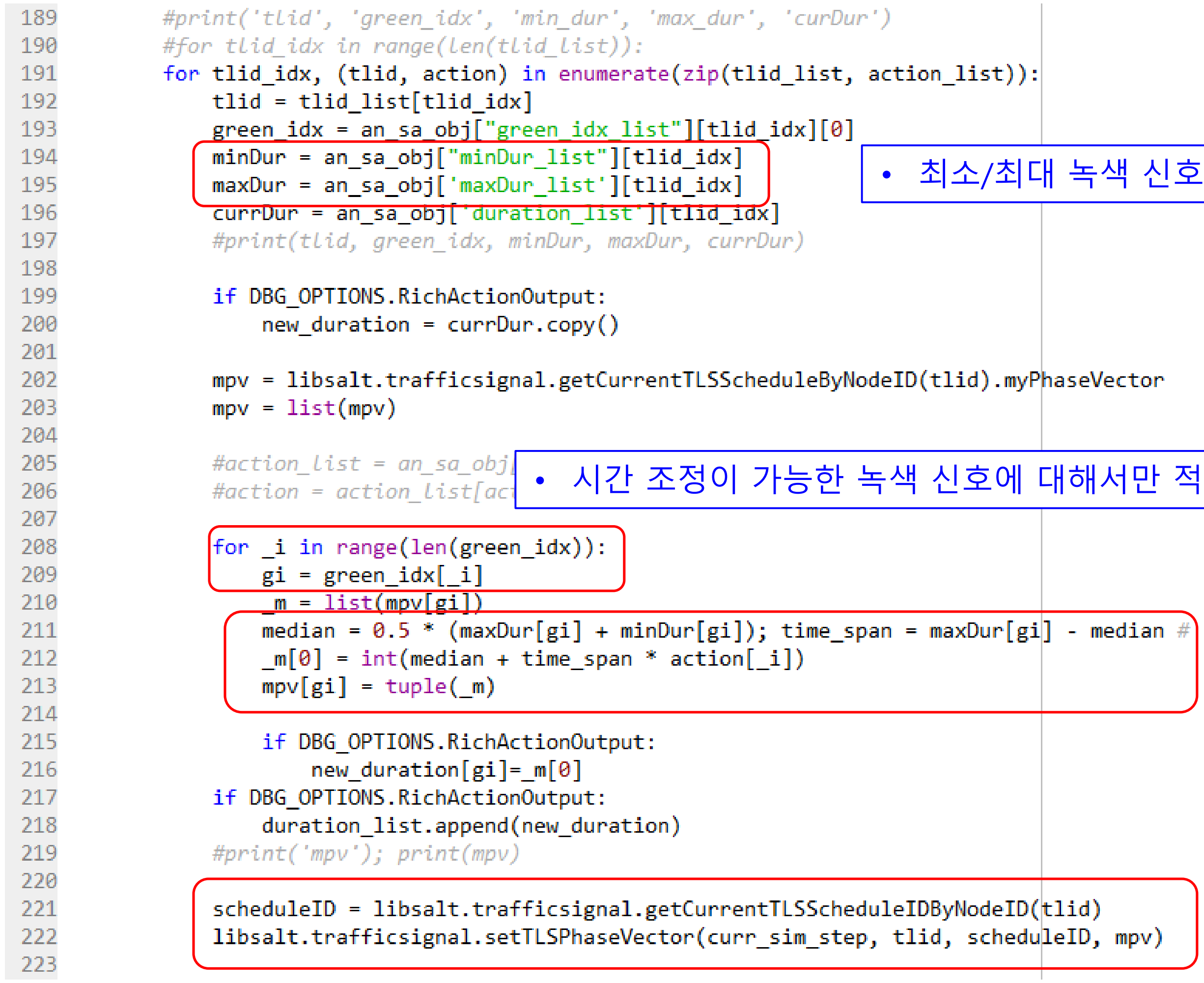


```python
        #print('tlid', 'green_idx', 'min_dur', 'max_dur', 'curDur')
        #for tlid_idx in range(len(tlid_list)):
        for tlid_idx, (tlid, action) in enumerate(zip(tlid_list, action_list)):
            tlid = tlid_list[tlid_idx]
            green_idx = an_sa_obj["green_idx_list"][tlid_idx][0]
            minDur = an_sa_obj["minDur_list"][tlid_idx]
            maxDur = an_sa_obj['maxDur_list'][tlid_idx]
            currDur = an_sa_obj['duration_list'][tlid_idx]
            #print(tlid, green_idx, minDur, maxDur, currDur)

            if DBG_OPTIONS.RichActionOutput:
                new_duration = currDur.copy()

            mpv = libsalt.trafficsignal.getCurrentTLSScheduleByNodeID(tlid).myPhaseVector
            mpv = list(mpv)

            #action_list = an_sa_obj['action_list_list'][tlid_idx]
            #action = action_list[actions[tlid_idx]]

            for _i in range(len(green_idx)):
                gi = green_idx[_i]
                _m = list(mpv[gi])
                median = 0.5 * (maxDur[gi] + minDur[gi]); time_span = maxDur[gi] - median #
                _m[0] = int(median + time_span * action[_i])
                mpv[gi] = tuple(_m)

                if DBG_OPTIONS.RichActionOutput:
                    new_duration[gi]=_m[0]
            if DBG_OPTIONS.RichActionOutput:
                duration_list.append(new_duration)
            #print('mpv'); print(mpv)

            scheduleID = libsalt.trafficsignal.getCurrentTLSScheduleIDByNodeID(tlid)
            libsalt.trafficsignal.setTLSPhaseVector(curr_sim_step, tlid, scheduleID, mpv)
```

