Evolutionary Migration Algorithms for Scheduling

Ján Vaščák

Centre for Intelligent Technologies, Department of Cybernetics and Artificial Intelligence, Technical University in Košice, Letná 9, 041 20 Košice, Slovakia
E-mail: Jan.Vascak@tuke.sk

Abstract: This paper deals with using a special kind of evolutionary computing algorithms, the so-called migration algorithms. Namely, SOMA – Self-Organising Migration Algorithm was used for scheduling tasks of construction machines. The main goal was to find the best maintenance schedule under conditions of the best machine utilization, as well as the lowest costs. The paper firstly describes the structure and processing SOMA. Further, it will be shown this kind of scheduling can be transformed into a task of a travelling salesman. Subsequently, the proposed structure and implementation will be shown. Finally, some experiments and conclusions will be made.

Keywords: migration algorithms, SOMA, scheduling, travelling salesman problem.

1 Introduction

Scheduling is a process leading to construction of such a sequence of activities, which fulfils a set of imperative conditions, border limits, etc. with aim to optimise the fulfilment of cost conditions (financial, minimization of technical breaks or operational time, etc.).

One of the most powerful means for such tasks is evolutionary computing (EC), especially, genetic algorithms (GA), which are the most known and spread means of EC. In the literature there are lots of GA variations, e.g. [1]. They are derived from a well-known Darwinian theory of evolution. GA rely on a hypothesis that newer generations of a species due to suitable mutations, crossover and subsequent selection will be usually of a better quality than their ancestors.

However, in spite of many successful technical applications GA show some lacks. First of all, Darwinian theory is hypothetic and probably it needs lots of corrections from the biological point of view. Further, generating new populations does not guarantee automatically also convergence of the solution. New

populations may degrade from any reason. Finally, only a part of knowledge will be advanced from parents to their descendants by inheritance process.

Therefore a new metaphor in EC has been found, the so-called migration algorithms (MA). It is derived from social behaviour of some mammals like for instance wolfs. They are organized in groups, which are managed by a leader. The basic difference between GA and MA lies in modification of parameters. Individuals in GA are processed by mutation and crossover operations and a new generation arises. On the other hand, no new generations are produced by MA. Individuals are the same during the whole optimisation process. Only their position in the search area is changing, which is equivalent to producing new generations. In other words individuals (e.g. wolfs) are searching or migrating to find the best source of food.

2 Structure and Principle of SOMA

Self-Organizing Migration Algorithm proposed in [4] is based on cooperative searching (migrating) the area of all possible solutions (search area). Individuals are mutually influenced during the search process, which leads to forming / cancelling groups of individuals. Such groups organize themselves the movement of individuals, therefore the adjective self-organizing.

The SOMA parameters can be roughly divided in 2 groups:

1 Managing parameters – they influence the quality of search.

2 Finishing parameters – they determine the stopping moment of the algorithm.

Further, we describe these parameters:

1 **Path** – it determines the distance of an individual to a leader after a migration step. If Path=1 the individual will stop directly on the leader's position. If Path=2 the individual will stop in the middle between the leader and its starting position before doing the migration step. It is recommended the Path value to set up > 1 to cover the search area by individuals on a larger surface to prevent skidding into a local minimum.

2 **Step** – it determines the size of a migration step or mapping. The smaller the step the greater the chance to find a significant minimum but also higher the computational complexity and vice versa.

3 **PRT** – perturbation, a parameter, which modifies the movement vector $\vec{m}$ of an individual to the leader.

4 **D** – number of optimised variables or arguments of the fitness function. This parameter is directly depended on the solved problem and defined fitness function.

5 **NP** – number of individuals (population size). This value depends usually on D and it directly influences the search quality. The greater NP the higher possibility to find a significant (maybe global) minimum.

6 **Migration** – it is analogous to the number of populations in GA.

7 **Accepted error** – it defines the maximum allowed difference between the best and the worst individual in the population. To find really a significant minimum and to prevent divergence from the optimal solution it is necessary to achieve good solution also for other individuals not only for the best one. It means if the real error is smaller than the accepted error then the algorithm will be stopped.

