Part type selection problem in flexible manufacturing systems: tabu search algorithms

Annals of Operations Research(1993)
Bharatendu Sivastava and Wun-Hwa Chen

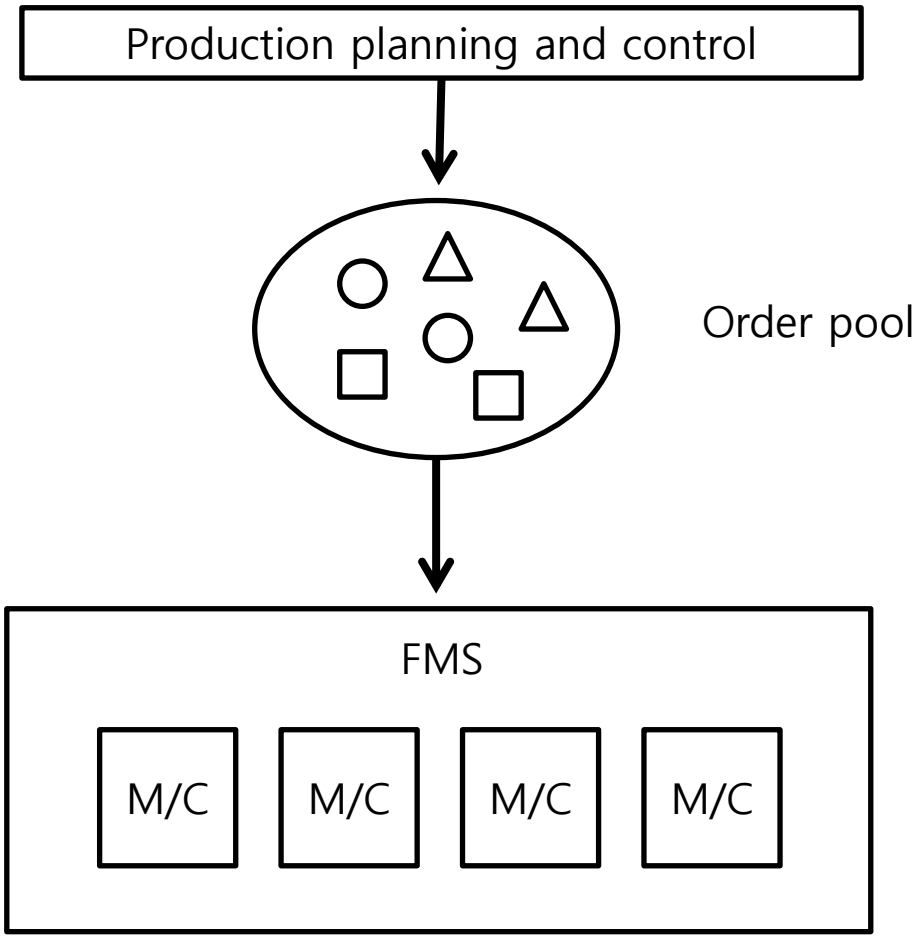
Kyung Cheol Jang
2016.01.28

INTRODUCTION

INDEX

Introduction

- **Part type selection problem**



Problem description

- **Problem**
  - Part type selection problem

- **Objective**
  - Maximizes the total weight of the part types selected in the batch

- **Decision variables**
  - Tool type selection
  - Batching assignment

- **Assumptions**
  - Only one tool can be used at a time, there is no tool duplication on any machine
  - There is no overlapping of tools on any machine
  - Each part type in the current batch must be processed completely for the entire order quantity
  - The quality of product is not considered

Problem description

- **Parameters**
  - n — number of part types
  - m — number of machine types
  - K — number of tool types
  - w_i — weight associated with part type i
  - g_k — number of tool slots required by tool type k
  - t_{ij} — required processing time for a set of operations, for the entire order quantity of part type i to be performed on machine type j
  - b_j — total processing time available on machine type j
  - h_j — tool magazine capacity of machine type j
  - K_{ij} — set of tool types required for part type i in machine type j

