Motivated Agent with Semantic Memory

Annonymous Authors

Randomness

All performed calculations, which utilize random numbers, give repeatable results (after taking an average value).

Computing infrastructure

All performed calculations can be easily repeated using a modern PC equipped with 8GB of RAM and 4 core CPU with installed Windows or Linux operating systems and Python 3.x or Matlab environments.

Hyperparameters

A random search was performed to find the best values of hyperparameters.
For a given set of hyperparameters, calculations were repeated 11 times.
Next, the median value was taken. The average value was close to the median.

The parameters were taken from a selected set. For example, the set to choose from for REINFORCE ALGORITHM was:

hidden_sizes = [1, 2, 4, 8, 16, 32, 64, 128]
learning_rates = [0.12, 0.06, 0.03, 0.01, 0.005, 0.001, 0.0005, 0.0003, 0.0001]
max_ts = [1, 3, 15, 25, 35, 50, 75, 100, 150, 300]
gammas = [1, 0.99, 0.95, 0.9, 0.8]

Several thousand combinations were drawn by lot.

Below are the best sets of hyperparameters for the task of solving THE SIMPLE GRAPH ENVIRONMENT:

- REINFORCE: hidden_size = 128, learning_rate = 0.0003, max_t = 15, gamma = 0.99
- hQ-learning player: alpha_controler = 0.3, gamma_controler = 1.0, epsilon_controler = 0.3, eps_dec_controler = 0.3, eps_end_controler= 5e-5, alpha_worker= 1, gamma_worker=0.3, epsilon_worker= 1, eps_dec_worker= 0.25, eps_end_worker= 1e-05
- SARSA: alpha = 0.8, gamma = 0.0, epsilon = 1, eps_end = 0.0, eps_dec = 1E-4
- Q-learning: alpha = 0.8, gamma = 0.0, epsilon = 1, eps_end = 0.0, eps_dec = 5E-4
- PPO: gamma = 0.99, alpha = 1E-4, gae_lambda = 0.95, policy_clip = 0.05, batch_size = 16, n_epochs = 8, Actor_fc1_dims = 256, Actor_fc2_dims = 1024, Critic_fc1_dims = 512, Critic_fc2_dims = 32, N = 100
- DQN: max_mem_size = 500, gamma = 1.0, epsilon = 1.0, batch_size = 16, n_actions = 81, eps_end = 0.2, eps_dec = 0.005, input_dims = 11, lr = 0.0005, fc1_dims = 32, fc2_dims = 1024
- Actor-Critic: mini_episode_iters = 10, gamma = 0.99, actor_layer1=512, actor_layer2=64, critic_layer1=64, critic_layer2=128,

- ICM: neurons_input_layer=64, neurons_1st_hidden_layer=32, neurons_2_second_layer_gru=64, icm_layer_size= 4, embedding_layer_1_size = 5, embedding_layer_2_size = 5
- PolicyGradient: mini_episode_iters = 35, layer1_size = 512, gamma = 0.95, learning_rate = 0.001
- hDQN: boss_batch_size=16, boss_max_mem_size=300, boss_alfa=0.0001, boss_gamma=0.9, boss_epsilon=1.0, boss_eps_end=0.0, boss_eps_dec=0.01, boss_fc1_dims=64, boss_fc2_dims=64, worker_batch_size=16, worker_max_mem_size=2500, worker_alfa=0.0001, worker_gamma= 1.0 worker_epsilon= 1.0, worker_eps_end=0.1, worker_eps_dec=0.001, worker_fc1_dims=512, worker_fc2_dims=512

Motivated Learing agent vs. Reinforcement Learning agent - comparison

A simple graph environment for RL agents is presented here. In the linked notebooks, you will find an exemplary solution selected RL agents. These examples will allow to repeat the results presented in Figure 6 in the manuscript or test a simple graph environment on an agent of yours choice. Optimal values of hyperparameters for studied agents RL are listed in paragraph Reproducibility.

