A competitive memetic algorithm for the distributed two-stage assembly flow-shop scheduling problem

Jin Deng, Ling Wang, Sheng-yao Wang & Xiao-long Zheng(2015)

Zhou Yidong

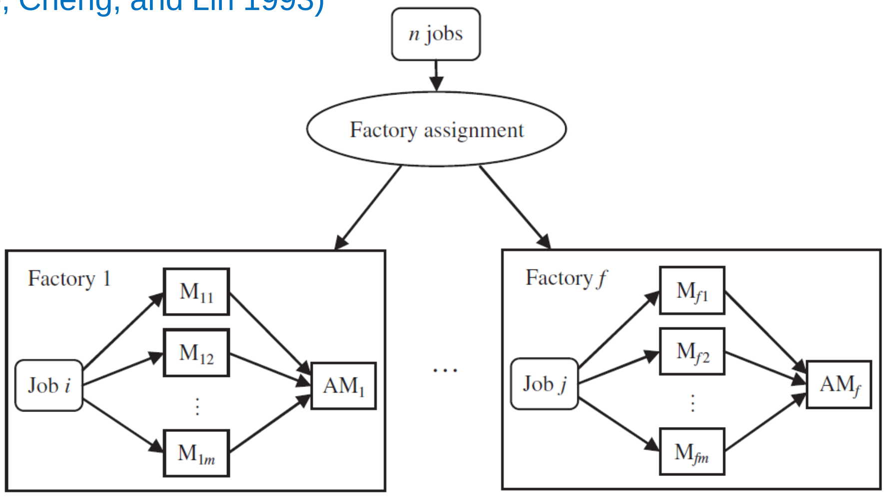
2016.01.03

Contents

Introduction

❖ Problem review

- Distributed two-stage assembly flow-shop scheduling(DTSAFSP)
  - *f* identical factories
    - ✓ First stage: Elements processed on parallel machines
    - ✓ Second stage: A single assembly machine
  - *n* jobs need to be processed
    - ✓ Factory assignment
    - ✓ Sequencing
- Np-hard problem
  - *f* = 1 is two stage assembly flow-shop scheduling problem(TSADSP)
    - ✓ TSADSP with makespan is np-hard(Lee, Cheng, and Lin 1993)



Introduction

❖ Literature review

Problem	Description	Authors
DSSP	a decomposition-based method by hybridising variable neighbourhood search (VNS), tabu search (TS)	Behnamian (2014)
	6 IP models to fomulated distributed permutation flowshop problem	Naderi and Ruiz(2010)
	an adaptive genetic algorithm (GA)	Chan, Chung, and Chan (2005)(2006)
DTSAFSP	developed a reduced VNS (RVNS) to perform local search withn makespan objective	Xiong and Xing (2014)
	GA-VNS with makespan and total completion time objective	Xiong et al. (2014)

- Performance comparison with Xiong and Xing (2014) and Xiong et al. (2014)

Introduction

❖ Problem description

- Problem
  - Distributed two-stage assembly flow-shop scheduling(DTSAFSP)
- Objective
  - Minimum makespan
- Decision variable
  - Factory assignment
  - Sequencing in each factory
- Algorithms
  - A competitive memetic algorithm
- Assumption
  - Permutation scheduling
  - Factories identical
  - Independent setup time

MILP Model

❖ Notations

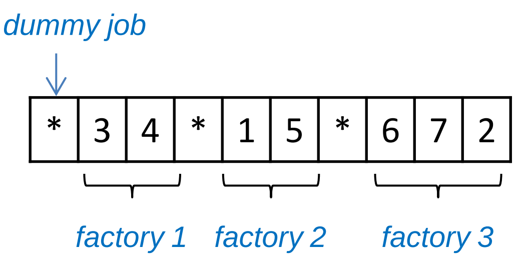
- n the total number of jobs
- m the total number of machines at the first stage
- f the total number of factories
- $S_{1,ik}$ the set-up time of job *i* on machine *k* at the first stage
- $S_{2,i}$ the set-up time of job *i* on assembly machine at the second stage
- $p_{i,k}$ the processing time of job *i* on machine *k* at the first stage
- t_i the assembly time of jobs job *i* on assembly machine
- L a very large positive number
- X_{ij} a binary variable that equals to 1 if job *j* is processed immediately after job *i* and 0 otherwise
- $C_{i,k}$ the completion time of job *i* on machine *k* at the first stage
- CA_i the completion time of job *i* on assembly machine
- C_{max} the makespan