- **Decision variables**
  - x_i — 0-1 variable that has value 1 if part type i is selected in the current batch, and 0 otherwise
  - y_{kj} — 0-1 variable that has value 1 if tool type k is assigned to machine type j, and 0 otherwise

Problem description

- **Mathematical Formulation**
  - Maximizes the total weight of the part types selected in the batch

$$\max \sum_{i=1}^{n} w_i x_i \qquad (1)$$

$$\text{subject to} \sum_{i=1}^{n} t_{ij} x_i \leq b_j \quad \text{for j = 1,....m,} \qquad (2)$$

$$\sum_{i=1}^{n} g_k y_{kj} \leq h_j \quad \text{for j = 1,....m,} \qquad (3)$$

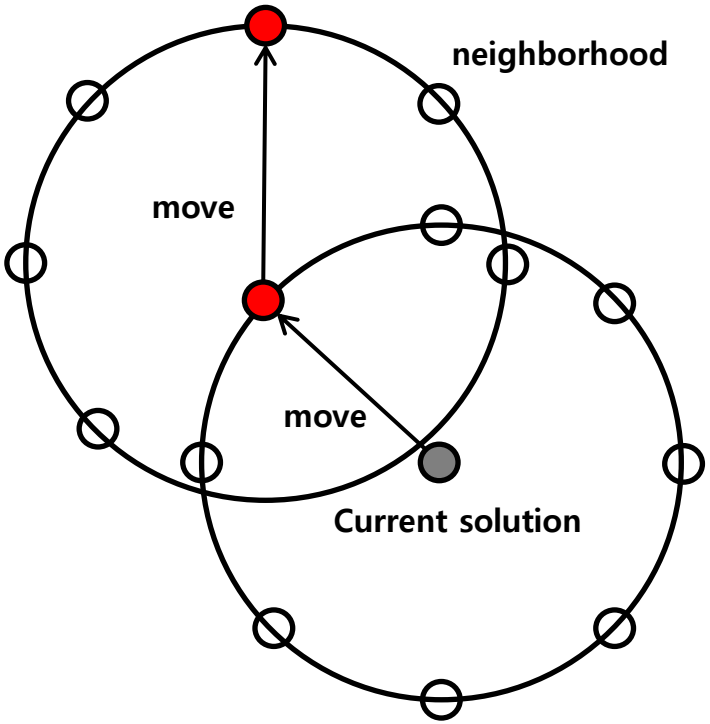
$$x_i \leq y_{kj} \quad \text{for i = 1,....n, j = 1,....m, and } k \in K_{ij} \qquad (4)$$

$$x_i = 0 \text{ or } 1 \quad \text{for i = 1,....n,} \qquad (5)$$

$$y_{kj} = 0 \text{ or } 1 \quad \text{for k = 1,....K, and j = 1,....m} \qquad (6)$$