Python notebooks:

- Q-learning agent https://colab.research.google.com/drive/1-rrRwaX7AAMrdlzzvQpospCpJwGongZH
- DQN agent https://colab.research.google.com/drive/1a2V5mDVbRYfP7l5it7UI9Sb9wB4vjaZO
- notebooks for the rest of tested RL agents will be shared in camera ready version in GitHub repository

Code for the ML agent (Matlab files) with a description can be downloaded from here:
https://ks3363596.kimsufi.com/ecai_2023/
In the archive **code_for_motivated_agent.7z** You will find a file **Goal_creation_experiment_rev2.doc** with detailed description of motivated learning agent.

ANAKG semantic memory module

ANAKAG - semantic memory module works as a self-contained Windows application. It is based on papers [references will be given in camera ready version].

Executable version and source code (created using Visual Studio 2017 IDE) can be downloaded from here:
https://ks3363596.kimsufi.com/ecai_2023/

ANAKG app learns relationships between “a primitive pain and a resource and an action” or “a resource A to be refilled, a resource B used to refill resource A and action” from a text file. These relations can be recalled as providing ANAKG with a symbol of a resource to be refilled or primitive pain to be relieved.

Communication with the ANAKG module is by utilizing the TCP protocol.
The ANAKG app listens PORT = 11111

```
#connecting to the ANAKG module
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
        s.connect((HOST, PORT))

        if learn_anakg == True:
            s.sendall(b'open_dataset_and_learn<EOF>')
            data = s.recv(2048)
```

```
            print('Connected to ANAKG & ANAKG Learned')
            #print('Received', repr(data))
        else:
         print('Connected to ANAKG')
         s.sendall(b'Hello<EOF>')
         data = s.recv(2048)
            #print('Received', repr(data))
```

Using the ANAKG module as semantic memory:

```
                    #1 SEND SENSOR VALUES TO ANAKAG
                    import socket

                    if anakg_report == True:
                        print('Asking ANAKG')
                        print(iteration)
                        #creates sentence from perceived items by agent

                    white_space=' '

                    if anakg_mode == 'simple':
                        #in "simple" mode, the neuron representing max pain in ANAKG MEMORY is
activated only one time
                        MESSAGE = get_max_pain_as_word( paintrig,
sensory_pains_as_words_4_anakg)

                    elif anakg_mode == 'intrusive':
                        #in "intrusive" mode, neuron representing max pain in ANAKG MEMORY is
exited several times
                        MESSAGE = get_max_pain_as_word( paintrig,
sensory_pains_as_words_4_anakg) + white_space
                        for i in range(0,20):
                               MESSAGE += get_max_pain_as_word( paintrig ,
sensory_pains_as_words_4_anakg) + white_space

                    #encode message to transfer by TCP to ANAKG memory
                    MESSAGE = MESSAGE.encode()
                    MESSAGE += b'<EOF>'

                    if anakg_report == True:
                        print('Message: '+MESSAGE.decode())
                    # sends sentence to ANAKG Memory
                    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                    s.connect((TCP_IP, TCP_PORT))
                    s.send(MESSAGE)

                    #2 RECIEVE ANSWER
                    recieved_data = s.recv(2048)
                    recieved_data = recieved_data.decode()

                    #recieved_data=recieved_data[0:-5] #strip of <EOF>
                    s.close()

                recieved_data = recieved_data.strip().upper()

                if anakg_report == True:
                    print('Received: ', recieved_data)

                #3 DECODE ANSWER TO AGENT REPRESENTATION
                anakg_motorIDs_clue = get_motor_ids(recieved_data, motor_output_list)
                anakg_sensorIDs_clue = get_sensor_ids(recieved_data, sensory_perceived_items)

                #list of goals (clues) to increase
                goals_to_increase = get_goalval_ids(anakg_motorIDs_clue, anakg_sensorIDs_clue,
len(motor_output_list), len(sensory_perceived_items))
```

Text File contains information about rules governing an environment. Some of the examples of such relations for an environment presented in the picture below are:

FULL RELATIONS:

- to reduce primitive pain P0 use action m_25 on resource s_16
- to reduce primitive pain P0 use action m_0 on resource s_0
- to reduce primitive pain P1 use action m_2 on resource s_0
- to resupply resource s_3 use action m_7 on resource s_5
- to resupply resource s_7 use action m_15 on resource s_8
- …

PARTIAL RELATIONS:

- to reduce primitive pain P0 use action m_25 {proper resource is not given}
- to reduce primitive pain P0 use resource s_0 {proper action m_0 is not given}
- to resupply resource s_3 use action m7
- to resupply resource s_7 use resource s_8

