A Shuffled Complex Evolution Algorithm for The Multidimensional Knapsack Problem

Marcos Baroni & Flávio Varejão

The Problem (MKP)

The multidimensional knapsack problem (MKP) is a strongly NP-hard combinatorial optimization problem which can be viewed as a resource allocation problem and defined as follows:

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

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

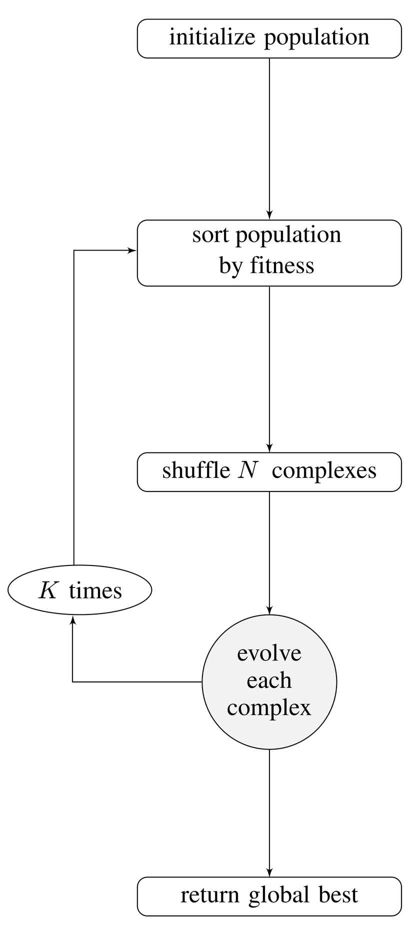
$$x_j \in \{0, 1\}, \quad j \in \{1, \dots, n\}.$$

The work address the application of a meta-heuristic called shuffled complex evolution (SCE) to the MKP.

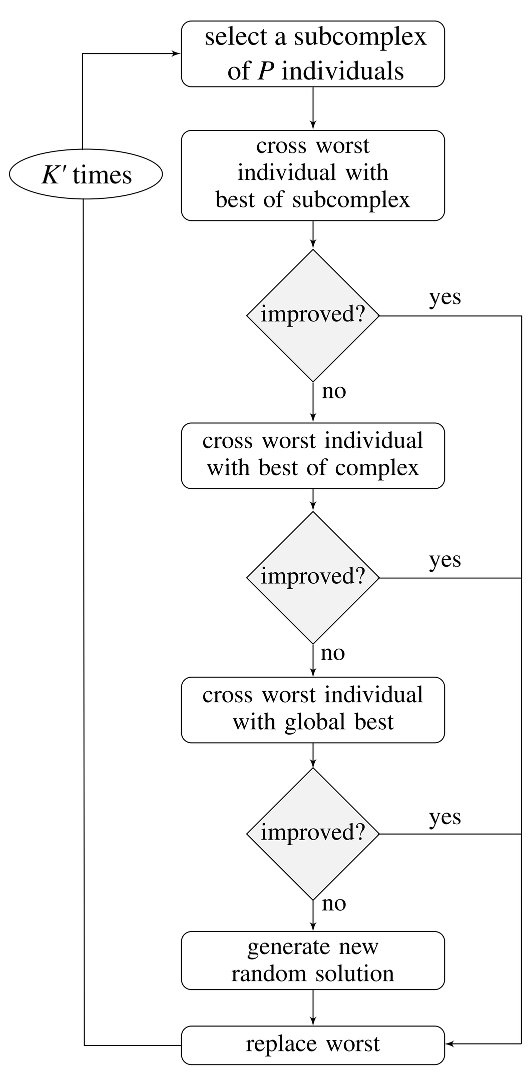
The Meta-heuristic (SCE)

The shuffled complex evolution 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 complex through a fixed amount of K' steps: the individuals in each complex is sorted by descending order of fitness quality. In each step a subcomplex of P individuals is selected from the complex, prioritizing individuals with better fitness.

Then the worst individual from the subcomplex is identified to be replaced by a new solution generated by its crossing with best individual of the subcomplex. If the new solution has not improved, the best individual of the complex is considered for crossing and latter the best individual of whole population. If all the crossing steps couldn't improve the worst solution, it is replaced by a new random solution.

Results

Two main tests was considered: (a) using the well-known set of problems defined by Chu and Beasley [1] and (b) a large set of randomly generated instances using uniform distribution. The number of constraints m varies among 5, 10 and 30, and the number of variables n varies among 100, 250 and 500.

n	m	SCE t (s)	gap (%)
100	5	0.81	97.6
	10	0.86	97.0
	30	1.02	96.7
	average gap		**97.1**
250	5	1.75	95.3
	10	1.83	95.0
	30	2.24	94.7
	average gap		**95.0**
500	5	3.23	93.7
	10	3.40	93.7
	30	3.91	93.3
	average gap		**93.6**

Chu-Beasley problems.

n	m	α	SCIP t (s)	SCE t (s)	gap (%)
500	10	0.25	278.85	2.23	96.1
		0.50	177.32	2.14	98.4
		0.75	8.47	1.87	99.6
	average gap				**98.0**
	20	0.25	284.11	2.30	96.7
		0.50	275.68	2.16	98.6
		0.75	33.67	1.90	99.7
	average gap				**98.3**
	30	0.25	283.78	2.50	96.9
		0.50	283.54	2.32	98.7
		0.75	71.66	1.96	99.7
	average gap				**98.3**

Random generated problems.

For the set of random instances all best known solution was found by the solver SCIP running for at least 10 minutes. SCIP is an open-source integer programming solver which implements the branch-and-cut algorithm.

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 random solution (Algorithm 1) and the crossing procedure of two 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_j^w ← x_j^b                      ▷ 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
```

Experimental Setup

A batch of preliminary tests was driven to find the best parameters for the problem:

	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

All the experiments was run on a 3.40GHz computer with 4GB of RAM. SCE lgorithm was implemented in C programming language.

Conclusions

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 very effective in finding good solution for hard instances of MKP, demanding a very small amount of processing time to reach high quality solutions for MKP.

Future Remarks

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.

References

References

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

[2] 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.

Acknowledgement

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