Algorithm

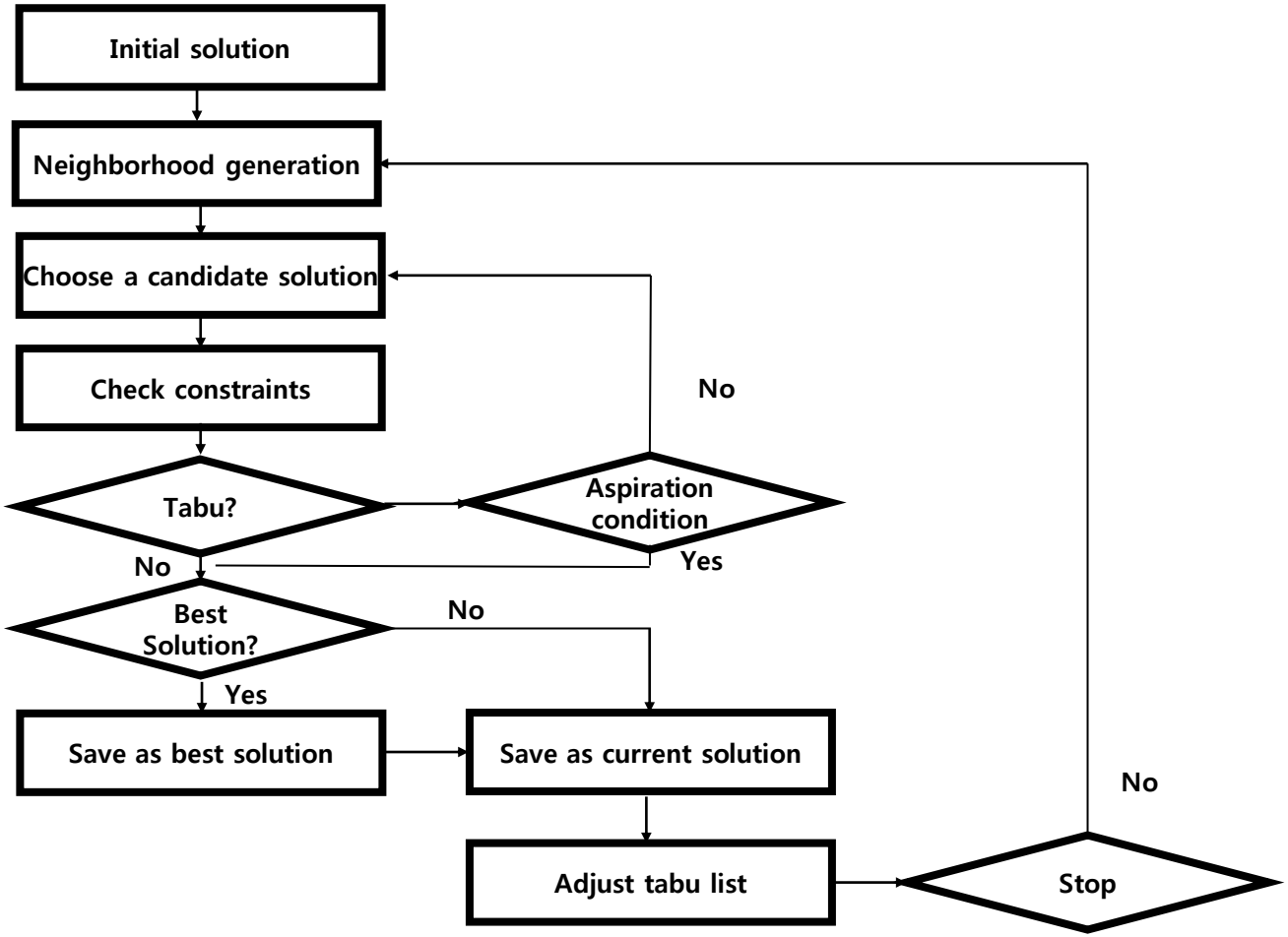
➢ Tabu search



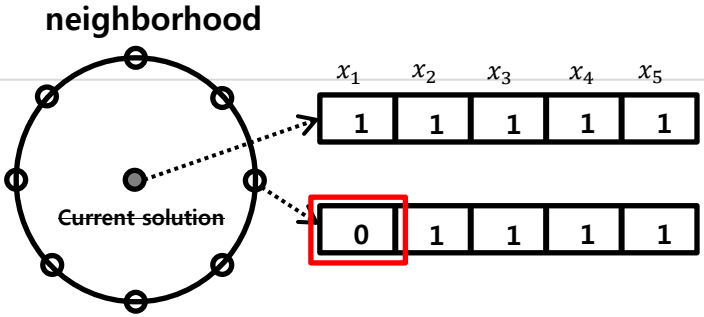
- The method explores the solution space by moving at each iteration from a solution s to the best solution in a subset of its neighborhood.
- Transition is done even though the best neighboring solution is wore than the given solution.
- Tabu list
  : To avoid cycling, solutions possessing some attributes of recently explored solutions are temporarily declared tabu or forbidden
- Aspiration level
  : a tabu move can be allowed if it create a solution better than the best solution obtained so far

