A SHUFFLED COMPLEX EVOLUTION ALGORITHM FOR THE MULTIDIMENSIONAL KNAPSACK PROBLEM USING CORE CONCEPT

MARCOS BARONI & FLÁVIO VAREJÃO

UNIVERSIDADE FEDERAL DO ESÍRITO SANTO, VITÓRIA, ES, BRAZIL

THE PROBLEM (MKP)

The **multidimensional knapsack problem** (MKP) is a strongly NP-hard combinatorial optimization problem which can be viewed as a allocation problem with multiple resources:

$$\max \sum_{j=1}^{n} p_j x_j$$

$$\text{s. to} \sum_{j=1}^{n} w_{ij} x_j \leqslant c_i \quad i \in \{1, \ldots, m\}$$

$$x_j \in \{0,1\}, \quad j \in \{1, \ldots, n\}.$$

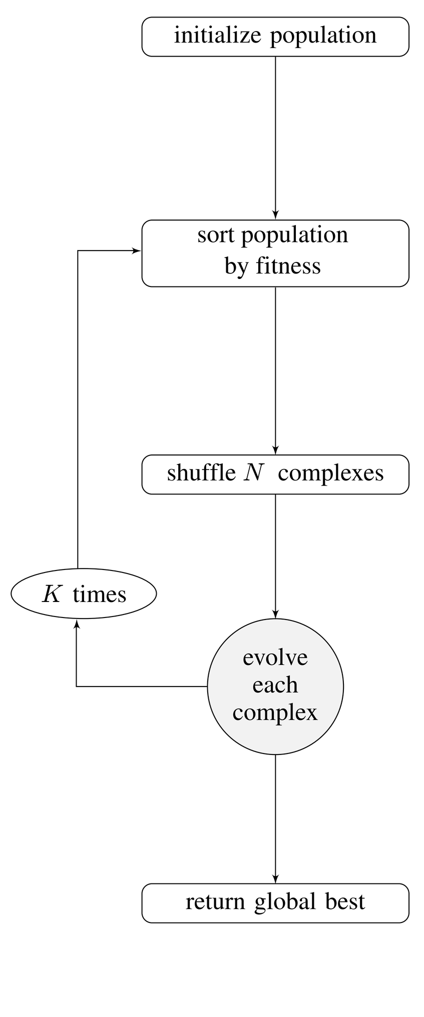
The problem can be applied on budget planning, cutting stock, loading problems and allocation of processors and databases in distributed computer programs.

This work addresses an hybrid heuristic using an efficient variable fixing procedure and the SCE metaheuristic as an improvement proposal for the work in [1].

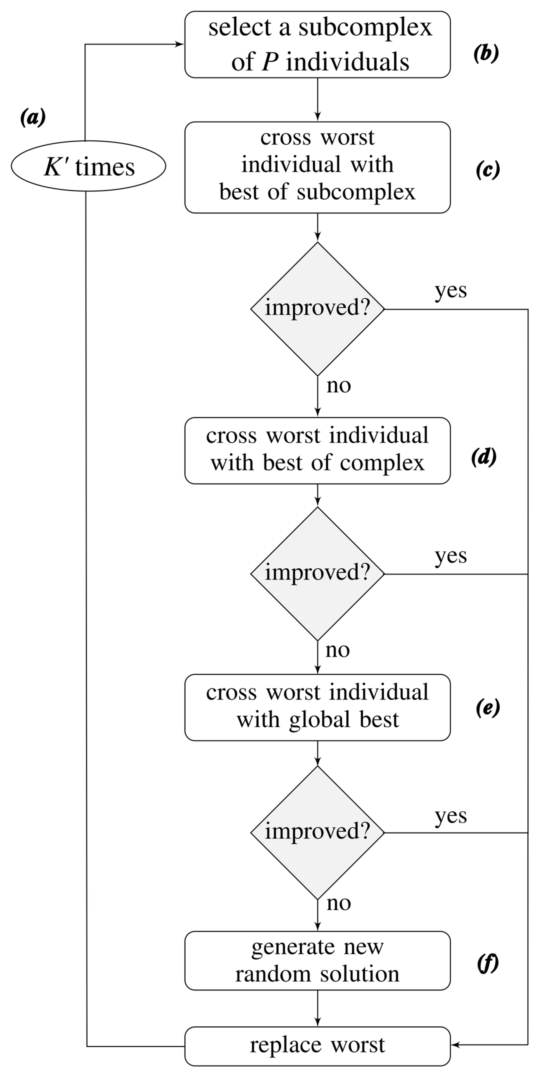
THE META-HEURISTIC (SCE)

The **shuffled complex evolution** (SCE) [3] is a population based evolutionary optimization algorithm that regards a natural evolution happening simultaneously in N independent communities (or **complexes**).

