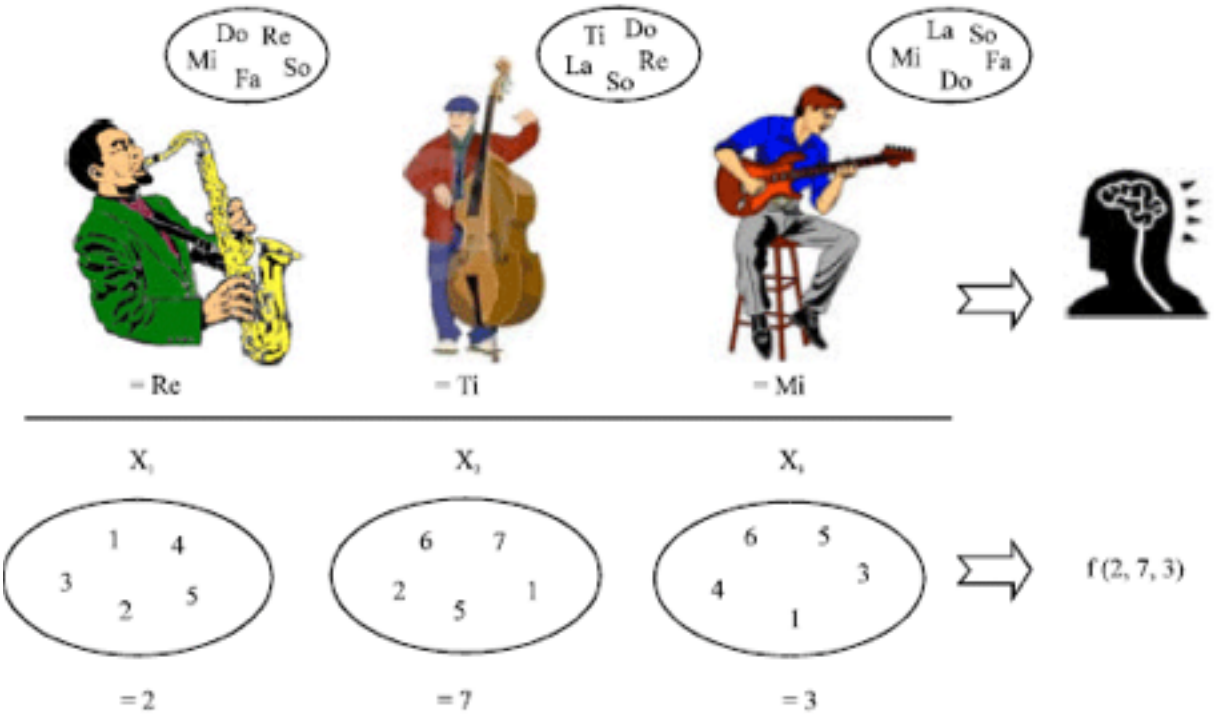
Harmony Search for Traveling Salesman Problem and Continuous Optimization Problem

工工所碩一 **R03546012** 許芸瑄 柔性演算法與應用期末報告

2015年1月22日



Introduction

和聲搜尋法(Harmony Search, HS)最早是由 Geem 等人 (2001)所提出，以模擬樂團中多位樂器演奏者之即興演奏，並將其演奏調整至最協調且最美妙的現象，即最佳化的狀態，而發展出的一種新的全域式搜尋法。

和聲搜尋法之發展基礎，需先設定多名使用不相同樂器之演奏者，各發出一個音符，希望形成最優美曼妙的旋律組合，而每個演奏者心中擁有各自最符合理想的音符，在嘗試多種組合後，期盼找尋到最為唯美的和弦。 比較圖如下表：

Comparison Factor	Optimization Process	Performance Process
Best state	Global Optimum	Fantastic Harmony
Estimated by	Objective Function	Aesthetic Standard
Estimated with	Value of Variables	Pitches of Instruments
Process unit	Each iteration	Each Practice

表一、和聲法與優化問題比較

和聲搜尋法為跳脫區域解，在組成新的可行解時，將分成三種不同方式決定決策變數之數值，分別如下所述：

1. 在一定和聲記憶考量機率(Harmony Memory Considering Rate, HMCR)內，於和弦記憶空間中取得決策變數數值。

2. 經由和弦記憶空間中取得決策變數之數值，再以一定調音機率(Pitch Adjusting Rate, PAR)決定此數值是否進行調整。

3. 在 HMCR以外之機率選取不屬於和弦記憶空間之數值為決策變數，此數值必需滿足該決策變數之可行範圍。

本實驗將進行和聲演算法在旅行銷售員問題(Traveling Salesman Problem, TSP)以及連續優化問題(Continuous Optimization Problem, COP)，並分別於蟻拓法、粒子群'算法比較。