- 각 교차로의 신호 시간 조절

SappoActionMgmt.py #5

__setGreenTimePhaseArray()



```
        #print('tlid', 'green_idx', 'min_dur', 'max_dur', 'curDur')
        #for tlid_idx in range(len(tlid_list)):
        for tlid_idx, (tlid, action) in enumerate(zip(tlid_list, action_list)):
            tlid = tlid_list[tlid_idx]
            green_idx = an_sa_obj["green_idx_list"][tlid_idx][0]
            minDur = an_sa_obj["minDur_list"][tlid_idx]
            maxDur = an_sa_obj['maxDur_list'][tlid_idx]
            currDur = an_sa_obj['duration_list'][tlid_idx]
            #print(tlid, green_idx, minDur, maxDur, currDur)

            if DBG_OPTIONS.RichActionOutput:
                new_duration = currDur.copy()

            mpv = libsalt.trafficsignal.getCurrentTLSScheduleByNodeID(tlid).myPhaseVector
            mpv = list(mpv)

            #action_list = an_sa_obj
            #action = action_list[ac

            for _i in range(len(green_idx)):
                gi = green_idx[_i]
                _m = list(mpv[gi])
                median = 0.5 * (maxDur[gi] + minDur[gi]); time_span = maxDur[gi] - median #
                _m[0] = int(median + time_span * action[_i])
                mpv[gi] = tuple(_m)

                if DBG_OPTIONS.RichActionOutput:
                    new_duration[gi]=_m[0]
            if DBG_OPTIONS.RichActionOutput:
                duration_list.append(new_duration)
            #print('mpv'); print(mpv)

            scheduleID = libsalt.trafficsignal.getCurrentTLSScheduleIDByNodeID(tlid)
            libsalt.trafficsignal.setTLSPhaseVector(curr_sim_step, tlid, scheduleID, mpv)
```

