```
601         x = theta_core_radius * math.cos(theta_core[i])
602         y = theta_core_radius * math.sin(theta_core[i])
603         theta_core_pos[0, i] = x
604         theta_core_pos[1, i] = y
605
606     segAmount = 6
607
608     # ####print('theta_core_pos is: ' + str(theta_core_pos))
609
610     routing_SOL = []
611     totaltime = 0
612     pos_matrix = []
613     adjacency_matrix_buff = []
614     destID = numpy.zeros(segAmount)
615     pos_matrix=numpy.zeros((aus_num*(max_graph_size+addon+1),2))
616     for i in range(segAmount):
617         ang = theta_bound[i]
618         (pos_matrix1, adjacency_matrix, ID) = segGraph(i, segAmount,
619                 theta_core_pos, ang)
620         name = 'posdata' + str(i) + '.txt'
621         pos_matrix_temp = numpy.loadtxt(name)
622         col = [0, 0]
623         pos_matrix_temp = numpy.insert(pos_matrix_temp, 0, col, axis=1)
624         #print('pos_matrix_temp:'+str(pos_matrix_temp))
625         for j in range(max_graph_size+addon+1):
626                 idx=i*(max_graph_size+addon+1)+j
627                 pos_matrix[idx][0]=pos_matrix_temp[0][j]
628                 pos_matrix[idx][1]=pos_matrix_temp[1][j]
629
630         adjacency_matrix_buff.append(copy.copy(adjacency_matrix))
631         destID[i] = ID
632
633     obsolete_idx = []
634
635     # destID=numpy.zeros(segAmount)
636     #pos_matrix=pos_matrix.transpose()
637     FND_flag = False
638     HND_flag = False
639     LND_flag=False
640     DeadRatio_1500R = False
641     DR_1500 = 0
642     seg_count = 0
643
644     LND_num_1 = 0
645
646     cl_buff = numpy.ones(aus_num * (max_graph_size+addon + 1)) * 60
647
648     run_flag = True
649
650     P = 0.2
651
652     round_limit = int(1 / P)
653     alpha = 1
```