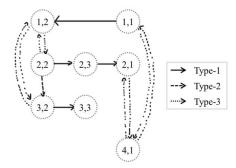
1

```
for k in range(message):
import numpy as np
                                                                                                                        flag=1
                                                                                                                       Q[k]=B[k]
import matplotlib.pyplot as plt
import pandas as pd
                                                                                                                        while flag==1:
from PIL import Image
                                                                                                                            j=0
from skimage import measure
                                                                                                                            tmp[k]=0
import math
                                                                                                                            for j in range(k):
import random
                                                                                                                               tmp[k]=tmp[k]+math.ceil((Q[k]+tao)/T[j])*C[j]
                                                                                                                            rhs[k]=B[k]+tmp[k]
count=0
                                                                                                                           if rhs[k]+C[k]>T[k]:
R[k]=1500
tmax=100000
tnow=tmax
tmin=1
                                                                                                                           elif rhs[k]+C[k]<=T[k]:
r=0.95
                                                                                                                               if Q[k]==rhs[k]:
message=17
                                                                                                                                   R[k]=Q[k]+C[k]
tao=0.001
                                                                                                                                   flag=0
                                                                                                                                elif Q[k]!=rhs[k]:
vnow=208.55999994277954
vbest=vnow
                                                                                                                                   Q[k]=rhs[k]
vbestforever=vnow
                                                                                                                    u=summary(R)
C=np.array([0.52,0.6,0.52,0.6,0.52,0.6,0.92,0.52,0.6,0.68,0.52,0.76,0.52,0.52,0.68,0.52,0.52])
P=np.array([0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16])
                                                                                                               print(objective(C,T,P,tmax,tmin,r,message,tao)) #印初始的總和208.56
B=np.zeros([message])
Q=np.zeros([message])
                                                                                                               evetime_value = []
R=np.zeros([message])
newP=np.zeros([message])
                                                                                                               while True:
                                                                                                                   if tnow<=tmin:
def change(C,T,P,m,n):#找鄰居的解,用兩值互換的方法
                                                                                                                   for i in range(15):
   C[m]=C[n]
C[n]=tmp
                                                                                                                       num+=1
                                                                                                                        m=random.randint(0, 8)
    tmp=T[m]
                                                                                                                        n=random.randint(9, 16)
    T[m]=T[n]
T[n]=tmp
                                                                                                                        change(C,T,P,m,n)
                                                                                                                        vnow=objective(C,T,P,tmax,tmin,r,message,tao)
    tmp=P[m]
                                                                                                                        diff=vnow-vbest
    P[m]=P[n]
P[n]=tmp
                                                                                                                        if(vnow<vbestforever):#將出現過最小的總和值存入vbestforever
                                                                                                                           vbestforever=vnow
                                                                                                                        if diff<=0:
def summary(R):
    for k in range(message):
                                                                                                                           print(objective(C,T,P,tmax,tmin,r,message,tao),1)
        add=add+(R[k])
                                                                                                                           evetime_value.append(vnow)
                                                                                                                       elif diff>0:
                                                                                                                           prob =math.exp(-diff/tnow)
def objective(C,T,P,tmax,tmin,r,message,tao):#算此順序下的總和
                                                                                                                           randum=random.uniform(0,1)
                                                                                                                           if randum<prob:
    for k in range(message):
                                                                                                                               vbest=vnow
        B[k]=C[k]
                                                                                                                               print(objective(C,T,P,tmax,tmin,r,message,tao),2)
                                                                                                                               evetime_value.append(vnow)
                                                                                                                           elif randum>=prob:
   for k in range(message):
    for i in range(k,message,1):
        if C[i]>B[k]:
                                                                                                                               change(C,T,P,m,n)
                                                                                                                               vnow=objective(C,T,P,tmax,tmin,r,message,tao)
                                                                                                                   tnow=tnow*r
               B[k]=C[i]
                                                                                                               print(objective(C,T,P,tmax,tmin,r,message,tao),3)
    k=0
                                                                                                                for i in range(17):
                                                                                                                   print("message",i,"的priority為:",P[i])
    flag=1
                                                                                                               print("最佳的summation of the worst-case response time:",vbestforever)
    tmp=np.zeros([message])
                                                                                                               print(num)
    rhs=np.zeros([message])
                                                                                                               plt.figure(figsize = (10,6))
    for k in range(message):
                                                                                                               plt.xlabel("Iteration", fontsize = 15)
        flag=1
                                                                                                               plt.ylabel("value", fontsize = 15)
        Q[k]=B[k]
                                                                                                               plt.plot(evetime_value,linewidth = 1, label = "smallest value ever", color = 'r')
        while flag==1:
                                                                                                               plt.legend()
            j=0
                                                                                                               plt.show()
```

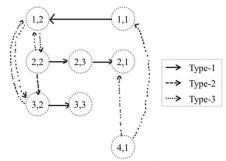
```
Console 1/A ×
203.4 3
message 0 的priority為: 5
message 1 的priority為: 7
message 2 的priority為: 3
message 3 的priority為: 1
message 4 的priority為: 4
message 5 的priority為: 2
message 6 的priority為: 8
message 7 的priority為: 10
message 8 的priority為: 6
message 9 的priority為: 9
message 10 的priority為: 16
message 11 的priority為: 13
message 12 的priority為: 12
message 13 的priority為: 14
message 14 的priority為: 0
message 15 的priority為: 15
message 16 的priority為: 11
最佳的summation of the worst-case response time: 204.6799999999998
```