MILP Model

❖ MILP Model

$$\text{Minimise} \quad C_{\max} \tag{1}$$

Subject to:



$$\sum_{i=0, i \neq j}^{n} X_{i,j} = 1 \quad \forall j = 1, 2, \ldots, n \tag{2}$$

$$\sum_{j=0, j \neq i}^{n} X_{i,j} \leq 1 \quad \forall i = 1, 2, \ldots, n \tag{3}$$

Jobs precedence and succeeding relations

$$\sum_{j=1}^{n} X_{0,j} = f \tag{4}$$

$$\sum_{i=1}^{n} X_{i,0} = f - 1 \tag{5}$$

Dummy jobs appeared times

$$X_{i,j} + X_{j,i} \leq 1 \quad \forall i = 1, 2, \ldots, n-1, \; j > i \tag{6}$$

$$C_{j,k} \geq C_{i,k} + S_{1,jk} + p_{j,k} - L(1 - X_{i,j}) \quad \forall i = 0, 1, \ldots, n; \; j = 1, 2, \ldots, n, \; j \neq i; \; k = 1, 2, \ldots, m \tag{7}$$

Precedence relations for the first stage

$$CA_j \geq CA_i + s_{2,j} + t_j - L(1 - X_{i,j}) \quad \forall i = 0, 1, \ldots, n; \; j = 1, 2, \ldots, n, \; j \neq i \tag{8}$$

$$CA_i \geq C_{i,k} + t_i \quad \forall i = 1, \ldots, n; \; k = 1, 2, \ldots, m \tag{9}$$

Precedence relations for the second stage

$$C_{\max} \geq CA_i \quad \forall i = 1, 2, \ldots, n \tag{10}$$

$$X_{i,j} \in \{0, \; 1\} \quad \forall i, j = 0, 1, \ldots, n \tag{11}$$

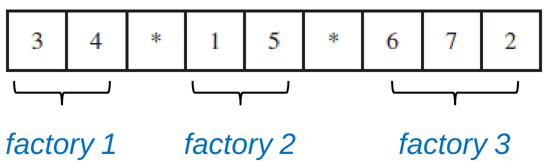
$$C_{i,k} \geq 0 \quad \forall i = 1, 2, \ldots, n; \quad \forall k = 1, 2, \ldots, m \tag{12}$$

$$C_{0,k} = CA_0 = 0, \quad \forall k = 1, 2, \ldots, m \tag{13}$$

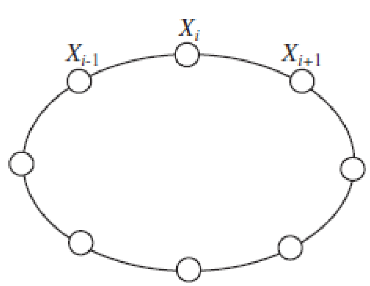
Competitive Memetic Algorithm

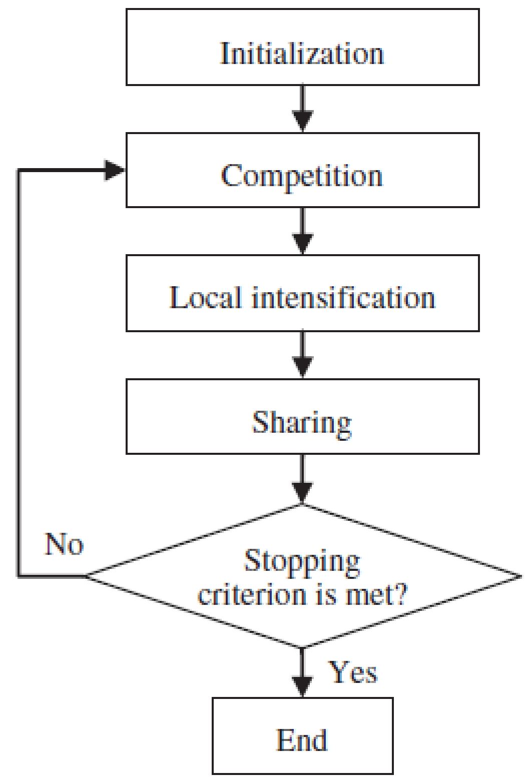
❖ Competitive Memetic Algorithm(CMA)(1)