Initialy $N * M$ individuals are randomly taken from the feasible solution space and sorted according to their fitness. Subsequently a **shuffling** process places the 1st in the first complex, the 2nd in second complex, individual Nth goes to Nth complex, individual $M+1$ goes back to the first complex, etc.



The SCE algorithm.



Evolving stage.

The next step after shuffling the complexes is to evolve each each of then through a fixed amount of *(a)* K' steps. In each step *(b)* a **subcomplex** of P individuals is selected from the complex, prioritizing those with better fitness.

The worst individual from the subcomplex is identified to be replaced by a new one generated by *(c)* its crossing with best individual of the subcomplex.

If the new solution has not improved *(d)* the best individual of the **complex** is considered for crossing and latter the *(e)* best one of whole **population**. If all the crossing steps couldn't improve the worst individual, it is *(f)* replaced by a **new random** solution.

THE CORE CONCEPT FOR MKP

The **core** of a knapsack problem is a set items for which it is hard to decide if they will occur or not in a good solution. The best way to acoomplish this is determining an **efficiency** (or *cost-benefit*) measure e_j for each item.

Those items with low efficiency are fixed outside the knapsack while items with high efficiency are fixed inside the knapsack. The remaining items, with medium efficiency, items constitute the core of the knapsack.

core = {6,7,..,10}

	1	2	3	4	5	6	7	8	9	10	11	12	13
efficiency	1.89	1.85	1.79	1.73	1.65	1.48	1.26	1.24	1.15	0.94	0.55	0.53	0.29
var fixing	1.00	1.00	1.00	1.00	1.00	*	*	*	*	*	0.00	0.00	0.00

Example of core for a hypothetical MKP instance.

For the case of a single constraint (KP) the following efficiency measure can be considered

$$e_j = \frac{p_j}{w_j}.$$

Unfortunately on the MKP there are multiple resources (constraint) to be considered as cost. Each of then may have different relevance on the cost the item. This drawback can be avoided by introducing **relevance values** r_i for every constraint:

$$e_j(relevance) = \frac{p_j}{\sum_{i=1}^{m} r_i w_{ij}}$$

According to [5] setting r_i to the values of an optimal solution to the dual problem of the MKP's LP-relaxation, achieves the best results and will be the one considered in this work. The optimal solution to the dual problem measures the **scarcity** of that resource

Considering the items in the interval $C = \{a, \ldots, b\}$ as the core, the **MKP core problem (MKPC)** can be defined as:

$$\text{maximize} \sum_{j \in C} p_j x_j + \tilde{p} \tag{1}$$

$$\text{subject to} \sum_{j \in C} w_{ij} x_j \leqslant c_i - \tilde{w}_i, \quad i = 1, \ldots, m \tag{2}$$

$$x_j \in \{0,1\}, \quad j \in C. \tag{3}$$

with $\tilde{p} = \sum_{j=1}^{a-1} p_j$ and $\tilde{w}_i = \sum_{j=1}^{a-1} w_{ij}, i = 1, \ldots, m$ respectively quantifying the total profit and the total weights of items fixed as selected.

This work addresses the development of a SCE metaheuristic applied to MKP core problem. This hybrid SCE will be referred to as **SCEcr**.

RESULTS

Two main tests was considered. The first one was the set of problems defined by Chu and Beasley [2] and the seconf was the set composed by 11 instances defined in [4].

Columns **n** and **m** indicate the size of each instance, **time** column shows the average execution time (lower is better), **quality** column shows the average ratio of the solution found and the best known solution from literature, variance values shown in parentheses.

Table I: Performance on Chu-Beasley problems.

		time (s)		quality (%)	
n	**m**	**SCE**	**SCEcr**	**SCE**	**SCEcr**
100	5	1.31 (0.03)	0.17 (0.00)	97.60 (0.56)	99.83 (0.02)
	10	1.43 (0.04)	0.26 (0.00)	96.96 (0.99)	99.75 (0.04)
	30	1.75 (0.08)	1.01 (0.04)	96.66 (0.66)	98.89 (0.11)
250	5	2.87 (0.09)	0.69 (0.01)	94.98 (0.33)	99.92 (0.00)
	10	3.08 (0.09)	0.83 (0.01)	94.95 (0.35)	99.75 (0.00)
	30	3.82 (0.14)	1.45 (0.05)	94.65 (0.39)	98.89 (0.04)
500	5	5.74 (0.14)	1.23 (0.01)	93.73 (0.28)	99.86 (0.00)
	10	5.85 (0.34)	1.33 (0.03)	93.65 (0.25)	99.71 (0.00)
	30	6.17 (1.12)	1.84 (0.19)	93.30 (0.34)	99.28 (0.01)

Table II: Performance on Glover-Kochenberger problems.

