A Cooperative Opposite-Inspired Learning Strategy for Ant-based Algorithms

Nicolás Rojas-Morales[1], María-Cristina Riff[1], Carlos A. Coello Coello[2], and Elizabeth Montero[1]

[1] Universidad Técnica Federico Santa María, Valparaíso, Chile,
[2] CINVESTAV-IPN (Evolutionary Computation Group), México
nicolasrojas@acm.org

Abstract. In recent years, there has been an increasing interest in Opposite Learning strategies. In this work, we propose COISA, a Cooperative Opposite-Inspired Strategy for Ants. Inspired on the concept of anti-pheromone, in this approach, sub-colonies of ants perform different search processes to construct an initial pheromone matrix. We aim to produce a repel effect to (temporarily) avoid components that were related to an undesirable characteristic. To assess the effectiveness of COISA, we selected Ant Knapsack, a well-known ant-based algorithm that efficiently solves the Multidimensional Knapsack Problem. Results in benchmark instances show that the performance of Ant Knapsack is improved considering the opposite information, so that it can reach better solutions than before.

1 Introduction

We propose here an Opposite-Inspired Learning strategy where the search process of an ant-based algorithm is divided into two steps: a *First Step* used to identify a $u\mathcal{D}$-characteristic from complete instantiations and a *Second Step* used to solve the problem of interest. Three sub-colonies of ants cooperate obtaining information during the *First Step*. Such information will be considered in the *Second Step* to change its decisions during the construction process. Each sub-colony performs a search process defined by a *Method*. Here, we propose a collaboration between these *Methods* that were previously proposed in [10, 11]. Sections 2 and 3 present details of our proposed strategy.

Opposite Learning (OL) is a search strategy that has been applied for mapping candidate solutions with the objective of increasing the coverage of the solution space [5]. Opposition-Inspired Learning (OIL) [12] was proposed considering that, in some cases, the idea of mapping solutions is not intuitive because of some algorithm-specific properties. Some previous OIL ant-based approaches have been proposed [2, 4, 6, 7]. In our case, the term *opposite* is related to the possible decisions made by ant-based algorithms, that could lead the search process towards poor quality candidate solutions.

To evaluate our strategy, we selected the well-known Ant Knapsack (AK) algorithm [1] originally proposed for solving the Multidimensional Knaspsack

Problem (MKP). The implementation in AK is described in Section 4. It is important to mention, however, that our objective is not to propose the best algorithm for the MKP. The idea is to evaluate the use of a learning strategy to focus the search process of a specific ant-based algorithm.

2 An OIL Strategy for Ant-based Algorithms

Let's assume a combinatorial problem $\mathbb{P}$ and an ant-based algorithm A. We are interested in improving the search process of A, in terms of the quality of the solutions that A can build. For this, we are interested in providing useful information to A in order to improve its *intermediate* decisions. Let's assume that each ant k of A incrementally constructs a complete instantiation of a solution I_C^k, making stochastic *intermediate* decisions to include components into a partial instantiation I_P^k.

In most cases, components are included in I_C^k because a certain preference related to their heuristic knowledge (η) and pheromone information (τ) was considered in the *intermediate* decisions of A. As η is particularly defined in A and the information in the pheromone matrix is limited by the vertices which were already visited during the current execution, in some cases, the information provided to perform *intermediate* decisions might be poor. Considering that $\mathbb{P}$ is complex to solve, this information can affect some *intermediate* decisions and lead the construction process to solutions with less quality than expected.

Let's assume that I_C^k has some characteristic w that can be measurable and related to: a structural property of I_C^k, a quality feature of I_C^k, a feature related to the (in)feasibility in I_C^k, and a problem-specific property feature not detectable by $\mathbb{A}$, among others. During the construction process of I_C^k, *intermediate* decisions are biased giving priority to some components that look more promising than others. We name this characteristic as *undesirable* ($u\mathcal{D}$) because these intermediate decisions prefer components that are locally interesting, but finally produced that $F(I_C^k) < F(I_C^*)$.[3] It is important to remark that this characteristic is not inherent to the problem $\mathbb{P}$, but it cannot be perceptible by the current pheromone information and by the heuristic knowledge, as it is specifically defined in A. We propose to learn about this $u\mathcal{D}-$characteristic w in I_C^k to decrease the attraction to components that A considers promising. The objective is to allow A to consider other *intermediate* decisions during its construction process and, finally, obtain better quality solutions.