Harmony Search

Step 1. 設定參數與母體初始化

如同多數萬用啟發式演算法，第一步為設定參數;在此演算法應用問題中，本研究需要設定和聲記憶數量(Harmony memory size, HMS)、考量機率,與迭代停止條件等。

隨機產生多個解(樂器數),並計算其適應值 (objective value)大小，和聲記憶(HM)即為母體，由多組解組成。

Step 2. 全域搜尋

使用者將自行設定一個具有演算法中全域搜尋特性的和聲記憶考量機率(Harmony memory consideration rate, HMCR)的閥值，用來控制新的解(X^{new})，X^{new} 可由HM 中選取或由 S_i (所有可行解)中隨機選取：

$$X_i^{new} = \begin{cases} X_i \in random\{X_i^1, X_i^2, ..., X_i^{HMS}\} if\ rnd_1 < HMCR \\ X_i \in S_i\ if\ rnd_1 > HMCR \end{cases}$$

藉由隨機產生器產生一個 0 到 1 的均勻分佈亂數 rnd_1 , 當 rnd_1 小於 HMCR 時,代表變數X_i 從 HM 裡隨機挑選，挑選公式如下:

$$j = \text{int}(rnd_2 \times HMS) + 1$$
$$X_i^{new} = X_i^j$$

j 代表 HM 中第 j 組的變數組合，int(rnd_2 ×HMS)代表均勻分佈亂數rnd_2 乘上 HMS 後，再以無條件捨去法將小數點後的數字捨去得到一個整數值，此時設計變數 X^j 值即為 HM 中第 ji 組的解,將 X^j 設為新的解。

若 rnd_1 大於或等於 HMCR，X_i 則是從設計變數可選用之集合 S_i 中隨機挑選，S_i 中共有N組 (i =1,2.....N)設計變數如下：

$$j = \text{int}(rnd_2 \times N) + 1$$
$$X_i^{new} = S_i(j)$$

亦即設計變數 X^{new} 為新的變數,藉由內插的方式，由集合 S 中選取其值。

另外，使用者將自行定義一個具有演算法中本地搜尋特性的調音機率(Pitch adjustment rate, PAR)，用來控制由上述步驟選出來的 X^{new} 值是否進行局部調音,也就是本地搜尋

$$X_i^{new} = \begin{cases} adjusting\ pitch\ if\ rnd_3 < PAR \\ doing\ nothing\ if\ rnd_3 < PAR \end{cases}$$

r_3 小於PAR時,表示 X_i 值須做局部調整，此時便在 X_i 值的鄰近範圍值進行挑選，以獲得新的 X_i 值。

Step 3. 更新

經過和聲記憶考量與調音等程序產生新的和聲變數組合後，計算其適應函數值大小並與和聲記憶庫中的和聲比較,若優於原有和聲記憶庫中最差的變數組合，則執行過濾動作且替換位置，反之則捨棄不用。

Step 4. 停止

如果未達到停止條件，再回到**Step 2.**。

System Implementation

Data structures

變數名稱	用途
numberOfHM	初始解的組數(使用者設定)
numberOfInstrument	變數個數
evaluation[]	每組解的objective value
HM[][](harmony memory)	紀錄每組解的每個變數
BestID	紀錄最佳解ID
WorstID	記錄最差解ID
HMCR	和聲記憶考量機率(使用者設定)
PAR	調音機率(使用者設定)

表二、資料結構

Software Structure

- TSP問題

MainForm for TSP Problem

---

begin

OpenTSPProblem
Reset, define initial HM and HMCR, PAR
while(t < Max number of iterations)
Generate new harmonics by accepting best harmonics
Adjust pitch to get new harmonics (solutions)

```
            if (rand < HMCR), choose an existing harmonic randomly
                  if (rand > PAR), adjust the pitch randomly within limits
            else generate new harmonics via randomization
            end if
            Accept the new harmonics (solutions) if better
      end while
end
```

(註：TSP限制為不允許重複路徑)