The parameters from Path to NP belong to the first group and the last two parameters are finishing ones.

2.1 SOMA Operations

One important advantage of this algorithm is based on its ability to process diverse data types of parameters like integers, real or discrete values. They can be mixed mutually, too.

These parameters define the structure and universes of discourse of individuals. To generate an initial population the so-called *specimen* is defined firstly:

$$Specimen = (spc_1^{TYPE}(spc_1^{LL}, spc_1^{UL}), \ldots, spc_D^{TYPE}(spc_D^{LL}, spc_D^{UL})), \tag{1}$$

where *Type* denotes the data type of a parameter, *LL* and *UL* are the low and upper limits of the universe of discourse, respectively. These intervals of values represent permitted parameter values or from another point of view physical limitations of the application. The population (real individuals) will be generated by:

$$P^0 = \{x_{1,1}^0, \ldots, x_{i,j}^0, \ldots, x_{NP,D}^0\} = \{rnd(x_{i,j}^{UL} - x_{i,j}^{LL}) + x_{i,j}^{LL}\}, \tag{2}$$

where P^0 is the initial population and $x_{i,j}$ represents *j*-th dimension of the *i*-th individual ($i=1, \ldots, NP$ and $j=1, \ldots, D$).

Besides SOMA uses also operators of perturbation and migration.

The perturbation is analogous to the mutation process in GA. However, the result of such an operation is not a property change of an individual but its movement vector $\vec{m}$ to the leader is perturbed (interfered), i.e. it is not direct to the leader (as seen in fig. 1).

The movement vector $\vec{m}$ represents the distance between starting point of a given individual and the leader, i.e. in the vector description:

$$\vec{m} = \vec{r}_L - \vec{r}_0, \tag{3}$$

where $\vec{r}_L$ and $\vec{r}_0$ are vectors of the leader and the starting point of a given individual, respectively.

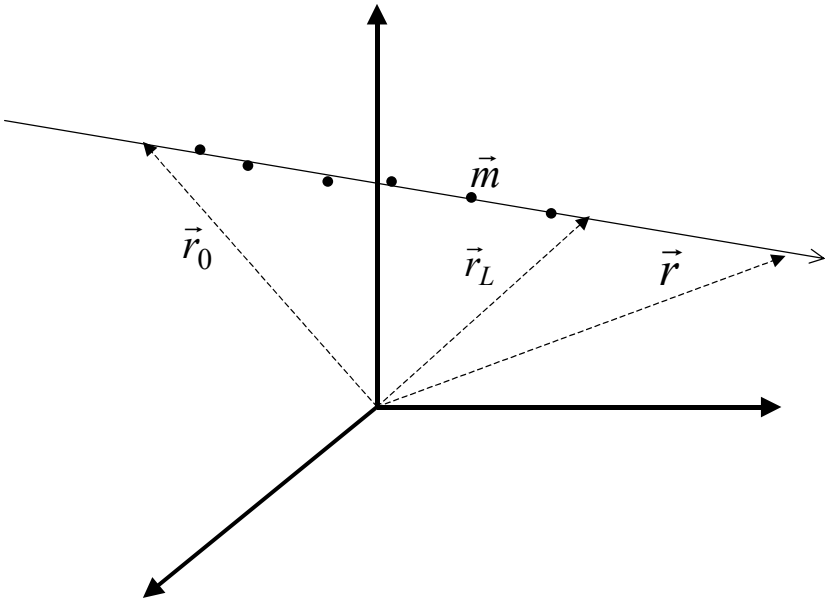

Figure 1
Relations between vectors $\vec{r}_0$, $\vec{r}_L$ and $\vec{r}$ in a 3-dimensional space (D=3).