Let $S_{(A,i)}$ be a set of complete instantiations obtained by A during its i^{th} iteration and w a $u\mathcal{D}$-characteristic. As w is measurable, solutions in $S_{(A,i)}$ can be compared considering the presence of a $u\mathcal{D}$-characteristic. We define $S_{(A,i)}^w$ as the set of complete instantiations that have more presence of w. As the pheromone produces a modification of the way in which the problem is represented and perceived by artificial ants [3], we decided to use the pheromone to learn about the $u\mathcal{D}$-characteristics. Our hypothesis is the following: if we consume a certain

[3] Considering that $\mathbb{P}$ is a maximization problem with an objective function F and I_C^* is an optimal solution.

amount of resources in identifying and learning about some $u\mathcal{D}$-characteristic w in $S_{(A,i)}$, the search process could be further focused making decisions using this knowledge so that we can obtain complete instantiations of a better quality. For this, we propose to divide the search process of A into two steps.

2.1 Division of the Process

First, we propose to divide the search process of A into two steps: a *First Step* (*FS*) to learn about w in $S_{(A,i)}$ and a *Second Step* (*SS*) performed by A using the knowledge obtained in the *FS*. Inspired in the concept of *anti-pheromone* [13], the idea is to produce a repellent effect to some pairs of components of solutions in $S^w_{(A,i)}$, allowing A to consider other components that, originally, would not be included. From now on, the pheromone used during the *FS* will be called *anti-pheromone*. As A was designed, normal pheromone is used during the *SS*.

Let A° be an ant-based algorithm that will perform the *FS* and let's use *anti-pheromone* to decrease the attraction of paths that are related with complete instantiations in $S^w_{(A,i)}$. The definition of the representation for $\mathbb{P}$ and the *state transition rule* of A° is the same as in A. The *FS* is performed by A° consuming an amount $B * maxRes$ of resources[4], where $B \in [0, 1]$ is a parameter that defines the budget of resources designed for the learning step. At the end of the *FS*, an initial pheromone matrix will be obtained and used by A. Finally, A performs its search process considering the remaining $(1 - B) * maxRes$ resources.

2.2 Methods

In order to explore and compare different possibilities to identify a $u\mathcal{D}$-characteristic, we propose three different *Methods*. Each method will consider a different definition for the heuristic knowledge and anti-pheromone management for A°. The methods are named *Soft Opposite-Learning* (SOL), *Worst Opposite-Learning* (WOL) and *Half Opposite-Learning* (HOL).

SOL: This method is focused on identifying a $u\mathcal{D}$-characteristic related to the quality of complete instantiations but trying to perform a similar search process as in A. For this, η of A° will be the same as in A. On the other hand, anti-pheromone will be decreased in edges that are related to the lowest quality solution of each iteration. The information obtained during a *FS* performed by the SOL method will reduce the level of attraction produced by the heuristic knowledge in the corresponding *intermediate* decisions of A.

WOL: This method is focused on evaluating the effect of taking totally opposed decisions to the objective of the problem $\mathbb{P}$. For this, the heuristic knowledge should be inverted in each intermediate decision (Eq. 1), where $J_k(i)$ is a list of candidate components, η^A_{ij} is the heuristic knowledge of A, and the maximum and minimum heuristic knowledge of values of the current decision are also

[4] These resources can be execution time, a fixed number of evaluations, and conflict checks, among others. In general, the amount of resources can be defined considering how A was originally evaluated.

considered. Here, the construction process is biased towards actual poor quality solutions by the translated heuristic information and the anti-pheromone. Furthermore, components of the lowest quality solution obtained will be marked with *anti-pheromone* at each iteration.

$$\eta_{ij}^{A^\circ} = \max_{u \in J_k(i)} (\eta_{iu}^{A}) + \min_{u \in J_k(i)} (\eta_{iu}^{A}) - \eta_{ij}^{A} \tag{1}$$

HOL: This method is focused on detecting a problem-specific $u\mathcal{D}$-characteristic w. In order to detect this problem-specific feature, the heuristic knowledge η^{A° should be redefined in A°. The construction process should be guided considering new information from η^{A°, allowing the search process to consider information in the presence of w in the complete instantiations obtained. Anti-pheromone will be used to reinforce the objective of η^{A°.