  …

Text file stores relations coded in ANAKG representation as below:

FULL:

P0 m_25 s_16
P0 m_0 s_0
P1 m_2 s_0
S_3 m_7 s_5
…

PARTIAL:

P0 m_25
P0 s_0
s_7 s_8
…

where P0, P1, s_0, m_25 are symbols representing primitive pains, resources, and actions in the ANAKG module. If an agent wants to ask ANAKG about a clue to relieve pain p_m , the agent must encode an internal representation of pain p_m to the representation of the ANAKG network. Similarly, ANAKG recall will contain information about resources and actions encoded in ANAKG representation, so the agent must decode them to its internal representation.

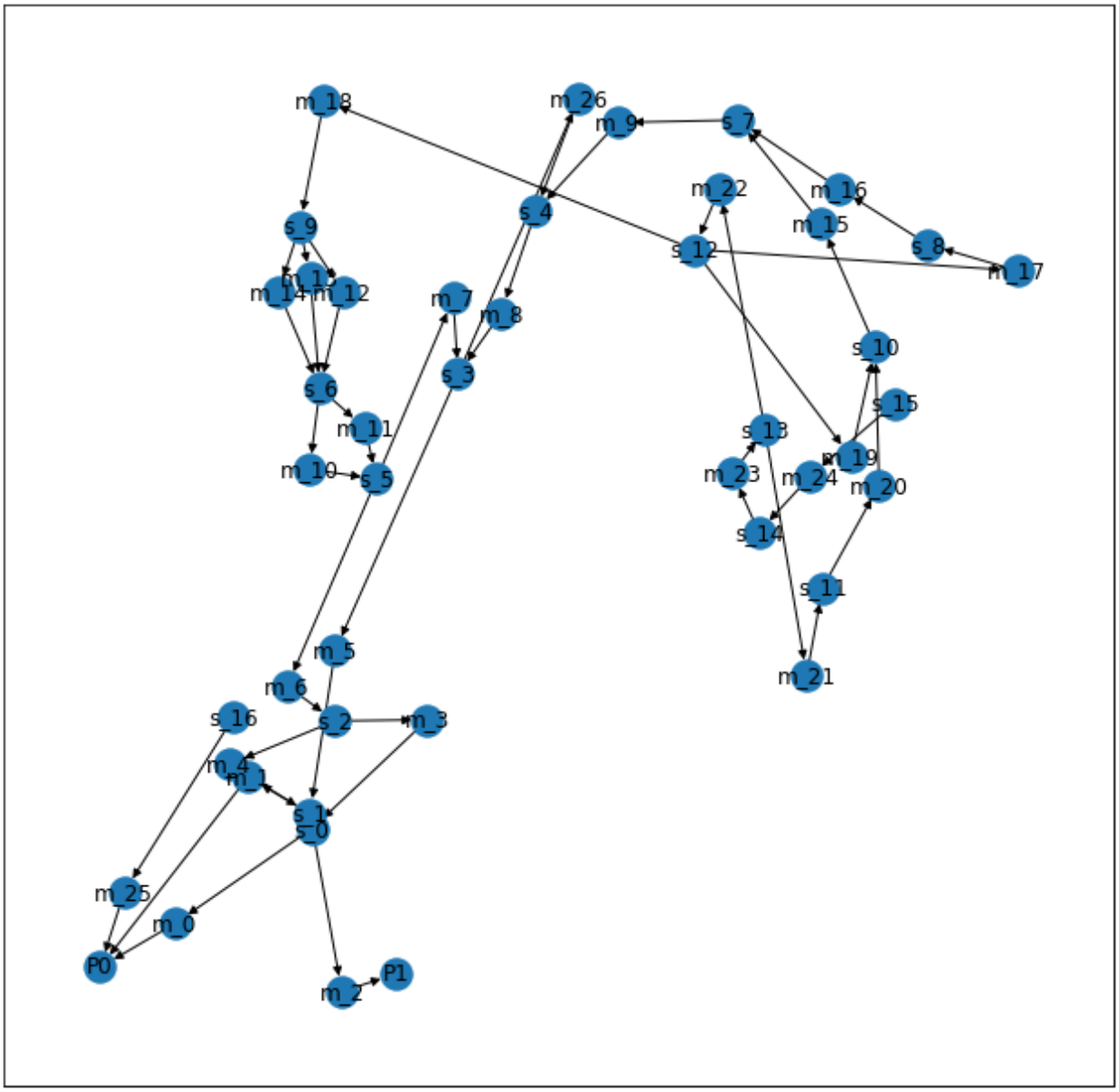

Figure 7. Relations between resources (s_0, s_1, …), actions (m_0, m_1,...) and primitive pains (P0 and P1) for the environment represented as a graph.