Hence, the perturbation has following influence on the real position of such an individual in next step $\vec{r}$:

$$\vec{r} = \vec{r}_0 + \vec{m}t\vec{v}_{PRT}, \tag{4}$$

where *t* is the order of steps on a path from a given individual at starting point $\vec{r}_0$ to the leader $\vec{r}_L$, i.e. *t=k.Step*, *k=0, 1, ...m*, where *Path=m.Step*. (individual steps in the fig. 1 depicted as bullets •). The elements of the perturbation vector $\vec{v}_{PRT}$ are then created in each migration cycle by a condition: *if* $rnd_r < PRT$ *then* $\vec{v}_{PRT\,j} = 1$, *else* $\vec{v}_{PRT\,j} = 0$, where rnd_j is a randomly generated number and *j* is the index for a given property (*j=1, …, D*). In other words, if *PRT* has a small value then $\vec{v}_{PRT}$ will have mostly zeros and the perturbation will affect direct movement of a given individual to the leader, i.e. the movement vector $\vec{m}$ will be modified. Only

the dimensions where values of $\vec{v}_{PRT\,j}$ are set to *1* will not be perturbed and the movement will be similar to original form of the vector $\vec{m}$ (see fig. 2).

Similarly, also migration is analogous to crossover in GA. During one migration cycle (4) is processed in steps, which corresponds to mapping the state space. Although there does not exist any generating new populations but this representation is equivalent to a sequence of descendants (one step – one descendant or one element of given population). Also the best solution will be chosen and after the migration cycle the individual will come back to the best position, which corresponds to the selection in GA. Generating new populations is substituted by migrating individuals in state space. There exists one significant difference in comparison to GA, where mutation and crossover are timely divided operations but as seen in (4) in SOMA perturbation and migration are processed simultaneously.

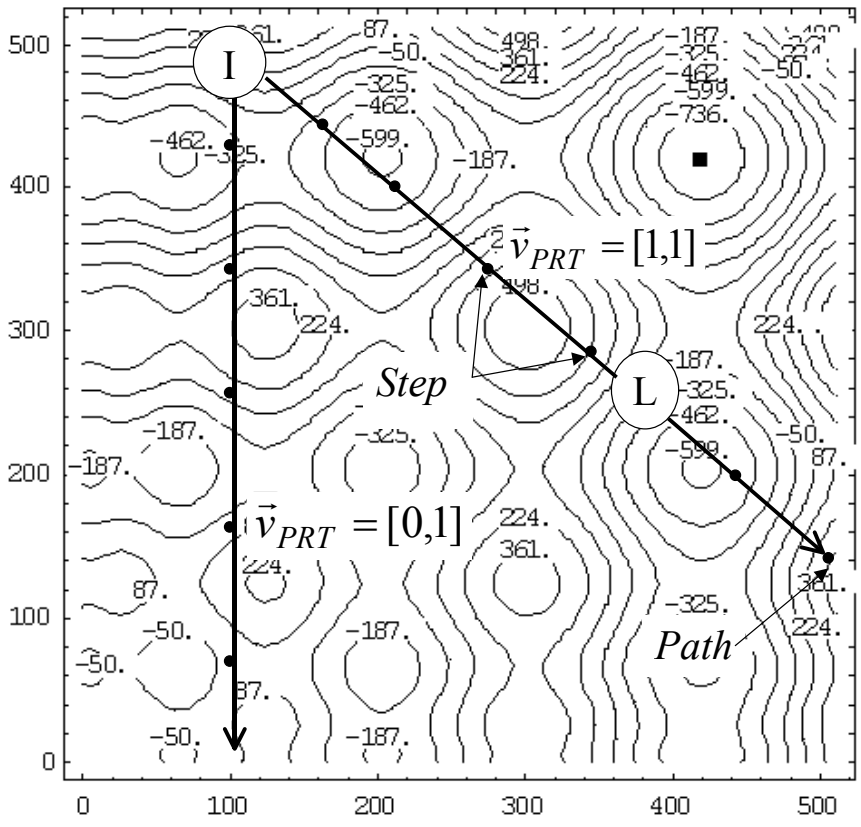

Figure 2
Relation between perturbation vector $\vec{v}_{PRT}$ and movement vector $\vec{m}$ in a 2-dimensional space; I – individual, L - leader.