3.Algorithm

2) Tabu search algorithm



3.Algorithm



2) Tabu search algorithm

✓ Neighborhood generation

- x = 1 - x

✓ Solution representation

x_1	x_2	x_3	x_4	x_5
1	1	1	1	1

x_1	x_2	x_3	x_4	x_5
0	1	1	1	1

✓ Move

- **Improving move**
  $w_{v_1} \geq w_{v_2} \geq \cdots\cdots \geq w_n$
- **Non improving move**
  $p_{u_1} \leq p_{u2} \leq \cdots\cdots \leq p_{un}$

$$p_i = w_i + \mu(freq_i)$$

✓ Termination condition

- Max_interation = 3n

Algorithm

- **Tabu List**
  - Tabu list size
    1) as a function of problem size $\left(q = \left\lfloor \frac{n}{10} \right\rfloor\right)$, (in the preliminary stages of developing the algorithm, several different functions of problem size were tested and this was found to be the best)
    2) the "magic" number, q = 7, which is cited as appropriate for many applications in the literature
    3) dynamically varying q by a process where q is initialized at initial_tabu_size and is incremented by tabu._size_inc after every n iterations (based on the trial runs, initial_tabu_size was set at 5 and tabu_size_inc was set at 3).

Algorithm

- **Tabu List**
  - Long term memory

$\mu : Penalty\ pramiter\ 10$

1) Penalized frequency method

$$p_i = w_i + \mu(freq_i)$$

2) Constrained frequency method

$$\mathrm{F} = \{i: freq_i/number\_of\_iterations < critical\_freq\}$$

Algorithm

- **Implementations of tabu search heuristics**

1) TS1 : tabu search with long term memory implemented using the penalized frequency method and the size of tabu list $q = \left\lfloor \frac{n}{10} \right\rfloor$
2) TS2 : tabu search with long term memory implemented using the constrained frequency method and the size of tabu list $q = \left\lfloor \frac{n}{10} \right\rfloor$
3) TS3 : tabu search with long term memory implemented using the penalized frequency method and the size of tabu list $q = 7$

4) TS4 : tabu search with long term memory implemented using the penalized frequency method and the size of tabu list is encremented dynamically

Example and results

- **Experimentation environment**
  - Tool : IBM 3092, C
  - Part type size : 50, 75, 100, 125
  - Machine type size : 10, 15, 25
  - Weight coefficient : U[10~100]
  - g_k(number of tool slots for tool type) : 1, with a probability of 0.8
    2, with a probability of 0.15
    3, with a probability of 0.05

Example and results

- **Result**

Comparison of solution quality from tabu search and simulated annealing algorithms.

Problem size*			*UBPG*						Average ($UBPG_{SA}$ – $UBPG_{TS1}$)	No of cases (out of 5) $UBPG_{TS1} < UBPG_{SA}$
				SA			TS1			
n	*m*	*K*	Min	Avg	Max	Min	Avg	Max		
50	10	20	0.00	1.55	3.06	1.19	2.67	5.35	−1.12	2
		30	0.81	3.99	6.72	0.35	1.05	2.84	2.93	5
	15	20	0.44	1.47	2.86	0.11	1.86	4.04	−0.39	2
		30	1.75	4.55	9.61	0.00	0.61	1.20	3.95	5
	25	20	0.00	3.62	9.67	0.00	2.29	7.49	1.33	3
		30	2.95	14.85	40.89	0.76	1.67	3.08	13.19	5
75	10	20	2.94	4.39	8.01	0.67	2.11	4.11	2.28	5
		30	1.05	1.91	3.82	0.28	0.83	1.91	1.09	5
	15	20	0.83	2.58	3.51	0.63	1.09	1.75	1.51	4
		30	0.95	2.50	3.85	0.43	1.21	1.68	1.29	4
	25	20	1.10	2.06	3.23	1.25	2.34	4.63	−0.28	1
		30	1.87	7.83	25.68	0.76	2.75	6.36	5.08	4
100	10	20	0.80	3.50	8.62	0.53	2.27	3.99	1.22	4
		30	0.91	2.27	4.62	0.24	1.00	1.71	1.26	5
	15	20	0.68	2.43	5.88	1.32	1.70	1.95	0.74	2
		30	0.36	2.09	3.73	0.29	0.85	1.58	1.24	5
	25	20	0.11	3.02	5.47	0.43	1.86	2.92	1.15	4
		30	1.10	2.69	4.05	0.78	1.28	1.86	1.41	4
125	10	20	0.66	1.94	5.01	0.46	1.29	2.26	0.67	4
		30	0.40	1.79	2.83	0.17	1.43	3.19	0.37	3
	15	20	0.70	1.61	2.28	0.62	1.44	2.20	0.17	3
		30	1.05	1.98	2.82	0.70	1.28	2.09	0.70	5
	25	20	1.17	9.35	35.82	1.10	1.52	2.24	7.83	5
		30	0.62	3.22	4.98	0.70	1.40	2.37	1.83	4