- COP問題

MainForm for COP Problem

```
begin
      OpenCOPProblem
      Reset, define initial HM and HMCR, PAR
      while(t < Max number of iterations)
            Generate new harmonics by accepting best harmonics
            Adjust pitch to get new harmonics (solutions)
            if (rand < HMCR), choose an existing harmonic randomly
                  if (rand > PAR), adjust the pitch randomly within limits
            else generate new harmonics via randomization
            end if
            Accept the new harmonics (solutions) if better
      end while
end
```

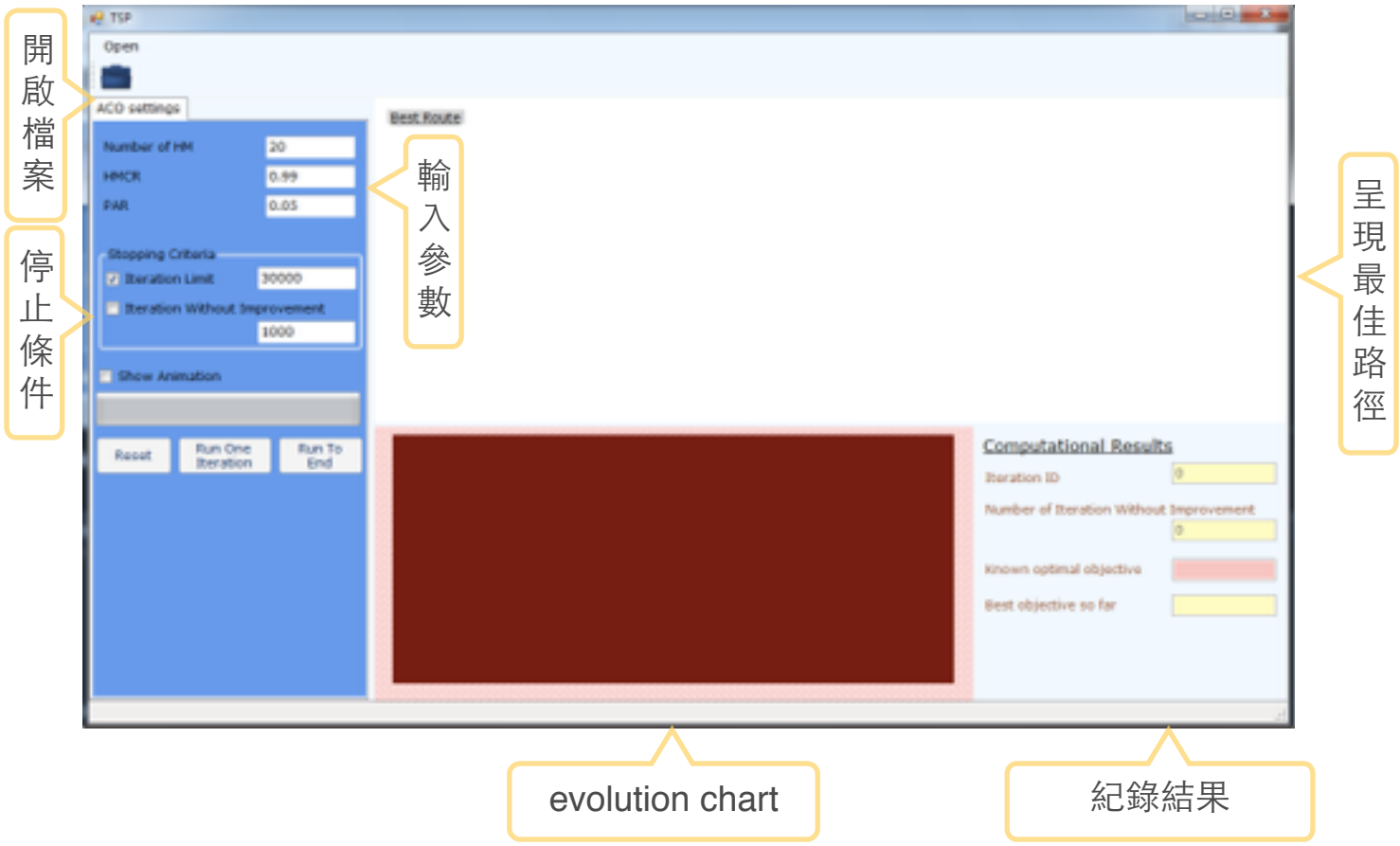
Algorithms

Harmony Search

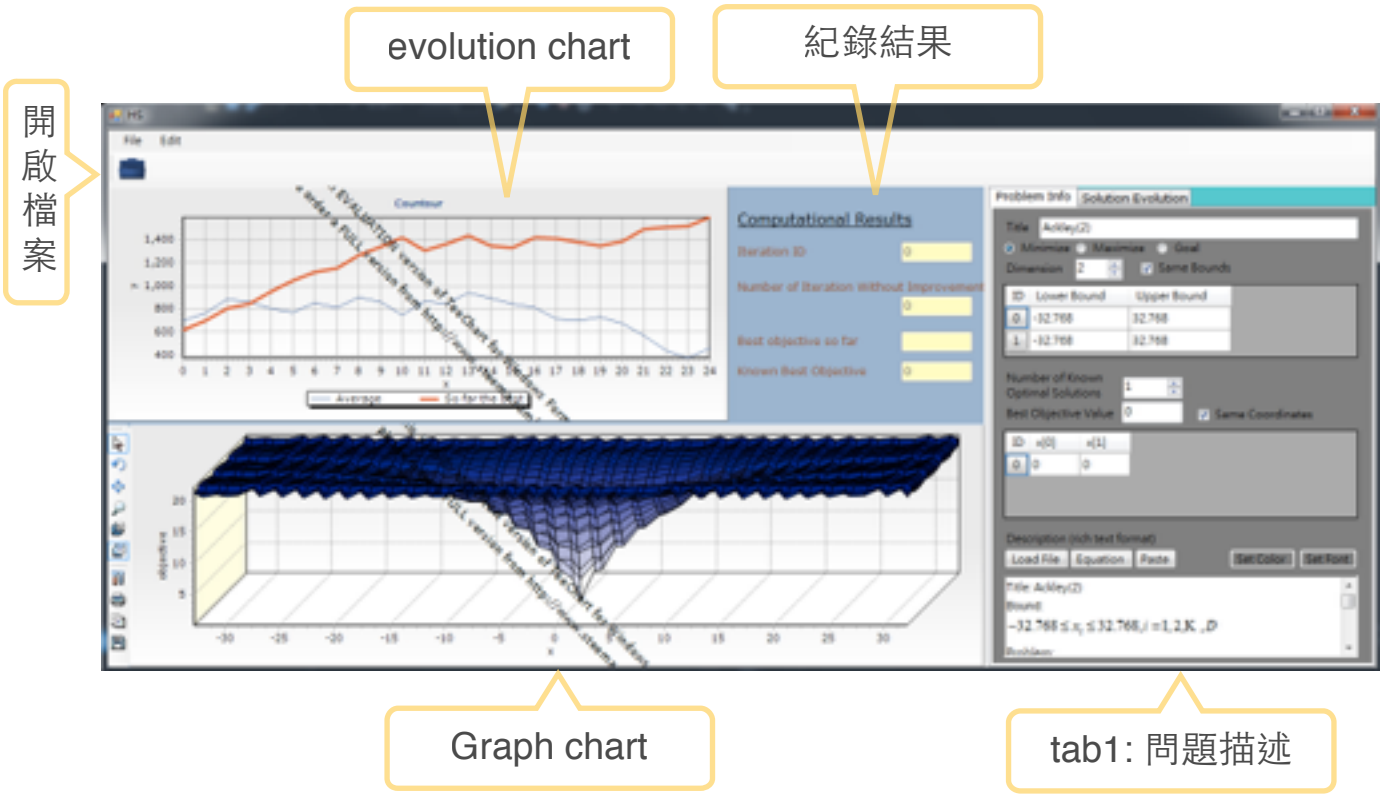
```
begin
      Generate initial HM(numberOfHM, numberOfInstrument)
      UpdateHM(NewVector, EvaluationOfNew Vector)
      ComputePenalty(penalty), if needed
      Sort, update worth HM and best HM
end
```