2.2 Processing SOMA

Processing in SOMA depends on the strategy used. There are several possible strategies but the strategy All-To-One seems to be the primary one and the following process will be explained using this kind of strategy.

First, after some initialising steps like defining managing and finishing parameters as well as creating the population from the specimen each individual will be evaluated by a fitness function. The best individual will be chosen for a leader in next migration cycle. After that individuals start to move to the leader in jumps calculated in (4) and mutually distanced by the parameter *Step*. After each jump (step) the individuals will be evaluated by the fitness function. If the evaluation is better than the previous one it will be remembered. After individuals will reach the last jump determined by the parameter *Path* they will return to new positions where they found the best fitness, i.e. beside the leader another individuals will be moved to another position (fig. 3). In other words they did searching in the state space and migrated. In such a manner one migration cycle has been finished and new one will immediately start after a new leader (the best individual) will be determined.

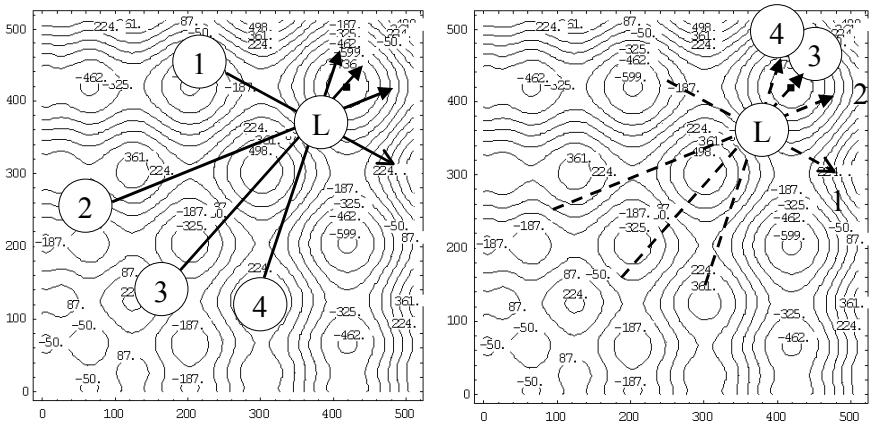

Figure 3
Migration process of individuals 1 – 4 to the leader L during one migration cycle.

As seen from above the size of population remains the same, even the individuals are the same. There will not be generated any new population and no selection in sense of GA will be done. The acquired knowledge remains in each individual and we can observe certain learning process during migration cycles. The only kind of selection can be observed in choosing the leader, which depends on the quality of an individual (fitness value).

Fig. 3 can be drawn for each migration cycle, subsequently a sequence of such figures can be done and we can observe very interesting arising as well as

cancelling structures of individuals. The strategy of information interchange, i.e. cooperation and competition in searching a leader can be withdrawn. In spite of GA where no information interchange among individuals arises it represents a new synergic quality. Therefore SOMA should be rather classified into the group of the so-called memetic algorithms because it is based on information evolution in other words. There is a kind of learning and the optimisation is based not only on blind stochastic generating new possible solutions. This effect can explain a quicker as well as more reliable optimisation process for searching global extreme than GA.

Concerning terminating the optimisation two finishing parameters are very important – migration and accepted error. The migration parameter is in other words number of migration cycles, comparing to GA number of populations. This parameter prevents never-ending optimisation process and it will play the role of a hard stop if the accepted error will not be reached. The sense of accepted error is based on an assumption the found result will be more reliable if also other individuals reach good fitness values. The principle of a certain preferred element is not supported by SOMA.

3 Application of SOMA in Scheduling

SOMA was used for purposes of scheduling in civil engineering [2, 3]. Each building company has a set of diverse construction machines, which are often very expensive equipments and therefore their use must be maximally effective under minimal risk of failures. In construction scheduling there are lots of contradictory efforts and design of a work schedule requires much skill and experience. Often fitness differences among work schedules are very extreme, which can be transformed directly into finances [2].