- 최소/최대 녹색 신호 시간

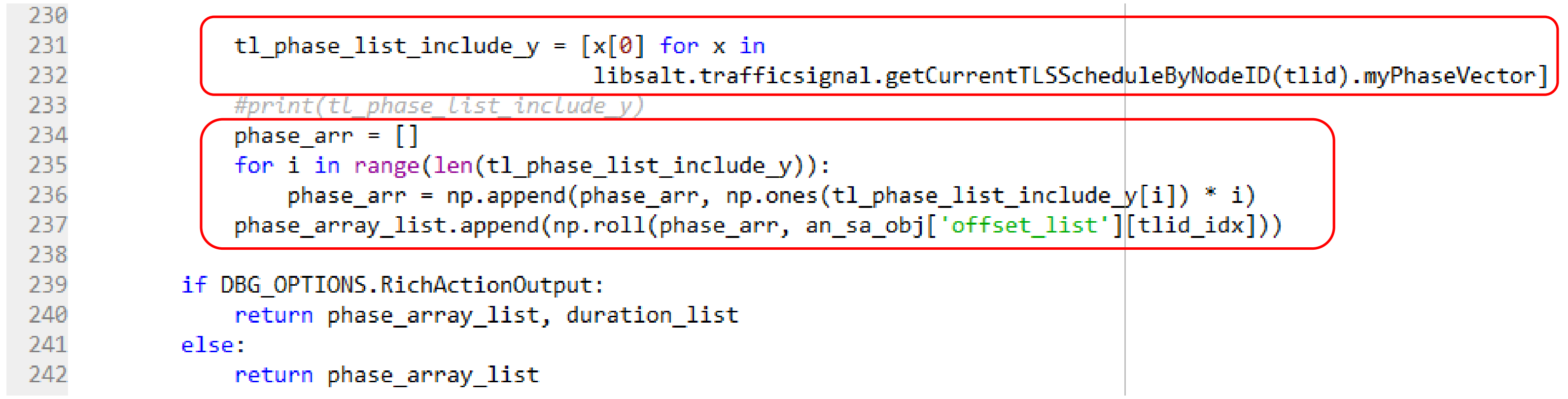
- 시간 조정이 가능한 녹색 신호에 대해서만 적용.

- 녹색 시간 조절
  - 녹색 시간 중간값 계산
  - '-1' ~ '+1'→ 'min' ~ 'max' 로 mapping
- 신호 table 작성(mpv)?

- 변경된 신호 table 등록

SappoActionMgmt.py #6

__setGreenTimePhaseArray()



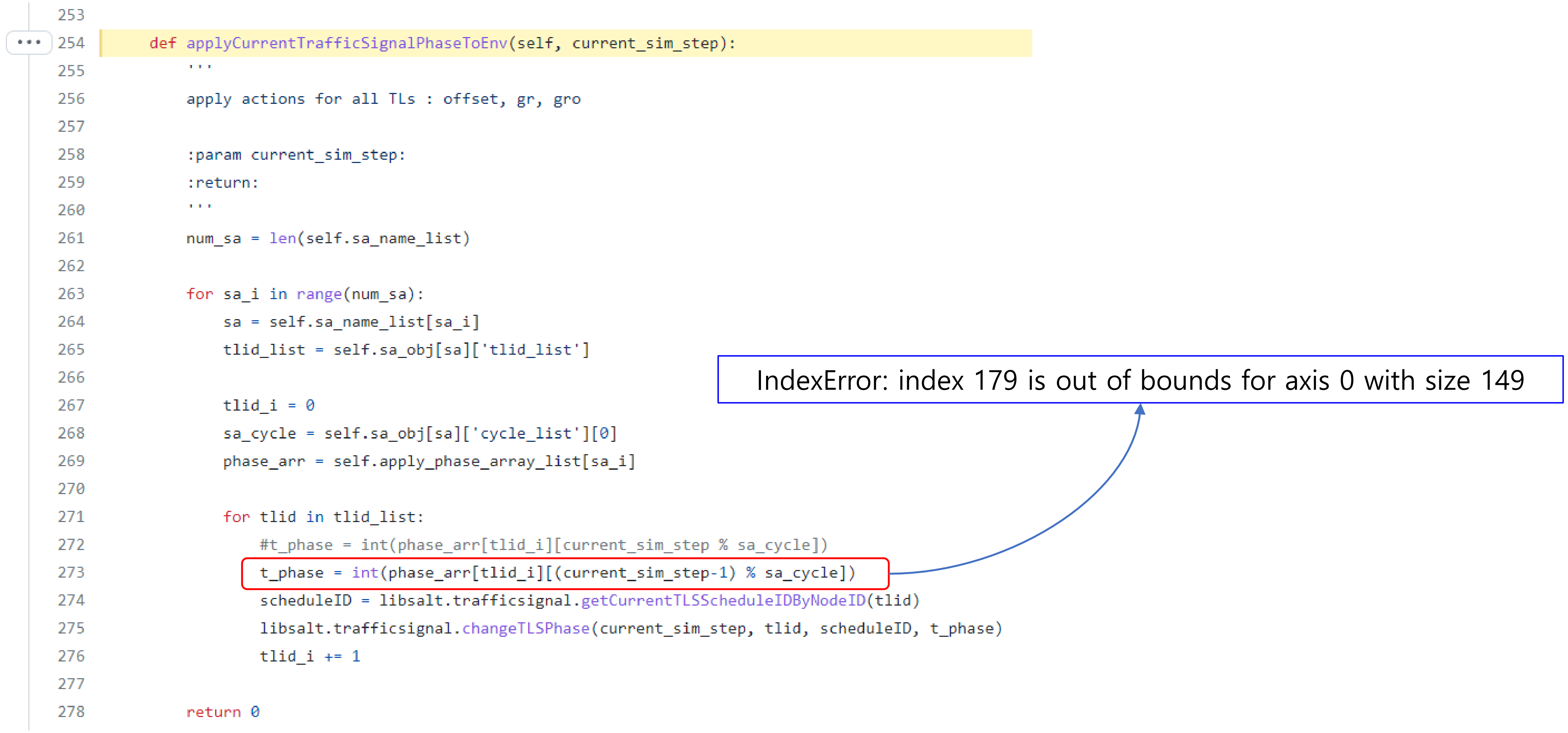
```
            tl_phase_list_include_y = [x[0] for x in
                                       libsalt.trafficsignal.getCurrentTLSScheduleByNodeID(tlid).myPhaseVector]
            #print(tl_phase_list_include_y)
            phase_arr = []
            for i in range(len(tl_phase_list_include_y)):
                phase_arr = np.append(phase_arr, np.ones(tl_phase_list_include_y[i]) * i)
            phase_array_list.append(np.roll(phase_arr, an_sa_obj['offset_list'][tlid_idx]))

        if DBG_OPTIONS.RichActionOutput:
            return phase_array_list, duration_list
        else:
            return phase_array_list
```