3 Cooperation Between Sub-colonies

In our proposed approach, three sub-colonies of ants cooperate in the construction of a pheromone matrix. Each sub-colony focuses in obtaining information about a $u\mathcal{D}$-characteristic and is guided by one *Method*. During the *FS*, all ants consider the same *anti-pheromone* matrix M to construct solutions. At the end of each iteration, *anti-pheromone* will be updated by ant^{SOL}, ant^{HOL} and ant^{WOL} considering the following rule:

$$anti\tau_{ij}^{new} = anti\tau_{ij}^{old} - \Delta_{ij}^{SOL} - \Delta_{ij}^{HOL} - \Delta_{ij}^{WOL} \tag{2}$$

where Δ_{ij}^{SOL}, Δ_{ij}^{HOL} and Δ_{ij}^{WOL} are the decreased amounts of antipheromone. As the collaboration of these three sub-colonies can be time consuming, we decided to execute the *FS* in parallel and the *SS* is executed sequentially. *COISA* was implemented in *POSIX Threads*. Two types of threads will be considered: *constructor* or *manager* threads. Considering a total of N threads and m ants, one *manager* thread will be focused on the pheromone management and $(N-1)$ *constructor* threads are focused in constructing and evaluating solutions. The *manager* thread waits until all constructor threads finish, using a barrier, to construct their tasks to update the pheromone matrix M. For the synchronization of all the threads, a barrier, a conditional variable and a mutex are used.

4 Case Study: Multidimensional Knapsack Problem

Multidimensional Knapsack Problem (MKP) is an *NP-hard* combinatorial optimization problem. It considers a set of objects and a knapsack with T dimensions, each one with a maximium capacity defined (b_t). Each object has a defined profit p_i and weight w_{it} in each problem dimension t. The idea is to select a subset of objects maximizing the total profit, satisfying each capacity constraint.

Here, we introduce *COISA* into Ant Knapsack [1] (AK), a well-known ACO algorithm designed for solving the MKP. AK is a $\mathcal{MAX}-\mathcal{MIN}$ Ant System [14] that constructs feasible complete instantiations. Pheromone represents the desirability of including pairs of objects simultaneously. The heuristic knowledge is defined as: $\eta_{I_P^k}(o_j) = \frac{p_j}{\sum_{t=1}^{T} \frac{w_{jt}}{CC_t}}$, where CC_t is the Current Capacity in dimension t (defined as $CC_t = b_t - \sum_{o_v \in I_P^k} w_{vt}$). Pheromone is deposited in each pair of objects of the best quality solution found of each iteration (L_{b_i}). Here, an amount of $\Delta\tau = \frac{1}{1+|F(L_{b\_f})-F(L_{b\_i})|}$ is deposited, considering that L_{b_f} is the best solution found during the execution.

4.1 Details of the Implementation

This section presents some details that should be considered before the implementation of *COISA* in AK. First, the amount of *anti-pheromone* $\Delta anti\tau$ is defined similarly as in AK. In this case, the worst solution found in the current iteration (L_{w_i}) and the worst solution found during the execution (L_{w_f}) are considered. Moreover, as in AK, one ant per sub-colony will be allowed to deposit anti-pheromone during the *FS*. In order to obtain information without any perturbation, the evaporation is not considered during the *FS*.

SOL and WOL methods are implemented as was already explained in Section 2.2. For the HOL method, it is necessary to define a heuristic knowledge for guide its search process. In this case, we considered the same η used in [11]: $\eta_{I_P^k}(o_j) = \frac{p_j}{\sum_{t=1}^{T} RC_t}$, where RC_t is the *remaining capacity* in the dimension t defined as $RC_t = b_t - w_{jt}$. In this case, the $u\mathcal{D}-$characteristic points to identify the *core* of objects for which it is hard to decide if they will be part of an optimal solution or not [8]. Moreover, *anti-pheromone* will mark the lower quality solution of each iteration.