The main approach is based in constructing individuals like schedules, which will be evaluated by a fitness function (almost always leading to financial costs). To reach a convenient description of such a problem following relations must be quantitatively described:

1. Properties of used construction machines.
2. Distances among work places (each with each).
3. Job reserve – ordered jobs, which are necessary still to be done.

In general, the main implementation problem is in knowledge representation of all characteristics and limitations. In the case of construction machines there are following characteristics: number of work types (e.g. piling-up, picking), work place, measure of reliability (depends on mass of performed activity), power (how much work is a machine able to perform during one day) and cost (price of work).

Job reserve contents usually the data about: starting date, duration and quantity of a given work type for a given work place.

Summarizing the above, we can construct an individual (schedule) as a set of parameters depicted in fig. 4.

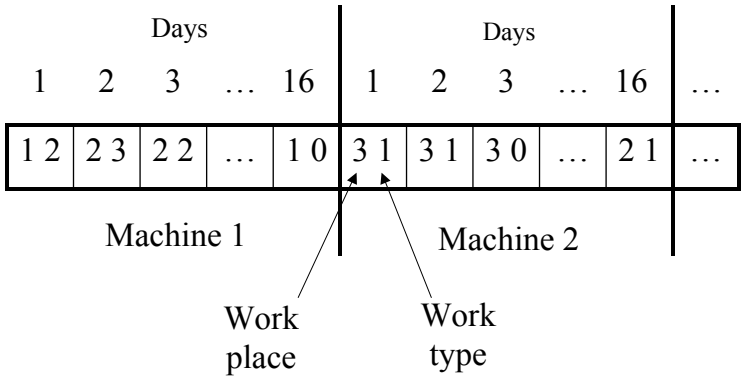

Figure 4
Data structure of an individual in form of a proposed schedule.

For each machine there is an extra schedule divided into days (in this case a 16-days schedule). Each day contents the information about the work place where the machine is doing its job as well as the work type. The length of such a parameter vector, which must be optimised, is therefore given as product of number of machines, number of days and 2 (work place and work type).

Further important step is defining the fitness function. Since, in this case the problem has a multi-criteria character the real fitness function was decomposed into several parts, which were merged by a weighted sum. Basically, the task was to find minimum cost value, where mainly working, transportation and maintenance costs were taken into consideration. It is always a problem-oriented task and it varies from application to application. For instance, a risk analysis plays a very important role because if a heavy machine will be defected on a place with difficult reachability then maintenance costs will be much greater than in the case of timely maintenance in a workshop.

4 Experiments and Their Evaluation

Several experiments were done with fictive companies, which but reflect usual situations [2]. The number of machines, work places and job reserve as well as length of schedules were changed. Further, the most complex experiment will be described in this section.

The proposed company has 12 machines whose parameters are described in tab. 1.

Machine	**1**	**2**	**3**	**4**	**5**	**6**
Number of work types	2	1	1	2	1	1
Work place	A	E	B	D	E	C
Reliability	750	380	370	350	185	670
Work type	1	2	11	7	8	3
Power	80	90	120	85	90	110
Cost	750	800	1000	600	1300	550
Work type	3			10		
Power	100			70		
Cost	650			500		
Machine	**7**	**8**	**9**	**10**	**11**	**12**
Number of work types	1	2	2	1	1	1
Work place	A	C	B	E	D	D
Reliability	395	370	410	610	245	290
Work type	10	9	1	4	5	6
Power	95	120	100	100	80	70
Cost	700	950	700	870	750	600
Work type		7	3			
Power		80	100			
Cost		600	700			

Table 1
Parameters of construction machines.

Similarly, the table of distances among work places can be in the form of tab. 2. There the places A – B are considered. Also the place F is depicted but it is a maintenance workshop as a special work place.