COMPLEX ENVIRONMENT rules

Rules (relations) between primitive pains P0, P1, P2, P3 and P4 (indexes 117, 118, 119, 120, 121) and resources marked with letter R and actions marked with letter M for the complex environment analyzed in the manuscript.

Row ID and brackets [,] indicate a:
id_of_pain_or_resource_to_be_restored using [*id_of_resource*, *id_of_action*]

Some resources can be restored using more than one path. ID of such resources are 4, 5, 12, 17, 26, 34, 48, 59, 69, 80, 89, 95, 103 and 109. Rows with an empty [ ] indicate resources which are inexhaustible (like seawater in sea).

RULES:

{0: [[12, 8]],
1: [[6, 0]],
2: [[10, 4]],
3: [[11, 4]],
4: [[9, 8], [116, 9]],
5: [[10, 4], [7, 0], [8, 4]],
6: [[14, 2]],
7: [[13, 1]],
8: [[17, 9]],
9: [[14, 5]],
10: [[14, 6]],
11: [[15, 9]],
12: [[15, 7], [16, 6]],
13: [[24, 4]],
14: [[19, 7]],
15: [[23, 3]],
16: [[25, 9]],
17: [[25, 3], [18, 0], [20, 9], [21, 1], [22, 6], [26, 7]],
18: [[27, 2]],
19: [[33, 7]],
20: [[30, 6]],
21: [[31, 7]],
22: [[34, 5]],
23: [[27, 4]],
24: [[28, 1]],
25: [[28, 7]],
26: [[33, 9], [29, 5], [32, 8]],
27: [[47, 1]],
28: [[40, 2]],
29: [[37, 8]],
30: [[48, 3]],
31: [[47, 4]],
32: [[37, 2]],
33: [[42, 6]],
34: [[35, 2], [36, 8], [38, 8], [39, 9], [41, 5], [43, 6], [44, 6], [45, 4], [46, 8]],
35: [[59, 3]],
36: [[51, 4]],
37: [[55, 2]],
38: [[54, 8]],
39: [[54, 9]],

40: [[51, 2]],
41: [[49, 7]],
42: [[52, 0]],
43: [[52, 5]],
44: [[50, 8]],
45: [[54, 8]],
46: [[59, 8]],
47: [[56, 8]],
48: [[52, 4], [53, 7], [57, 1], [58, 7]],
49: [[64, 6]],
50: [[61, 8]],
51: [[67, 8]],
52: [[60, 1]],
53: [[63, 2]],
54: [[68, 3]],
55: [[65, 8]],
56: [[68, 8]],
57: [[63, 0]],
58: [[66, 5]],
59: [[67, 1], [62, 3], [69, 6]],
60: [[76, 8]],
61: [[75, 5]],
62: [[71, 0]],
63: [[76, 0]],
64: [[72, 9]],
65: [[73, 2]],
66: [[72, 4]],
67: [[73, 1]],
68: [[74, 3]],
69: [[71, 4], [70, 0], [77, 3], [78, 2], [79, 6], [80, 4]],
70: [[89, 1]],
71: [[85, 6]],
72: [[88, 6]],
73: [[82, 5]],
74: [[82, 8]],
75: [[82, 0]],
76: [[86, 7]],
77: [[89, 9]],
78: [[87, 6]],
79: [[85, 5]],
80: [[88, 3], [81, 4], [83, 6], [84, 3]],
81: [[95, 3]],
82: [[92, 7]],
83: [[91, 2]],
84: [[92, 7]],
85: [[90, 1]],
86: [[90, 8]],
87: [[94, 0]],
88: [[92, 8]],
89: [[95, 4], [93, 9]],
90: [[101, 0]],
91: [[102, 2]],
92: [[99, 2]],
93: [[99, 7]],
94: [[99, 1]],

```
 95: [[99, 6], [96, 2], [97, 9], [98, 4], [100, 3], [103, 3]],
 96: [[106, 1]],
 97: [[109, 6]],
 98: [[108, 5]],
 99: [[108, 9]],
 100: [[107, 2]],
 101: [[109, 9]],
 102: [[104, 6]],
 103: [[109, 6], [105, 4]],
 104: [[114, 2]],
 105: [[113, 6]],
 106: [[111, 1]],
 107: [[111, 4]],
 108: [[114, 8]],
 109: [[111, 8], [110, 4], [112, 0]],
 110: [ ],
 111: [ ],
 112: [ ],
 113: [ ],
 114: [ ],
 115: [ ],
 116: [ ],
 117: [[0, 3]],
 118: [[5, 8], [115, 5]],
 119: [[3, 9]],
 120: [[5, 1]],
 121: [[5, 9], [1, 3], [2, 6], [4, 2]],
 122: [ ]
}
```