- Encoding scheme
  - Individual X_i is represented by a string,contains:
    - ✓ all jobs for sequencing
    - ✓ *f*-1 asterisks for factory assignment
  - Example:



- Initialization and neighbourhood structure
  - All population size initial individuals are generated randomly
  - Neighbourhood structure
    - ✓ ring structure
    - ✓ each individual has two adjacent neighbours
  - Figure:





Flow chart of the CMA

Competitive Memetic Algorithm

❖ Competitive Memetic Algorithm(CMA)(2)

- Competition
  - Elite competition with its two neighbours, better than neighbours than regarded as winner
  - Parallel search for global exploration
  - Insert and swap operators
    - ✓ Exter-factory insert

3	4	*	1	7	5	*	6	2	→	3	4	*	1	5	*	6	7	2

Longer makespan *Shorter makespan*

- ✓ Exter-factory swap

3	4	*	1	7	5	*	6	2	→	3	4	*	1	6	5	*	7	2

- ✓ Inter-factory swap

3	4	*	1	7	5	*	6	2	→	3	4	*	5	7	1	*	6	2

- For every winner three operators are performed one by one

Competitive Memetic Algorithm

❖ Competitive Memetic Algorithm(CMA)(3)

- Local intensification
  - Main idea: modifying the schedule in the factory with largest completion time
  - Three local intensification operators
    - ✓ LS1

      Randomly select a job from F_c and another factory $F_k (k \neq c)$ Remove the job from F_c and insert it into all possible positions in F_k, and choose the best (F_c the factory with largest completion time)
    - ✓ LS2

      Randomly select a job from F_c and another job from another factory $F_k (k \neq c)$, exchange two jobs
    - ✓ LS3

      Randomly select two jobs from F_c, swap two jobs
  - Using three operators ono by one with local search times for the best individual of the population

Competitive Memetic Algorithm

❖ Competitive Memetic Algorithm(CMA)(4)

- Sharing
  - Main idea: better individuals can guide worse ones to the promising regions (similar to GA algorithm's crossover operator)
  - For every loser X_i, denote better neighbour as X_j, order crossover is used to update X_i

Cross point   Cross point

X_i	3	4	*	1	7	5	*	6	2
X_j	6	7	2	*	5	1	3	*	4
Tentative O_1				*	5	1	3	*	
Tentative O_2				1	7	5	*	6	
O_1	4	7	6	*	5	1	3	*	2
O_2	2	*	3	1	7	5	*	6	4

Step 1: Randomly generate two cross points.

Step 2: O_1 and O_2 inherit the sub-sequences between the two cross points from X_j and X_i.

Step 3: From the position after the second cross point of X_i, delete the elements that appear in the sub-sequence of X_j ; and then fill the remaining elements into O_1 from the second cross point one by one. Similarly, produce O_2 by exchanging the roles of X_i and X_j

Computational Results And Comparisons

❖ Data generation and parameter setting

- Range of data

Problem parameters	Small scale	Large scale
f	{2, 3, 4}	{2, 3, 4, 5, 6}
n	{8, 10, 12}	{20, 50, 100, 200, 500}
m	{2, 3, 4}	{2, 4, 6, 8}
$S_{1,ik}$	U(1, 20)	U(1, 20)
$S_{2,i}$	U(1, 20)	U(1, 20)
$p_{i,k}$	U(1, 100)	U(1, 100)
t_i	U(1, 100)	U(1, 100)

- Stopping criterion: CPLEX 3600s ; CMA $0.1 \times n$s.

- Parameter setting
  - Population size(PS) and local search(LS) times
  - 4 factor levels(one-way ANOVA analysis)

	Factor levels			
Factor	1	2	3	4
PS	10	30	50	70
LS	10	50	100	150

  - One-way ANOVA analysis results
    - ✓ PS = 70, LS = 50 (small-scale)
    - ✓ PS = 30, LS = 50 (large-scale)