- tl_phase_listinclude_y와 phase_array_list 생성 부분은 변경하지 않았음.
- phase_array_list를 참조하는 SappoActionMgmt.applyCurrentTrafficSignalPhaseToEnv()에서 error가 발생함

SappoActionMgmt.py #7

applyCurrentTrafficSignalPhaseToEnv()

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/a9038816e570983a604ce14a272ba373d033acc8/atsc-rl/multiagent_tf2/env/SappoActionMgmt.py#L254



```python
    def applyCurrentTrafficSignalPhaseToEnv(self, current_sim_step):
        '''
        apply actions for all TLs : offset, gr, gro

        :param current_sim_step:
        :return:
        '''
        num_sa = len(self.sa_name_list)

        for sa_i in range(num_sa):
            sa = self.sa_name_list[sa_i]
            tlid_list = self.sa_obj[sa]['tlid_list']

            tlid_i = 0
            sa_cycle = self.sa_obj[sa]['cycle_list'][0]
            phase_arr = self.apply_phase_array_list[sa_i]

            for tlid in tlid_list:
                #t_phase = int(phase_arr[tlid_i][current_sim_step % sa_cycle])
                t_phase = int(phase_arr[tlid_i][(current_sim_step-1) % sa_cycle])
                scheduleID = libsalt.trafficsignal.getCurrentTLSScheduleIDByNodeID(tlid)
                libsalt.trafficsignal.changeTLSPhase(current_sim_step, tlid, scheduleID, t_phase)
                tlid_i += 1

        return 0
```