import cv2	def change(C,T,P,m,n):#找鄰居的解,用兩值互換的方法	if C[i]>B[k]:		randum=random.uniform(0,1)
import numpy as np	tmp=C[m]	B[k]=C[i]	print(objective(C,T,P,tmax,tmin,r,message,tao)) #印 初始的總和208.56	if randum <prob:< td=""></prob:<>
import matplotlib.pyplot as plt	C[m]=C[n]	k=0	MANUAL 1700-130	vbest=vnow
import pandas as pd	C[n]=tmp	j=0	evetime_value = []	print(objective(C,T,P,tmax,tmin,r,message,tao),2)
from PIL import Image		flag=1		evetime_value.append(vnow)
from skimage import measure	tmp=T[m]	tmp=np.zeros([message])	num=0 while True:	_
import math	T[m]=T[n]	rhs=np.zeros([message])		elif randum>=prob:
import random	T[n]=tmp	for k in range(message):	if tnow<=tmin:	change(C,T,P,m,n)
count=0	tmp=P[m]	flag=1	break	vnow=objective(C,T,P,tmax,tmin,r,message,tao)
tmax=100000	P[m]=P[n]	Q[k]=B[k]	for i in range(15):	tnow=tnow*r
tnow=tmax	P[n]=tmp	while flag==1:	num+=1	print(objective(C,T,P,tmax,tmin,r,message,tao),3)
tmin=1	return 0	j=0	m=random.randint(0, 8)	for i in range(17):
r=0.95		tmp[k]=0	n=random.randint(9, 16)	print("message",i,"的priority為:",P[i])
message=17	def summary(R):	for j in range(k):	change(C,T,P,m,n)	print("最佳的summation of the worst-case response
tao=0.001	add=0		vnow=objective(C,T,P,tmax,tmin,r,message,tao)	
vnow=208.55999994277954		tmp[k]=tmp[k]+math.ceil((Q[k]+tao)/T[j])*C[j]	diff=vnow-vbest	print(num)
vbest=vnow	add=add+(R[k])	rhs[k]=B[k]+tmp[k]	if(vnow <vbestforever):#將出現過最小的總和 值存入vbestforever</vbestforever):#將出現過最小的總和 	plt.figure(figsize = (10,6))
vbestforever=vnow	return add	if rhs[k]+C[k]>T[k]:	vbestforever=vnow	plt.xlabel("Iteration",fontsize = 15)
C=np.array([0.52,0.6,0.52,0.6,0.52,0.6,0.92,0.52,0.6,		R[k]=1500	if diff<=0:	plt.ylabel("value",fontsize = 15)
0.68,0.52,0.76,0.52,0.52,0.68,0.52,0.52])	def objective(C,T,P,tmax,tmin,r,message,tao):#算此順序下的總和	break	#count=count+1	plt.plot(evetime_value,linewidth = 1, label = "smallest value ever", color = 'r')
T=np.array([50,5,5,5,5,5,10,10,10,10,50,100,100,100,) k=0	elif rhs[k]+C[k]<=T[k]:		plt.legend()
P=np.array([0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]) for k in range(message):	if Q[k]==rhs[k]:	<pre>#print(objective(C,T,P,tmax,tmin,r,message,tao),cou nt)</pre>	plt.show()
B=np.zeros([message])	В[k]=С[k]	R[k]=Q[k]+C[k]	vbest=vnow	
Q=np.zeros([message])	i=0	flag=0		
R=np.zeros([message])	k=0	elif Q[k] l=rhs[k]:	print(objective(C,T,P,tmax,tmin,r,message,tao),1)	
newP=np.zeros([message])	for k in range(message):	Q[k]=rhs[k]	evetime_value.append(vnow)	
	for i in range(k,message,1):	u=summary(R)	elif diff>0:	
		return u	prob =math.exp(-diff/tnow)	

1. (8pts) Given the scenario in the figure above, follow the legend and draw the corresponding timing conflict graph.



2. (12pts) Following 1., given that Vehicle 4 enters Conflict Zone 1 before Vehicles 1 and 2, find a DEADLOCK solution which has no cycle in the corresponding timing conflict graph. Follow the legend and draw the corresponding timing conflict graph. Explain why there is a deadlock.



當 vehicle 1在 zone 1, vehicle 2在 zone 3, vehicle 3在 zone 2 時, 則 vehicle 1頻等vehicle 3離開 zone 2, vehicle 2頻等vehicle 1 離開 zone 1, vehicle 3 頻等 vehicle 2離開 zone 3, 比時發生Jealloulc.

因為只有一個conflict zone,所以當vehicle進入後便可以直接離開此zone給下一個vehicle使用,故不會出現在resource conflict graph中,同時此例也不會產生resource conflict graph,因此此時當原本的圖沒有cycle,則不會有deadlock,並且依照到達的先後順序通過即可

可以只使用rule1,4,因為rule1是處理同一個vehicle(只包含type1時),自己進入conflict zone的先後順序,即代表transition發生的先後順序,rule4是處理同一個conflict zone中,不同vehicle進入的先後順序(包含type2、type3),如此一來,可以涵蓋出rule2,3,5的處理方式,因此好處是可以簡化只考慮2個rule

- 1.如同大多車廠的看法,即使賓士有推出level3,我也認為level3中目前汽車與人的權責區分太模糊,除非能制定出一套固定的準則來定義,但由於真實情況的變化太多,所以應該很難制定完整的規範,且太過頻繁的控制權切換,也會有較差的使用者體驗,喪失原本自動駕駛的意義。
- 2.我認為level4要能真正使用需要能接受即使在被允許的環境下,仍有可能完全停止,而由於有特定的環境限制,故我認為已現今的狀況能夠達成,但另一個需要考量的是車廠是否願意為發生的事故負責。
- 3.我認為level 5是無法達成的,因為當假如路直接斷掉,則車輛必須停止無法使用自動駕駛,或像是當鏡頭可能被外界的東西蓋住,則若沒有人為去除也會無法使用。