Computational Results And Comparisons

❖ Computational results of the CMA and CPLEX

- Results of CPLEX and the CMA for small-scale instances

Instance	f	n	m	CPLEX Makespan	CPLEX Time (s)	CMA Best	CMA Ave	CMA SD	CMA Time (s)
1	2	8	2	282*	1.70	282	282.00	0.00	0.80
2	2	8	3	290*	1.92	290	290.00	0.00	0.80
3	2	8	4	346*	5.52	346	346.00	0.00	0.80
4	2	10	2	340*	288.32	340	340.05	0.22	1.00
5	2	10	3	425*	641.15	425	425.00	0.00	1.00
6	2	10	4	378	3600	374	374.25	0.45	1.00
7	2	12	2	382	3600	382	382.10	0.44	1.20
8	2	12	3	416	3600	414	414.15	0.49	1.20
9	2	12	4	401	3600	401	401.00	0.00	1.20
10	3	8	2	248*	0.47	248	248.00	0.00	0.80
11	3	8	3	277*	0.87	277	277.00	0.00	0.80
12	3	8	4	262*	0.70	262	262.00	0.00	0.80
13	3	10	2	255*	16.66	255	255.00	0.00	1.00
14	3	10	3	299*	140.48	299	299.20	0.61	1.00
15	3	10	4	299*	62.87	299	299.00	0.00	1.00
16	3	12	2	234	3600	234	234.20	0.89	1.20
17	3	12	3	341	3600	341	341.00	0.00	1.20
18	3	12	4	338	3600	338	338.15	0.36	1.20
19	4	8	2	170*	0.25	170	170.00	0.00	0.80
20	4	8	3	216*	0.33	216	216.00	0.00	0.80
21	4	8	4	268*	0.33	268	268.00	0.00	0.80
22	4	10	2	232*	4.84	232	232.00	0.00	1.00
23	4	10	3	242*	6.88	242	242.00	0.00	1.00
24	4	10	4	228*	1.14	228	228.00	0.00	1.00
25	4	12	2	246*	219	246	246.20	0.41	1.20
26	4	12	3	264*	573.68	264	264.00	0.00	1.20
27	4	12	4	251	3600	251	251.00	0.00	1.20

- ✓ CPLEX and the CMA obtain the same results on 25 instances including 19optimal solutions.
- ✓ The CMA outperforms CPLEX on instances #6 and #8.

- The CMA is more effective than CPLEX in solving the small-scale problem

Computational Results And Comparisons

❖ Comparison of the CMA and existing algorithms(1)

- Results of the algorithms for large-scale instances(f = 2)

Instance	CMA			VNS			Hypothesis test (CMA vs. VNS)		GA-RVNS			Hypothesis test (CMA vs. GA-RVNS)	
	Best	Ave	SD	Best	Ave	SD	*p*-Value	Sig	Best	Ave	SD	*p*-Value	Sig
F2_1	**686**	**686.35**	0.77	**686**	686.50	1.00	0.62	N	**686**	700.05	9.77	0.000	Y
F2_2	**711**	**711.10**	0.09	**711**	711.20	0.52	0.47	N	**711**	715.15	2.32	0.000	Y
F2_3	**773**	**773.30**	0.22	**773**	773.05	0.22	0.04	Y	**773**	780.10	7.77	0.001	Y
F2_4	**731**	**731.85**	0.56	**731**	732.70	1.45	0.03	Y	**731**	743.95	9.61	0.000	Y
F2_5	**1517**	**1517.20**	0.17	**1517**	1521.15	2.28	0.00	Y	1522	1529.95	9.19	0.000	Y
F2_6	**1598**	**1599.10**	1.15	**1598**	1602.55	3.49	0.00	Y	1599	1613.75	14.05	0.000	Y
F2_7	**1764**	**1764.05**	0.05	**1764**	1764.50	0.69	0.01	Y	**1764**	1766.95	2.84	0.000	Y
F2_8	**1607**	**1609.55**	1.73	1608	1616.25	4.91	0.00	Y	1611	1626.75	14.36	0.000	Y
F2_9	**3228**	**3228.00**	0.00	**3228**	3238.80	8.35	–	–	3231	3252.60	14.17	–	–
F2_10	**3141**	**3141.10**	0.20	**3141**	3144.50	2.04	0.00	Y	**3141**	3145.15	4.04	0.000	Y
F2_11	**3198**	**3198.35**	0.45	**3198**	3202.15	2.74	0.00	Y	**3198**	3204.25	5.01	0.000	Y
F2_12	**3414**	**3414.00**	0.00	**3414**	3414.65	0.67	–	–	**3414**	3418.75	4.58	–	–
F2_13	**6244**	**6244.00**	0.00	**6244**	6244.25	0.44	–	–	**6244**	6249.65	5.01	–	–
F2_14	**6431**	**6431.00**	0.00	**6431**	6431.85	0.93	–	–	6432	6437.15	5.70	–	–
F2_15	**6588**	**6588.00**	0.00	**6588**	6588.30	0.57	–	–	**6588**	6595.15	6.08	–	–
F2_16	**6162**	**6162.15**	0.13	**6162**	6164.90	2.83	0.00	Y	**6162**	6170.60	11.60	0.004	Y
F2_17	**15,403**	**15403.25**	0.30	15,406	15417.20	8.81	0.00	Y	15,408	15425.50	14.55	0.000	Y
F2_18	**15,683**	**15683.30**	0.54	15,684	15697.50	8.79	0.00	Y	15,684	15707.00	12.82	0.000	Y
F2_19	**15,598**	**15598.05**	0.05	**15,598**	15599.80	1.25	0.00	Y	**15,598**	15601.50	3.80	0.001	Y
F2_20	**15,864**	**15864.05**	0.05	**15,864**	15866.10	1.86	0.00	Y	**15,864**	15871.80	7.99	0.000	Y