			time (s)		quality (%)	
#	**n**	**m**	**SCE**	**SCEcr**	**SCE**	**SCEcr**
01	100	15	1.47 (0.00)	0.08 (0.0)	97.66 (0.03)	99.24 (0.02)
02	100	25	1.61 (0.00)	0.09 (0.0)	97.94 (0.04)	98.94 (0.09)
03	150	25	2.51 (0.01)	0.09 (0.0)	97.22 (0.04)	99.09 (0.02)
04	150	50	3.56 (0.03)	0.09 (0.0)	97.40 (0.04)	98.52 (0.02)
05	200	25	3.55 (0.01)	0.09 (0.0)	96.88 (0.03)	99.28 (0.01)
06	200	50	4.81 (0.09)	0.10 (0.0)	97.68 (0.02)	98.90 (0.03)
07	500	25	7.30 (0.09)	0.10 (0.0)	97.12 (0.01)	99.54 (0.00)
08	500	50	12.20 (0.47)	0.11 (0.0)	97.27 (0.01)	99.33 (0.01)
09	1500	25	24.61 (1.73)	0.12 (0.0)	95.40 (0.01)	98.22 (0.00)
10	1500	50	33.79 (2.44)	0.13 (0.0)	97.50 (0.00)	99.64 (0.00)
11	2500	100	121.28 (194.74)	0.15 (0.0)	97.95 (0.00)	99.70 (0.00)

It can be noticed that SCEcrachieved high quality solutions, at least 98.22% of best known solution, spending short amount of processing time.

EXPERIMENTAL SETUP

Several computational experiments was executed to verify the efficiency of the hybrid method (SCEcr) over the original SCE proposed in [1] (SCE). In all instances the **parameters** used for SCE and SCEcrwere the same recommended in [1] which was found after a batch test using Chu and Beasley instances:

	Value	Description
N	20	# of complexes
M	20	# of individuals in each complex
P	5	# of individuals in each subcomplex
K	300	# of algorithm iterations
K'	20	# of iterations of evolving process
c	$n/5$	# of genes carried from parent in crossing

The experiments were run on a 3.40GHz computer with 4GB of RAM. The algorithms was implemented in C programming language.

CONCLUSIONS & FUTURE REMARKS

In this work we addressed the application of the shuffled complex evolution (SCE) to the multidimensional knapsack problem and investigated it performance through several computational experiments.

The SCE algorithm, which combines the ideas of a controlled random search with the concepts of competitive evolution proved to be **effective** in finding good solution for hard instances of MKP, demanding short amount of processing time to reach high quality solutions for MKP.

Future work includes the investigation of different crossing procedures, the use of local search in the process of evolving complexes and the application of problem reduction procedures for the MKP.

THE SCE FOR MKP

As it can be noted in its description the SCE is easly applied to any optimization problem. The only steps needed to be specified is the creation of a **new MKP random solution** (Algorithm 1) and the **crossing procedure of two MKP solutions** (Algorithm 2).

Algorithm 1 Generation of a new random solution.

```
procedure NEW RANDOM SOLUTION
    v ← shuffle(1, 2, ..., n)
    s ← ∅                                  ▷ empty solution
    for i ← 1 : n do
        s ← s ∪ {v_i}                      ▷ adding item
        if s is not feasible then          ▷ checking feasibility
            s ← s − {v_i}
        end if
    end for
    return s
end procedure
```

Algorithm 2 Crossing procedure used on SCE algorithm.

```
procedure CROSSING(x^w : worst individual,
        x^b : better individual, c : n. of genes to be carried)
    v ← shuffle(1, 2, ..., n)
    for i ← 1 : c do
        j ← v_i
        x^w_j ← x^b_j                      ▷ gene carriage
    end for
    if s^w is not feasible then
        repair s^w
    end if
    update s^w fitness
    return s^w
end procedure
```

REFERENCES

[1] Marcos Daniel Valadão Baroni and Flávio Miguel Varejão. A shuffled complex evolution algorithm for the multidimensional knapsack problem. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 768–775. Springer, 2015.

[2] P. C. Chu and J. E. Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(1):63–86, June 1998.

[3] Qingyun Duan, Soroosh Sorooshian, and Vijai Gupta. Effective and efficient global optimization for conceptual rainfall-runoff models. *Water resources research*, 28(4):1015–1031, 1992.

[4] Fred Glover and Gary A Kochenberger. Critical event tabu search for multidimensional knapsack problems. In *Meta-Heuristics*, pages 407–427. Springer, 1996.

[5] Jakob Puchinger, Günther R Raidl, and Ulrich Pferschy. The core concept for the multidimensional knapsack problem. In *Evolutionary Computation in Combinatorial Optimization*, pages 195–208. Springer, 2006.

ACKNOWLEDGEMENT

Research supported by Fundação de Amparo à Pesquisa do Espírito Santo.