5 Experiments and Results

We considered two sets of 30 instances from the OR Library proposed by Chu and Beasley: 10×100 (10 dimensions and 100 objects) and 5×100 (5 dimensions and 100 objects). In order to compare the collaboration between the three sub-colonies, we present results by each method independently: SOL-AK, HOL-AK and WOL-AK. For all the executions we considered a number of ten threads. The hardware platform used was a Power Edge R630 server with 2 Intel(R) Xeon(R) CPUE5-2680v3 @ 2.50GHz, 128 GB of RAM using Ubuntu x64 16.10 distribution. We considered the same parameter values proposed in [1]: $\alpha = 1$, $\beta = 5$, $\rho = 0.01$, $N_{Total} = 30$, $\tau_{max} = 6$ and $\tau_{min} = 0.01$. To determine the parameter values for our approaches we used Evolutionary Calibrator (EVOCA) [9], a parameter tuner algorithm, considering randomly selected instances from both sets. The objective was to obtain the number of ants for each sub-colony and the budget B. The obtained parameter values after 3500 evaluations of EVOCA are: (1) for COISA are $N_{SOL} = 16$, $N_{HOL} = 11$, $N_{WOL} = 8$ and $B = 0.211$, (2) for SOL are $N_{SOL} = 20$ and $B = 0.241$, (3) for HOL are $N_{HOL} = 2$ and $B = 0.422$, (4) for WOL are $N_{HOL} = 16$ and $B = 0.408$. Table 1 shows the results obtained for the 10×100 set and Table 2 shows the results for the 5×100 set. We considered 50 independent runs per instance, each with 60000 evaluations (*maxRes*).

Table 1. Results for Set 10×100 from OR Library

#	BK	AK			COISA-AK			SOL-AK			HOL-AK			WOL-AK		
		AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST
1	23064	**23016.0**	42.2	23064	23014.3	46.4	23064	22998.6	47.5	23057	23008.0	41.0	23064	23006.7	42.7	23064
2	22801	**22714.0**	67.2	22801	22702.2	83.8	22801	22713.8	66.5	22801	22694.6	58.3	22801	22693.6	69.4	22801
3	22131	22034.0	66.9	22131	**22046.6**	56.0	22131	22024.4	69.7	22131	22008.2	69.9	22131	22035.5	65.3	22131
4	22772	**22634.0**	60.6	22717	22613.3	63.9	22763	22623.4	64.0	22772	22598.2	73.7	22772	22601.7	53.8	22709
5	22751	22547.0	66.3	22654	**22559.2**	47.6	22654	22543.2	70.8	22697	22533.0	66.9	22697	22542.7	51.4	22697
6	22777	22602.0	63.3	22716	22593.4	46.8	22716	**22610.3**	51.4	22716	22594.7	46.0	22664	22591.9	40.5	22675
7	21875	21777.0	44.9	21875	**21790.8**	36.7	21875	21773.4	45.5	21875	21780.1	48.6	21875	21774.3	54.2	21875