Training sentences for the Complex environment (100% full support)

```
P0 R0 M3
P0 M3 R0
P1 R5 M8
P1 M8 R5
P1 R115 M5
P1 M5 R115
P2 R3 M9
P2 M9 R3
P3 R5 M1
P3 M1 R5
P4 R5 M9
P4 M9 R5
P4 R1 M3
P4 M3 R1
P4 R2 M6
P4 M6 R2
P4 R4 M2
P4 M2 R4
R0 R12 M8
R0 M8 R12
R1 R6 M0
R1 M0 R6
R2 M4 R10
R2 R10 M4
R3 R11 M4
```

```
R3 M4 R11
R4 R9 M8
R4 M8 R9
R4 R116 M9
R4 M9 R116
R5 R10 M4
R5 M4 R10
R5 R7 M0
R5 M0 R7
R5 R8 M4
R5 M4 R8
R6 R14 M2
R6 M2 R14
R7 R13 M1
R7 M1 R13
R8 R17 M9
R8 M9 R17
R9 R14 M5
R9 M5 R14
R10 R14 M6
R10 M6 R14
R11 R15 M9
R11 M9 R15
R12 R15 M7
R12 M7 R15
R12 R16 M6
R12 M6 R16
R13 R24 M4
R13 M4 R24
R14 R19 M7
R14 M7 R19
R15 R23 M3
R15 M3 R23
R16 R25 M9
R16 M9 R25
R17 R25 M3
R17 M3 R25
R17 R18 M0
R17 M0 R18
R17 R20 M9
R17 M9 R20
R17 R21 M1
R17 M1 R21
R17 R22 M6
R17 M6 R22
R17 R26 M7
R17 M7 R26
R18 R27 M2
R18 M2 R27
R19 R33 M7
R19 M7 R33
R20 R30 M6
R20 M6 R30
R21 R31 M7
R21 M7 R31
R22 R34 M5
R22 M5 R34
R23 R27 M4
R23 M4 R27
R24 R28 M1
R24 M1 R28
R25 R28 M7
R25 M7 R28
```

```
R26 R33 M9
R26 M9 R33
R26 R29 M5
R26 M5 R29
R26 R32 M8
R26 M8 R32
R27 R47 M1
R27 M1 R47
R28 R40 M2
R28 M2 R40
R29 R37 M8
R29 M8 R37
R30 R48 M3
R30 M3 R48
R31 R47 M4
R31 M4 R47
R32 R37 M2
R32 M2 R37
R33 R42 M6
R33 M6 R42
R34 R35 M2
R34 M2 R35
R34 R36 M8
R34 M8 R36
R34 R38 M8
R34 M8 R38
R34 R39 M9
R34 M9 R39
R34 R41 M5
R34 M5 R41
R34 R43 M6
R34 M6 R43
R34 R44 M6
R34 M6 R44
R34 R45 M4
R34 M4 R45
R34 R46 M8
R34 M8 R46
R35 R59 M3
R35 M3 R59
R36 R51 M4
R36 M4 R51
R37 R55 M2
R37 M2 R55
R38 R54 M8
R38 M8 R54
R39 R54 M9
R39 M9 R54
R40 R51 M2
R40 M2 R51
R41 R49 M7
R41 M7 R49
R42 R52 M0
R42 M0 R52
R43 R52 M5
R43 M5 R52
R44 R50 M8
R44 M8 R50
R45 R54 M8
R45 M8 R54
R46 R59 M8
R46 M8 R59
R47 R56 M8
```

```
R47 M8 R56
R48 R52 M4
R48 M4 R52
R48 R53 M7
R48 M7 R53
R48 R57 M1
R48 M1 R57
R48 R58 M7
R48 M7 R58
R49 R64 M6
R49 M6 R64