IndexError: index 179 is out of bounds for axis 0 with size 149

Q & A

Action Space Modeling

2023/5/17

Proposed I

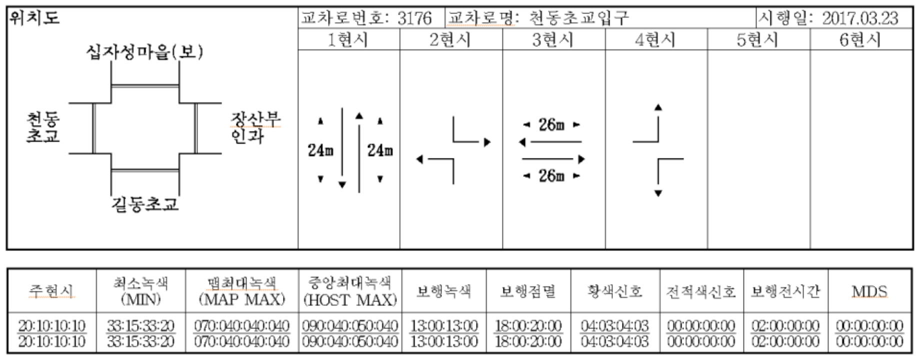
❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0
Action	$-1 \le a_1 \le +1$	$-1 \le a_2 \le +1$	$-1 \le a_3 \le +1$	$-1 \le a_4 \le +1$	$101 \le T \le 190$

$$T = (51.5 + a_1 \times 18.5) + (27.5 + a_2 \times 12.5) + (36.5 + a_3 \times 3.5) + (20.0 + a_4 \times 10.0)$$

- For each intersection, #Action outputs = #Phases
  - 신호 조합이 아닌, #Phases에 선형 비례로 증가.
- Action space를 4차원 vector로 표현
  - 모든 action 조합을 표현할 수 있음.
  - 비슷한 action vector는 실제로도 유사한 제어 신호임.
- 제약 조건
  - 최소/최대 녹색 시간 만족
  - 주기 (T=150)는 만족하지 않음.
    - Penalty 추가 (추후 고려)

천동초교입구


위치도
십자성마을(보)
천동초교
장산부인과
길동초교
24m
24m
26m
26m

교차로번호: 3176	교차로명: 천동초교입구				시행일: 2017.03.23
1현시	2현시	3현시	4현시	5현시	6현시

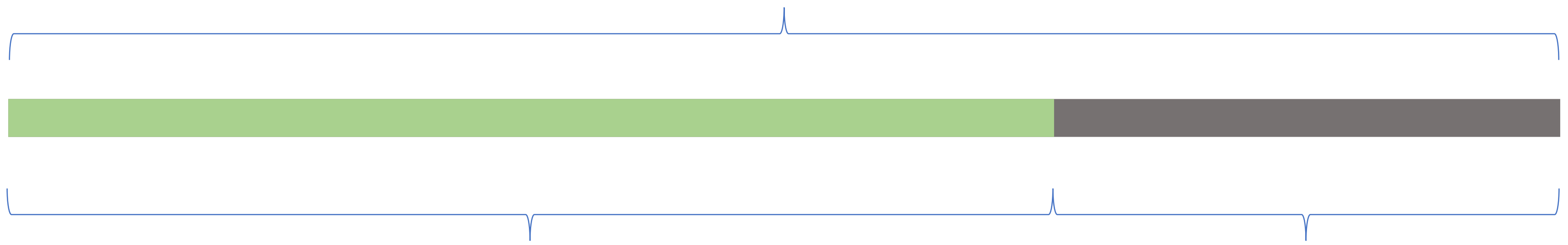
주현시	최소녹색 (MIN)	맵최대녹색 (MAP MAX)	중앙최대녹색 (HOST MAX)	보행녹색	보행점멸	황색신호	전적색신호	보행전시간	MDS
20:10:10:10	33:15:33:20	070:040:040:040	090:040:050:040	13:00:13:00	18:00:20:00	04:03:04:03	00:00:00:00	02:00:00:00	00:00:00:00
20:10:10:10	33:15:33:20	070:040:040:040	090:040:050:040	13:00:13:00	18:00:20:00	04:03:04:03	00:00:00:00	02:00:00:00	00:00:00:00