8	22635	22453.0	89.2	22551	22498.8	54.1	22635	**22512.0**	40.6	22551	22511.7	57.8	22635	22500.1	57.5	22635
9	22511	22351.0	69.4	22511	**22379.6**	47.0	22511	22369.7	40.3	22438	22362.4	51.6	22511	22352.2	62.5	22511
10	22702	22591.0	88.5	22702	**22616.0**	102.9	22702	22600.1	99.8	22702	22576.5	91.0	22702	22572.9	88.8	22702
1	41395	41329.0	48.5	41395	41324.1	47.4	41395	**41329.1**	49.8	41395	41312.4	51.8	41393	41309.0	48.7	41395
2	42344	42214.0	49.5	42344	**42233.5**	47.0	42344	42232.2	60.4	42344	42210.2	45.5	42344	42221.4	54.9	42344
3	42401	42300.0	58.1	42401	42309.0	38.4	42401	42311.5	41.7	42401	**42316.1**	47.2	42401	42313.6	43.5	42401
4	45624	45461.0	73.6	45624	**45484.2**	69.4	45624	45450.2	70.9	45585	45462.3	71.6	45585	45474.4	64.2	45598
5	41884	41739.0	57.3	41884	**41770.0**	53.0	41884	41769.9	52.0	41884	41758.8	53.0	41884	41750.4	50.2	41884
6	42995	42909.0	76.3	42995	42910.6	76.5	42995	42898.8	72.7	42995	42891.3	78.1	42995	**42923.4**	69.8	42995
7	43574	43464.0	71.7	43553	43466.9	50.0	43553	43470.0	43.0	43553	**43479.0**	47.6	43553	43463.7	46.5	43552
8	42970	42903.0	47.7	42970	42904.7	39.4	42970	42901.5	48.1	42970	**42924.6**	35.3	42970	42915.2	40.1	42970
9	42212	42146.0	48.0	42212	**42167.3**	39.8	42212	42165.7	39.7	42212	42160.6	38.4	42212	42162.5	42.2	42212
10	41207	41067.0	89.7	41207	**41098.7**	36.9	41207	41085.9	39.0	41134	41093.5	38.3	41207	41077.7	44.8	41207
1	57375	57318.0	59.5	57375	57295.9	66.1	57375	57307.8	68.1	57375	57311.7	74.1	57375	**57321.9**	61.3	57375
2	58978	58889.0	40.2	58978	**58914.2**	32.4	58978	58899.4	54.6	58978	58886.4	43.3	58934	58898.1	24.1	58978
3	58391	58333.0	29.5	58391	**58337.7**	26.3	58391	58321.2	47.8	58391	58326.8	32.5	58391	58335.4	27.2	58391
4	61966	61885.0	42.4	61966	**61891.2**	36.4	61966	61876.0	47.9	61966	61873.9	40.6	61966	61882.7	36.6	61966
5	60803	60798.0	5.0	60803	**60800.6**	3.0	60803	60799.9	3.2	60803	60800.5	3.1	60803	60800.0	5.2	60803
6	61437	61293.0	52.7	61437	61295.3	55.6	61437	61294.1	52.6	61437	61288.3	48.6	61437	**61297.5**	52.5	61437
7	56377	56324.0	35.7	56377	56319.0	35.4	56377	56311.0	47.0	56377	56313.9	49.1	56377	**56328.3**	33.8	56377
8	59391	59339.0	53.3	59391	59340.7	42.6	59391	59331.5	51.4	59391	59331.2	53.0	59391	**59341.3**	37.5	59391
9	60205	60146.0	62.6	60205	**60167.7**	50.7	60205	60123.1	73.5	60205	60096.8	70.9	60205	60155.8	56.9	60205
10	60633	60605.0	36.1	60633	**60613.9**	32.2	60633	60589.4	47.6	60633	60571.7	48.4	60633	60613.5	30.7	60633