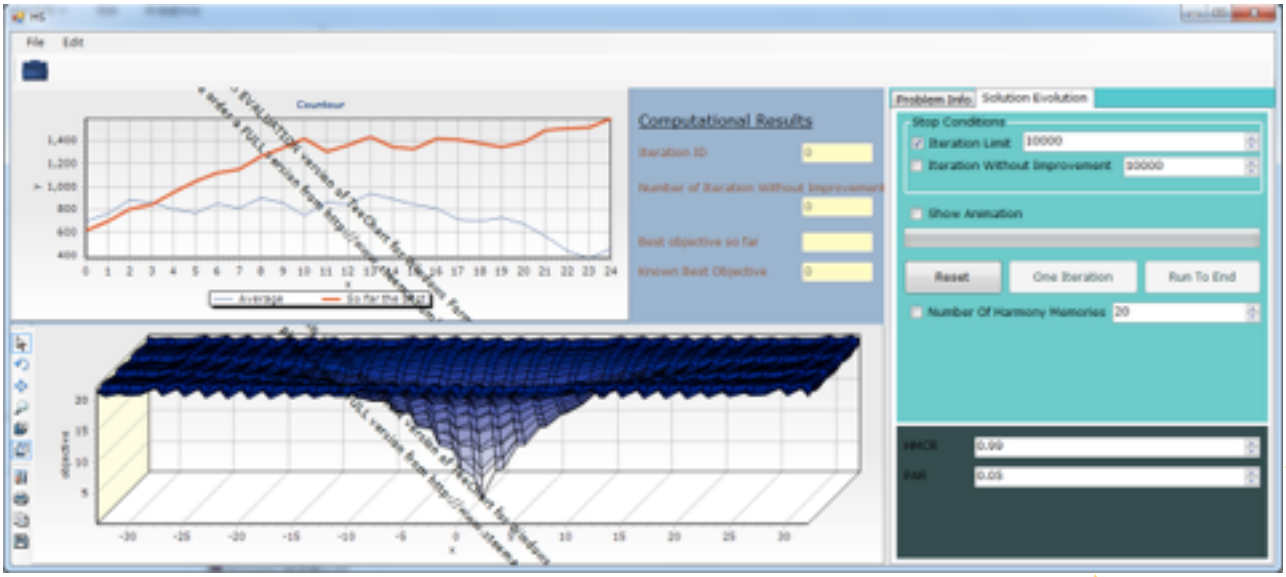
Interface Design

- TSP問題



- COP問題





System Execution

- TSP問題(Taiwan50)下的參數比較(function call=30000)

HMCR	0.95	0.9	0.99	0.99	0.95	0.9	0.5
PAR	0.1	0.1	0.1	0.5	0.5	0.5	0.5
	1668	1936	1829	1507	1692	1820	3681
	1600	1699	1442	1923	1807	1917	3725
	1287	1610	1757	1561	1810	1832	3444
	1748	1738	1509	1496	1531	1827	3486
	1506	1580	1777	1845	1728	1809	3532
	1701	1688	1409	1805	1906	1993	3738
	1499	1815	1462	1879	2171	2202	3717
	1675	1392	1942	1982	1755	1969	3426
	1408	1598	1896	1511	1591	2000	3732
	1437	1581	1794	1824	1581	1997	3757
Average	1552.9	1663.7	1681.7	1733.3	1757.2	1936.6	3623.8
Standard deviation	149.2979497	148.7959453	203.4852165	191.9311509	186.6617857	122.7238997	134.8083413

表三、各參數下 HS 在 TSP 問題的表現

- COP問題(Ackley(2))下參數比較(function call=30000)

HMCR	0.95	0.9	0.99	0.99	0.95	0.9	0.5
PAR	0.1	0.1	0.1	0.5	0.5	0.5	0.5
	0.03	0.08	0.58	0.16	0.16	0.19	0.007
	0.04	0.004	0.66	2.77	0.03	0.03	0.01
	0.1	0.03	1.7	2.53	0.12	0.05	0.003
	0.05	0.15	0.38	1.47	0.05	0.1	0.002
	0.12	0.07	0.61	0.32	0.06	0.05	0.03
	0.07	0.02	0.18	1.31	0.13	0.09	0.01
	0.1	0.03	1.5	1.91	0.14	0.05	0.06
	0.24	0.09	0.19	1.07	0.04	0.1	0.06
	0.02	0.07	1.76	1.45	0.17	0.12	0.01
	0.24	0.12	0.21	1.68	0.43	0.08	0.03
Average	0.101	0.0664	0.777	1.467	0.133	0.086	0.0222
Standard deviation	0.080201136	0.046339808	0.632579376	0.835052227	0.1164331	0.0464758	0.022195095