✓ Hypothesis test with 95% confidence interval :

- P-value smaller than 0.05, the difference between two algorithms is significant, and marked “Y”.

Computational Results And Comparisons

❖ Comparison of the CMA and existing algorithms(2)

- Results of the algorithms for large-scale instances(f = 6)

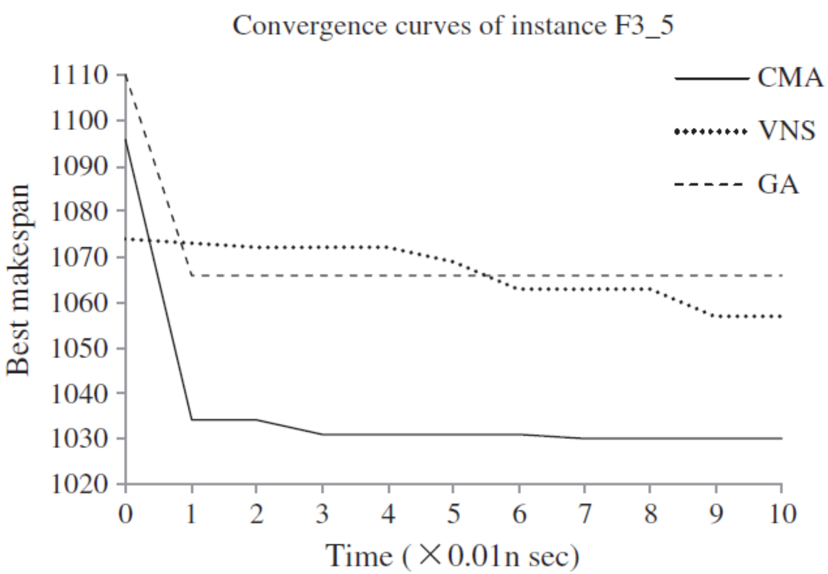
✓ f = 3,4,5 omitted

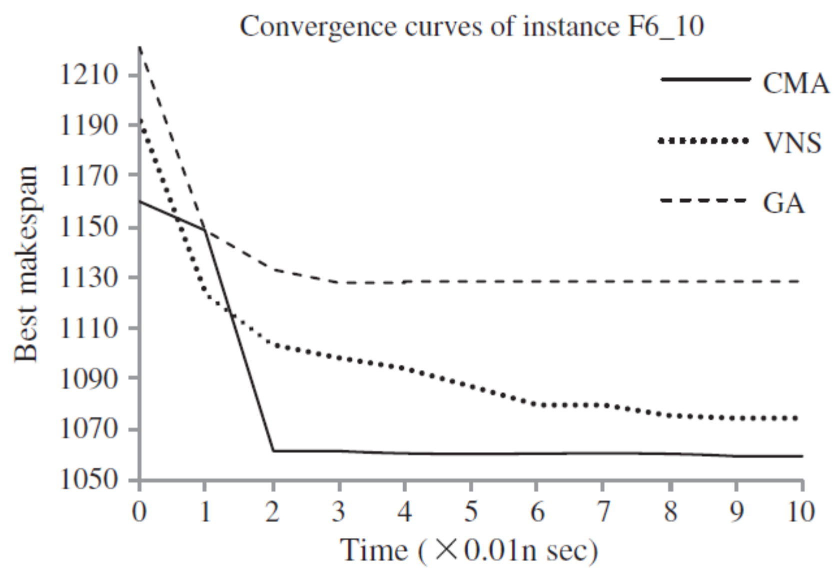
Instance	CMA			VNS			Hypothesis test (CMA vs. VNS)		GA-RVNS			Hypothesis test (CMA vs. GA-RVNS)	
	Best	Ave	SD	Best	Ave	SD	*p*-Value	Sig	Best	Ave	SD	*p*-Value	Sig
F6_1	**258**	**260.75**	1.25	262	280.95	13.95	0.00	Y	265	290.70	17.20	0.00	Y
F6_2	**268**	**270.35**	1.69	269	282.50	7.60	0.00	Y	277	293.95	14.04	0.00	Y
F6_3	**278**	**280.85**	1.87	285	298.15	10.58	0.00	Y	283	301.85	11.69	0.00	Y
F6_4	**268**	**271.35**	2.56	278	285.65	6.16	0.00	Y	278	294.90	11.57	0.00	Y
F6_5	**520**	**521.80**	1.54	538	551.95	7.00	0.00	Y	555	574.60	13.93	0.00	Y
F6_6	**563**	**563.55**	0.60	569	584.95	5.88	0.00	Y	586	604.75	13.56	0.00	Y
F6_7	**535**	**539.95**	2.56	559	572.70	9.55	0.00	Y	549	592.30	16.60	0.00	Y
F6_8	**581**	**586.60**	2.98	609	625.10	9.63	0.00	Y	617	638.15	14.79	0.00	Y
F6_9	**1125**	**1125.30**	0.57	1145	1157.90	6.66	0.00	Y	1153	1186.80	20.65	0.00	Y
F6_10	**1059**	**1060.05**	0.69	1072	1085.55	7.16	0.00	Y	1083	1113.20	20.06	0.00	Y
F6_11	**1069**	**1071.90**	1.92	1106	1115.60	6.87	0.00	Y	1100	1146.25	28.49	0.00	Y
F6_12	**1120**	**1121.60**	1.23	1153	1162.60	4.99	0.00	Y	1160	1192.50	26.77	0.00	Y
F6_13	**2021**	**2021.30**	0.98	2045	2063.60	11.27	0.00	Y	2067	2104.75	23.57	0.00	Y
F6_14	**2174**	**2174.15**	0.37	2184	2195.75	8.08	0.00	Y	2190	2225.45	37.16	0.00	Y