Light grey cells show the best average quality (AVG) of the 50 seeds and dark grey cells show the Best quality solution obtained. Also, the standard deviation (SDV) is shown for each instance and algorithm. First, results show that AK could find most of the best known solutions for the instances from both sets (51 of the 60 instances). Moreover, *COISA-AK* outperformed AK obtaining the best known solution in 53 of the 60 instances. This shows that the collaboration between sub-colonies is better than each method on their own. Regarding the average quality, results show that AK obtained better results in the 5×100 set and *COISA-AK* was better for the 10×100 set. Finally, considering the independent and the cooperative approaches, all the best known solutions can be found using opposite information. The non-parametric Wilcoxon test was applied to assess that these algorithms are statistically different ($pvalue = 0.01$). About the Speedup obtained by *COISA-AK*, the average was 1.8, with a maximum of 4.9 and a minimum of 1.4. As the *FS* only consumes 20% of the evaluations, these metrics show the positive effect of using a parallel architecture.

6 Conclusions

In this work, we proposed a Cooperative Opposite-Inspired Strategy for ants-based algorithms. The objective of this approach is to obtain information about

Table 2. Results for Set 5×100 from OR Library

#	BK	AK			COISA-AK			SOL-AK			HOL-AK			WOL-AK		
		AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST	AVG	SDV	BEST
1	24381	**24342.0**	29.3	24381	24340.6	29.0	24381	24329.4	38.2	24381	24335.1	35.3	24381	24330.4	31.4	24381
2	24274	**24247.0**	38.5	24274	24241.2	35.1	24274	24234.9	42.8	24274	24246.0	33.5	24274	24229.2	38.8	24274
3	23551	**23529.0**	8.0	23551	23527.2	9.2	23551	23526.3	13.6	23551	23525.6	14.1	23551	23527.1	11.9	23551
4	23534	**23462.0**	32.6	23534	23460.1	32.7	23527	23453.3	44.4	23534	23458.4	34.8	23527	23457.8	30.2	23511
5	23991	23946.0	31.8	23991	23942.5	26.8	23991	23934.2	33.8	23991	23940.1	35.0	23991	**23950.5**	29.0	23991
6	24613	**24587.0**	31.3	24613	24585.3	25.8	24613	24583.0	28.9	24613	24573.3	34.2	24613	24579.2	28.8	24613
7	25591	25512.0	43.8	25591	25521.8	41.3	25591	**25524.2**	47.8	25591	25506.2	39.7	25591	25509.5	45.9	25591
8	23410	23371.0	30.3	23410	23375.1	33.5	23410	23378.2	29.6	23410	23378.8	29.4	23410	**23381.5**	31.9	23410
9	24216	24172.0	32.9	24216	**24177.0**	31.1	24216	24171.7	32.7	24216	24163.1	38.0	24216	24164.5	39.3	24216
10	24411	**24356.0**	44.3	24411	24346.1	45.8	24411	24340.5	44.5	24411	24342.9	47.2	24411	24346.7	44.8	24411
1	42757	42704.0	14.3	42757	42706.5	25.4	42757	**42709.8**	21.0	42757	42700.6	11.4	42757	42701.6	14.8	42757
2	42545	42456.0	15.8	42510	42458.4	14.6	42471	**42459.5**	12.9	42494	42455.0	21.3	42545	42458.8	25.4	42545
3	41968	41934.0	22.3	41967	**41939.8**	15.9	41968	41935.2	23.0	41967	41930.9	26.3	41967	41930.8	27.7	41967
4	45090	45056.0	24.0	45071	45056.1	24.1	45071	**45058.3**	23.1	45071	45041.0	29.4	45071	45049.4	31.7	45071
5	42218	42194.0	33.2	42218	42201.9	31.4	42218	42196.0	31.2	42218	42189.6	41.8	42218	**42202.2**	28.1	42218
6	42927	42911.0	33.3	42927	**42913.5**	32.6	42927	42903.5	40.8	42927	42913.0	34.0	42927	42908.0	39.5	42927
7	42009	41977.0	45.2	42009	**41985.2**	40.9	42009	41978.9	42.9	42009	41984.6	40.6	42009	41978.0	49.0	42009
8	45020	44971.0	32.5	45010	**44988.8**	22.1	45020	44984.4	29.2	45020	44969.9	35.5	45010	44979.7	31.9	45010
9	43441	**43356.0**	38.5	43441	43349.1	42.9	43441	43353.6	49.7	43441	43345.1	40.9	43441	43347.1	40.7	43441
10	44554	44506.0	25.2	44554	44512.8	23.5	44554	44513.9	25.5	44554	44510.4	28.2	44554	**44515.8**	25.2	44554
1	59822	59821.0	3.2	59822	**59822.0**	0.0	59822	59815.1	20.2	59822	**59822.0**	0.0	59822	**59822.0**	0.0	59822
2	62081	62010.0	47.1	62081	62010.7	44.1	62081	62003.6	44.8	62081	61994.7	31.0	62081	**62011.0**	49.0	62081
3	59802	59759.0	21.7	59802	59757.9	16.1	59802	59745.8	24.7	59802	59750.2	20.2	59802	**59760.1**	22.2	59802
4	60479	60428.0	21.8	60479	**60444.8**	27.3	60479	60417.2	30.0	60479	60438.6	24.2	60479	60435.8	23.6	60479
5	61091	61072.0	20.0	61091	61075.5	18.7	61091	61066.8	35.9	61091	**61078.4**	21.2	61091	61077.6	18.2	61091
6	58959	**58945.0**	14.5	58959	58940.6	12.3	58959	58929.4	35.5	58959	58943.9	19.4	58959	58943.3	14.1	58959
7	61538	**61514.0**	24.0	61538	61511.7	26.6	61538	61508.9	27.2	61538	61498.9	36.9	61538	61513.6	25.9	61538
8	61520	61492.0	25.6	61520	61494.0	22.7	61520	61473.7	34.7	61520	61475.4	32.7	61520	**61496.5**	23.0	61520
9	59453	**59436.0**	40.5	59453	59434.8	43.1	59453	59413.9	59.1	59453	59427.4	53.4	59453	59435.2	39.7	59453
10	59965	59958.0	8.4	59965	59956.0	11.2	59965	59944.9	26.9	59965	59946.0	25.8	59960	**59959.1**	5.2	59965