Proposed II

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

$T = 150$



$Min = 33 + 15 + 33 + 20 = 101$

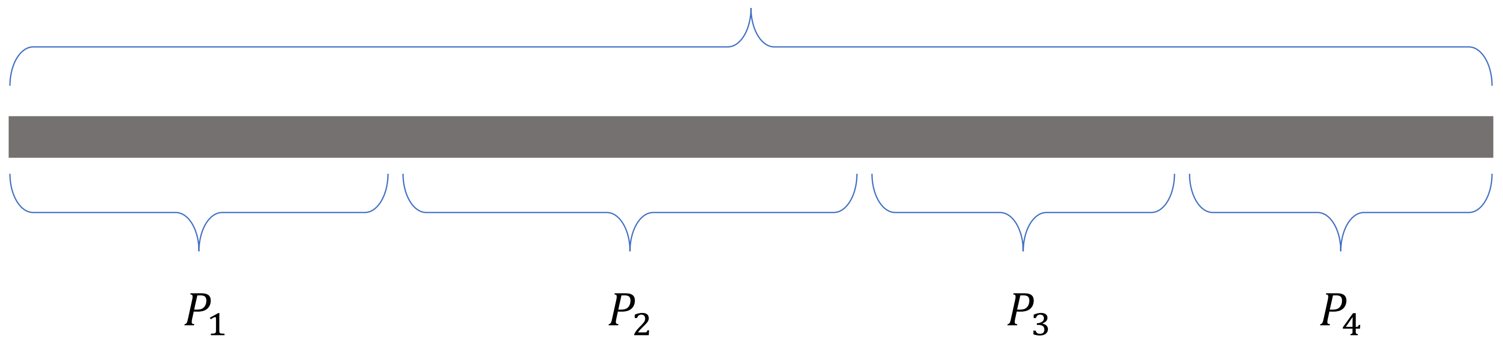
$R = T - Min = 49$

Proposed II

❖ 63:24:37:26(주기 150)

	Phase 1	Phase 2	Phase 3	Phase 4	Total
Min	33.0	15.0	33.0	20.0	101.0
Median	51.5	27.5	36.5	30.0	145.5
Max	70.0	40.0	40.0	40.0	190.0

$$R = T - Min = 49$$



- 제약 조건
  - 현시 별 최소 녹색 시간 만족. 주기(T=150) 만족.
  - 현시 별 최대 녹색 시간은 만족하지 않음.