F6_15	**2130**	**2130.00**	0.00	2145	2155.45	6.50	–	–	2148	2192.40	24.43	–	–
F6_16	**2151**	**2151.35**	0.88	2177	2192.15	10.40	0.00	Y	2192	2229.10	25.33	0.00	Y
F6_17	**5166**	**5166.45**	0.51	5189	5206.50	10.71	0.00	Y	5203	5258.95	39.85	0.00	Y
F6_18	**5107**	**5107.05**	0.22	5136	5153.65	11.88	0.00	Y	5157	5223.95	47.76	0.00	Y
F6_19	**5176**	**5178.40**	1.79	5237	5257.95	12.43	0.00	Y	5277	5353.75	65.09	0.00	Y
F6_20	**5213**	**5214.80**	1.44	5258	5276.10	11.53	0.00	Y	5281	5363.50	69.94	0.00	Y

- ➢ The CMA significantly outperforms the VNS on 85 of 100 instances , the CMA significantly outperforms the GA-RVNS on 87 of 100 instances
- ➢ As the number of factories f increases, the superiority of the CMA over the VNS and GA-RVNS becomes more obvious

Computational Results And Comparisons

❖ Comparison of the CMA and existing algorithms(3)





➢ CMA is faster in solving the instances

- Computational results: The CMA is a more powerful algorithm for solving the large-scale DTSAFSP.

Conclusion

- **Distributed two-stage assembly flow-shop scheduling**
  - Makespan objective with setup time
  - Influence of parameter setting
  - Comparisons between CMA and CPLEX; CMA and the existing VNS and GA-RVNS
- **Suggested**
  - MILP model
  - CMA algorithms
- **Further research**
  - The design of new mechanisms for performing competition and sharing
  - The applications to other kinds of scheduling problems, including the multi-objective optimisation problems
  - Apply the algorithm to some real-world distributed scheduling problems(multi-factories car engine manufacturing)
- **Adv & Disadv**

THANK YOU