some $u\mathcal{D}$-characteristic that could bias the search process to poor quality solutions. We proposed to divide the search process into two steps: a *First Step* for learning about an $u\mathcal{D}$-characteristic and, a *Second Step* performed by a target ant-based algorithm. During the *First Step*, three sub-colonies of ants cooperate to define an initial pheromone matrix. Each sub-colony is guided by one *Method*: *SOL*, *HOL* and *WOL*. To evaluate our strategy, we used the well-known Ant Knapsack algorithm for solving the MKP. Our preliminary results show that the inclusion of *COISA* in Ant Knapsack improves its robustness and helps to obtain better quality solutions. Additionally, we were able to show that the cooperation between the three methods adopted is better than using only one of them in isolation. As part of our future work, we are interested in evaluating *COISA* in other ant-based algorithms for solving other combinatorial optimization problems. Also, we are interested in comparing *COISA* with other existing pre-processing schemes for ant-based algorithms.

Acknowledgements. Second author is partially supported by the Centro Científico Tecnológico de Valparaíso (CCTVal) Project No. FB0821. The first author is supported by CONICYT-PCHA/National Doctorate/2015-21150696. Third author acknowledges support from CONACyT project no. 221551.

References

1. Alaya, I., Solnon, C., Ghedira, K.: Ant algorithm for the multi-dimensional knapsack problem. In: International Conference on Bioinspired Optimization Methods and their Applications (BIOMA 2004). Citeseer (2004)
2. Cordon, O., de Viana, I., Herrera, F., Moreno, L.: A new aco model integrating evolutionary computation concepts: The best-worst ant system (2000)
3. Dorigo, M., Stützle, T.: Ant Colony Optimization. Bradford Company, Scituate, MA, USA (2004)
4. Malisia, A.R.: Investigating the application of opposition-based ideas to ant algorithms. Master's thesis, University of Waterloo (2007)
5. Malisia, A.R.: Improving the exploration ability of ant-based algorithms. In: Oppositional Concepts in Computational Intelligence, vol. 155, pp. 121–142. Springer Berlin Heidelberg (2008)
6. Malisia, A., Tizhoosh, H.: Applying opposition-based ideas to the ant colony system. In: 2007 IEEE Swarm Intelligence Symposium, SIS 2007. pp. 182–189 (2007)
7. Montgomery, J., Randall, M.: Anti-pheromone as a tool for better exploration of search space. In: Dorigo, M., Di Caro, G., Sampels, M. (eds.) Ant Algorithms, Third International Workshop, ANTS 2002. Lecture Notes in Computer Science, vol. 2463, pp. 100–110. Springer (2002)
8. Puchinger, J., Raidl, G.R., Pferschy, U.: The Core Concept for the Multidimensional Knapsack Problem. In: European Conference on Evolutionary Computation in Combinatorial Optimization. pp. 195–208. Springer (2006)
9. Riff, M.C., Montero, E.: A new algorithm for reducing metaheuristic design effort. In: Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013. pp. 3283–3290. IEEE (2013), http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6552460
10. Rojas-Morales, N., Riff, M.C., Montero, E.: Ants can learn from the opposite. In: Friedrich, T., Neumann, F., Sutton, A.M. (eds.) Proceedings of the 2016 on Genetic and Evolutionary Computation Conference, Denver, CO, USA, July 20 - 24, 2016. pp. 389–396. ACM (2016). https://doi.org/10.1145/2908812, http://doi.acm.org/10.1145/2908812
11. Rojas-Morales, N., Riff, M.C., Montero, E.: Learning from the opposite: Strategies for Ants that solve Multidimensional Knapsack problem. In: IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, July 24-29, 2016. pp. 193–200. IEEE (2016)
12. Rojas-Morales, N., Riff, M.C., Montero, E.: A survey and classification of opposition-based metaheuristics. Computers & Industrial Engineering **110**, 424–435 (2017)
13. Schoonderwoerd, R., Bruten, J.L., Holland, O.E., Rothkrantz, L.J.M.: Ant-based load balancing in telecommunications networks. Adaptive Behavior **5**(2), 169–207 (1996)
14. Stützle, T., Hoos, H.H.: MAX–MIN Ant System. Future generation computer systems **16**(8), 889–914 (2000)