Proposed II

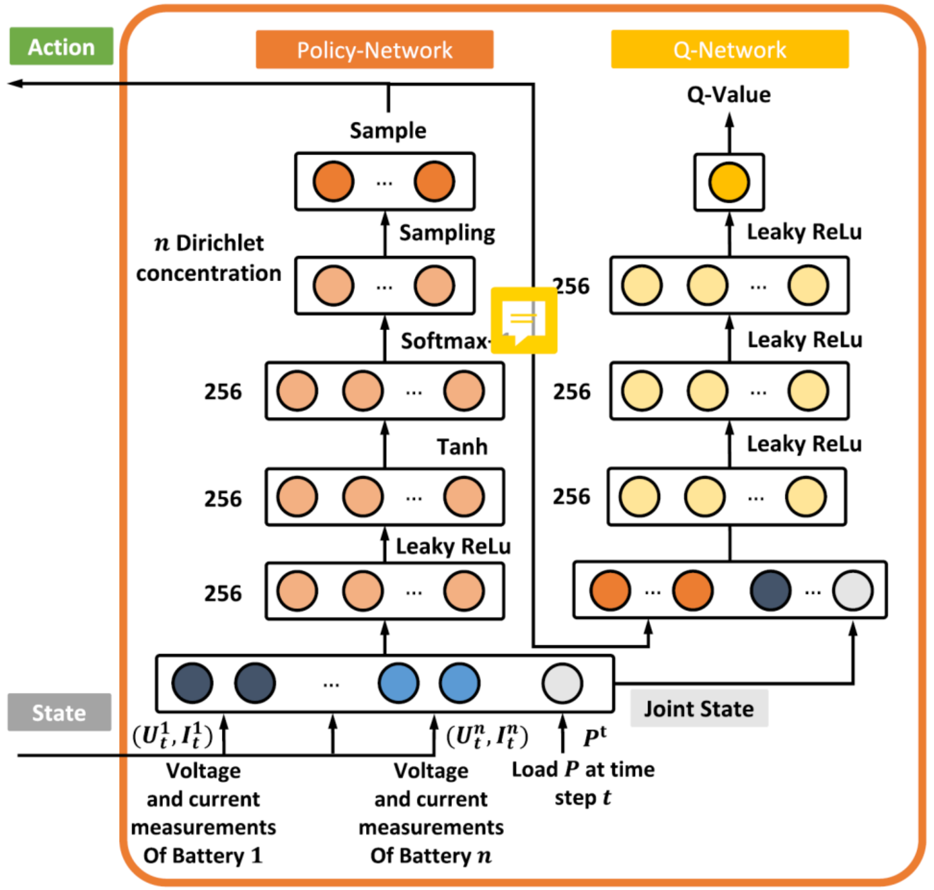
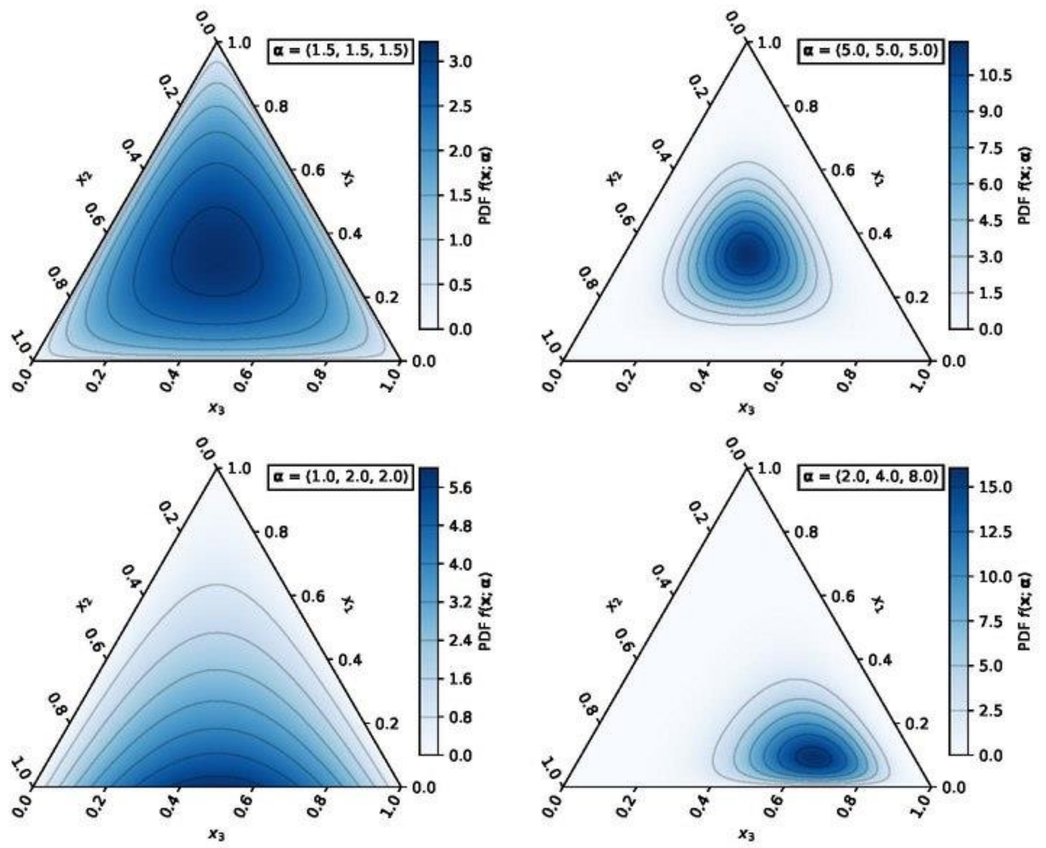

Action
Policy-Network
Q-Network
Q-Value
Sample
Sampling
n Dirichlet concentration
Softmax
Tanh
Leaky ReLu
256
State
Joint State
(U_t^1, I_t^1)
(U_t^n, I_t^n)
P^t
Voltage and current measurements Of Battery 1
Voltage and current measurements Of Battery n
Load P at time step t

Fig. 3. Overview of the neural network architectures.

Dirichlet distribution



Source: A Prescriptive Dirichlet Power Allocation Policy with Deep Reinforcement Learning

https://en.wikipedia.org/wiki/Dirichlet_distribution

Q & A