A	**B**	**C**	**D**	**E**	**F**	
0	40	36	90	120	55	**A**
	0	22	85	80	45	**B**
		0	70	55	22	**C**
			0	15	40	**D**
				0	70	**E**
					0	**F**

Table 2
Table of distances among work places A – B in km.

The experiments were done first of all with the number of individuals [3]. They varied from 30 to 150 and the optimal number was about 80. The optimal number of migration cycles was between 5000 and 8000 cycles. Besides two strategies of SOMA were compared – the already known All-To-One and All-To-One-Rand, which is different from the first one only in a random choice of leader, i.e. the leader needs not be the best one. As a surprise, just this second method had a little better results than the method All-To-One.

The fitness value represents in fact financial costs for these tasks, i.e. we tried to find the global minimum, the less the smaller costs. Therefore a more comprehensive comparison will probably be if we compare our designed schedule to a schedule proposed by an experienced expert as seen in tab. 3 and 4, respectively. The syntax of data in these tables is following: x:y (z), i.e. x – machine number, y – work type number, z – quantity of scheduled job.

The comparisons in tab. 3 and 4 show that results obtained by SOMA are very similar to results of a human expert. The system reached almost the global minimum. However, computational efforts are very high and calculating a longer schedule or a schedule for a bigger company would lead to considerably long computational times (from hours to tens of hours).

Conclusions

The obtained experiments show that SOMA is suitable for solving tasks of middle complexness. There was done also an experiment with 24 machines and for a 26-days schedule but results were not so much satisfactory and also computational time was very long.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	1 : 1 (310)				1 : 3 (200)			1:3 (100)	7 : 10 (270)			8 : 9 (350)			12:6 (50)	
A								6:3 (90)								
B	9:1 (380)				3 : 11 (350)			2 : 2 (170)	10 : 4 (400)					5 : 8 (175)		
C	6 : 3 (530)					8 : 9 (360)			5 : 8 (175)	2 : 2 (175)		3 : 11 (360)				
D		12 : 6 (275)				6:3 (100)		4 : 7 (160)		11 : 5 (240)				4:10 (70)		
D						9:3 (80)							7:10 (90)		7:10 (60)	
E	5 : 8 (175)		2 : 2 (355)				9:3 (100)	10 : 4 (200)		12 : 6 (275)				7:10 (90)		
F			5		9	12	2		3		8	5		12	8	3
F							6	5	12			7			2	5
F															11	10

Table 3
A 16-days schedule proposed by SOMA.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	1 : 1 (310)				1 : 3 (390)				7 : 10 (270)			8 : 9 (350)			12:6 (50)	
B	9:1 (380)				3 : 11 (350)			2 : 2 (170)	10 : 4 (400)					5 : 8 (175)		
C	6 : 3 (530)					8 : 9 (360)			5 : 8 (175)	2 : 2 (175)		3 : 11 (360)				
D		12 : 6 (275)				9 : 3 (180)		4 : 7 (160)		11 : 5 (240)				4 :10 (210)		
E	5 : 8 (175)		2 : 2 (355)				6:3 (100)	10 : 4 (200)		12 : 6 (275)				7:10 (90)		
F			5		9	12	2	6	1		5		11	10	7	5
F								3	8					12		3

Table 4
A 16-days schedule proposed by a human expert.

However, it is necessary to notify this task represents the travelling salesman problem, which is from its nature non-polynomial and all another methods will once meet with computational complexity problems.

References

[1] Kvasnička, V., Pospíchal, J., Tiňo, P.: Evolution algorithms (in Slovak); STU Bratislava 2000, Slovakia, ISBN 80-277-1377-5.

[2] Makši, P.: Modelling Processes of Machinery Capacities in Building Production (in Slovak); minimum work, TU Košice, Slovakia, 2002.

[3] Mašura, J.: Schedule optimization of construction machines (in Slovak); diploma thesis, TU Košice, Slovakia, 2004.

[4] Zelinka, I.: Artificial Intelligence in Problems of Global Optimisation (in Czech); BEN, Prague, Czech Republic, 2002, ISBN 80-7300-069-5.