*n: number of part types, m: number of machine types, K: number of tool types.

$$UBPG = (UB - v(P))/UB * 100$$

	Avg(Max)	Avg(Total)	standard deviation
SA	14.35	3.633	5.56
TS1	2.75	1.573	1.207

Example and results

- **Result**

Comparison of various versions of tabu search algorithms.

Problem size[a]			Average[b] improvement over TS1			Average CPU time (sec)			
n	m	K	TS2	TS3	TS4	TS1	TS2	TS3	TS4
50	10	20	−1.14	−1.10	−1.84	2.974	2.746	2.690	2.188
		30	−0.38	−0.48	−2.18	4.340	3.522	3.906	2.836
	15	20	−0.98	−0.58	−0.36	2.998	3.188	3.304	2.986
		30	−0.62	−0.42	−1.04	4.830	3.730	4.826	3.172
	25	20	0.08	0.66	−0.68	7.206	7.770	5.718	5.040
		30	−0.14	0.16	−0.20	7.728	10.176	7.058	6.652
75	10	20	−0.58	0.00	−0.80	4.936	6.490	4.936	4.264
		30	−0.80	0.00	−0.10	6.056	4.598	6.056	5.312
	15	20	−0.38	0.00	−0.38	4.674	4.152	4.674	4.160
		30	−0.40	0.00	−0.56	9.804	7.196	9.804	8.802
	25	20	−0.86	0.00	−0.02	11.088	8.690	11.088	9.670
		30	−1.46	0.00	−0.70	20.782	14.370	20.782	18.908
100	10	20	−0.16	0.20	−0.02	6.672	10.214	8.552	8.254
		30	−0.24	−0.30	−0.66	11.898	16.114	12.254	9.998
	15	20	−0.40	0.14	0.08	13.116	11.608	11.492	11.378
		30	−0.18	−0.10	−0.24	13.960	13.638	15.588	13.998
	25	20	0.56	0.62	−0.08	17.908	27.090	18.624	18.674
		30	−0.08	0.08	−0.14	25.376	31.678	26.246	25.212
125	10	20	−0.02	0.14	−0.28	12.968	14.878	13.218	12.732
		30	−0.22	−0.12	−0.32	17.406	18.218	19.652	16.012
	15	20	0.20	−0.22	−0.28	14.618	18.312	17.956	16.254
		30	−0.18	−0.12	−0.44	23.974	29.186	23.842	23.290
	25	20	−0.32	−0.06	−0.36	33.564	39.624	30.664	31.612
		30	−0.02	−0.32	−0.74	42.502	57.146	45.152	42.392

[a] n: number of part types, m: number of machine types, K: number of tool types.
[b] $UBPG_{TS1} - UBPG_{(\cdot)}$, where $UBPG_{(\cdot)}$ is the upper bound percentage gap from algorithm (•).

Conclusion

- **Conclusion**
  - Part type selection problem
  - Tabu Search Algorithm
    1) Penalized frequency method
    2) Constrained frequency method

- **Adv & Disadv**