表四、各參數下 HS 在 COP 問題的表現

- TSP問題（與蟻拓法(ACO)比較）

參數設定：

Harmony Search	Ant Colony Optimization
number of HM = 20	number of ants = 20
HMCR = 0.99	pheromone factor = 1.0
PAR = 0.05	heuristic factor = 3.0
	initial pheromone value = 0.05
	evaporate rate = 0.95
	dropping amount = 0.01

表五、(TSP問題) HS及ACO參數設定

Best Objective（Function call=6000)				
Sample	Taiwan50		Taiwan400	
Iteration	HS	ACO	HS	ACO
1	2025	1076	20133	3345
2	2105	1147	19686	3120
3	1802	1111	20592	3242
4	1844	1130	20379	3233
5	1759	1105	21257	3019
6	1599	1109	21465	3330
7	1622	1186	20887	3345
8	1914	1166	20861	3230
9	1993	1144	19804	3400
10	1792	1093	21175	3291
Average	**1845.5**	**1126.7**	**20623.9**	**3255.5**
Standard Deviation	**166.32**	**34.19**	**613.85**	**115.15**

表六、(TSP問題)HS及ACO針對不同維度的實驗數據

- COP問題（與例子群演算法(PSO)比較）

參數設定

Harmony Search	Particle Swarm Optimization
number of HM = 20	number of particle = 20
HMCR = 0.99	friction factor = 0.5
PAR = 0.05	self factor = 0.5
	social factor = 0.5

表七、(COP問題) HS及PSO參數設定

Best Objective(Function call = 6000)						
Sample	Ackley(2) (best objective = 0)		Peak(2) (best objective = -6.55)		Ackley(30) (best objective = 0)	
Iteration	HS	PSO	HS	PSO	HS	PSO
1	4	0	-6.52	-6.55	4.44	18.13
2	3.28	0	-6.44	-3.05	5.4	16.5
3	1.5	0	-6.39	-6.55	4.98	17.49
4	3.99	0	-6.52	-6.55	5.36	18.93
5	4.52	0	-6.44	-6.55	5.62	18.68
6	1.52	0	-6.54	-6.55	5.4	17.5
7	3.73	0	-5.98	-6.55	4.51	18.26
8	0.2	0	-6.51	-6.55	6.55	17.8
9	2.87	0	-6.55	-6.55	6.3	18.32
10	3.5	0	-6.38	-3.05	5.47	17.85
Average	2.911	0	-6.427	-5.85	5.403	17.946
Standard Deviation	1.38924	0	0.16859221	1.475729575	0.673086588	0.691828495

表八、(COP問題)HS及PSO針對不同類型、不同維度的實驗數據

Discussion

對於 TSP 問題：本次期末報告利用兩種演算法(HS、ACO)進行不同維度的測試，每組進行十次實驗，從表六中可以發現，無論是小型問題(城市數：50)或大型問題(城市數：400)，ACO 較 HS 來的好，可能原因是HS面對限制式沒有明確的計算方。另外值得一提的是，在大型問題中，雖然 ACO 的表現明顯較HS好，但 ACO 運行時間是 HS 的好幾倍。

在 COP 問題：本次期末報告利用兩種演算法(HS、PSO)對不同類型(Ackley(2)、Peak(2))、不同維度(Ackley(2)、Ackley(40))的標竿問題進行求解。每個配對組合試行十次，從表八我們可以發現當求解低維度的問題時，兩種演算法普遍都能找到全域最佳解,其中 PSO 擁有較佳的收斂速度，HS 略差一點。在 Peak(2)的問題中，HS平均表現較PSO表現好，但PSO卻有比較高的機率找到全域最佳解。 然而在面對高維度的問題時，HS 的收斂速度遠比 PSO 快，得到的解也比 PSO 來得好。

Conclusion

總體來說，和聲演算法利用HMCR及PAR這兩個參數增加解的隨機性，的確能夠避免掉數區域最佳解且趨近最佳解，在解高維度連續優化問題(COP)的結果最好，甚至優於粒子群演算法(PSO)，但由於對於限制式的定義在此篇中不夠明確，在此篇中採用懲罰法(Penalty)，造成 TSP 結果不佳，在未來可以參考其他篇文獻所提供對限制式的策略來改進和聲演算法解存在限制式的問題。

Reference

1. Zong Woo Geem and Joong Hoom Kim, “A New Heuristic Optimization Algorithm: Harmony Search”, Simulation 76:2, 60-68