R50 R61 M8
R50 M8 R61
R51 R67 M8
R51 M8 R67
R52 R60 M1
R52 M1 R60
R53 R63 M2
R53 M2 R63
R54 R68 M3
R54 M3 R68
R55 R65 M8
R55 M8 R65
R56 R68 M8
R56 M8 R68
R57 R63 M0
R57 M0 R63
R58 R66 M5
R58 M5 R66
R59 R67 M1
R59 M1 R67
R59 R62 M3
R59 M3 R62
R59 R69 M6
R59 M6 R69
R60 R76 M8
R60 M8 R76
R61 R75 M5
R61 M5 R75
R62 R71 M0
R62 M0 R71
R63 R76 M0
R63 M0 R76
R64 R72 M9
R64 M9 R72
R65 R73 M2
R65 M2 R73
R66 R72 M4
R66 M4 R72
R67 R73 M1
R67 M1 R73
R68 R74 M3
R68 M3 R74
R69 R71 M4
R69 M4 R71
R69 R70 M0
R69 M0 R70
R69 R77 M3
R69 M3 R77
R69 R78 M2
R69 M2 R78
R69 R79 M6
R69 M6 R79
```

```
R69 R80 M4
R69 M4 R80
R70 R89 M1
R70 M1 R89
R71 R85 M6
R71 M6 R85
R72 R88 M6
R72 M6 R88
R73 R82 M5
R73 M5 R82
R74 R82 M8
R74 M8 R82
R75 R82 M0
R75 M0 R82
R76 R86 M7
R76 M7 R86
R77 R89 M9
R77 M9 R89
R78 R87 M6
R78 M6 R87
R79 R85 M5
R79 M5 R85
R80 R88 M3
R80 M3 R88
R80 R81 M4
R80 M4 R81
R80 R83 M6
R80 M6 R83
R80 R84 M3
R80 M3 R84
R81 R95 M3
R81 M3 R95
R82 R92 M7
R82 M7 R92
R83 R91 M2
R83 M2 R91
R84 R92 M7
R84 M7 R92
R85 R90 M1
R85 M1 R90
R86 R90 M8
R86 M8 R90
R87 R94 M0
R87 M0 R94
R88 R92 M8
R88 M8 R92
R89 R95 M4
R89 M4 R95
R89 R93 M9
R89 M9 R93
R90 R101 M0
R90 M0 R101
R91 R102 M2
R91 M2 R102
R92 R99 M2
R92 M2 R99
R93 R99 M7
R93 M7 R99
R94 R99 M1
R94 M1 R99
R95 R99 M6
R95 M6 R99
R95 R96 M2
```

```
R95 M2 R96
R95 R97 M9
R95 M9 R97
R95 R98 M4
R95 M4 R98
R95 R100 M3
R95 M3 R100
R95 R103 M3
R95 M3 R103
R96 R106 M1
R96 M1 R106
R97 R109 M6
R97 M6 R109
R98 R108 M5
R98 M5 R108
R99 R108 M9
R99 M9 R108
R100 R107 M2
R100 M2 R107
R101 R109 M9
R101 M9 R109
R102 R104 M6
R102 M6 R104
R103 R109 M6
R103 M6 R109
R103 R105 M4
R103 M4 R105
R104 R114 M2
R104 M2 R114
R105 R113 M6
R105 M6 R113
R106 R111 M1
R106 M1 R111
R107 R111 M4
R107 M4 R111
R108 R114 M8
R108 M8 R114
R109 R111 M8
R109 M8 R111
R109 R110 M4
R109 M4 R110
R109 R112 M0
R109 M0 R112
```