On the Effectiveness of Genetic Algorithms for the Multidimensional Knapsack Problem

Jean P. Martins
Institute of Mathematics and Computer Science
University of São Paulo, Brasil,
jean@icmc.usp.br

Humberto Longo
Informatics Institute
Federal University of Goiás, Brasil,
longo@inf.ufg.br

Alexandre C. B. Delbem
Institute of Mathematics and Computer Science
University of São Paulo, Brasil,
acbd@icmc.usp.br

ABSTRACT

In the Multidimensional Knapsack Problem (MKP) there are items easily identifiable as highly (lowly) profitable and likely to be chosen (not chosen) to compose high-quality solutions. For all the other items, the Knapsack Core (KC), the decision is harder. By focusing the search on the KC effective algorithms have been developed. However, the true KC is not available and most algorithms can only rely on items' efficiencies. Chu & Beasley Genetic Algorithm (CBGA), for example, uses efficiencies in a repair-operator which bias the search towards the KC. This paper shows that, as the search progresses, efficiencies lose their descriptive power and, consequently, CBGA's effectiveness decreases. As a result, CBGA rapidly finds its best solutions and stagnates. In order to circumvent this stagnation, extra information about the KC should be used to implement specific operators. Since there is a correlation between marginal probabilities in a population and efficiencies, we show that KCs can be estimated from the population during the search. By solving the estimated KCs with CPLEX, improvements were possible in many instances, evidencing CBGA's weakness to solve KCs and indicating a promising way to improve GAs for the MKP through the use of KC estimates.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Global Optimization—*Evolutionary Combinatorial Optimization and Metaheuristics*

Keywords

Knapsack Problems; Genetic Algorithms; Knapsack-Core

1. INTRODUCTION

The *Multidimensional Knapsack Problem* (MKP) is an well-known NP-Hard combinatorial optimization problem, defined as the Integer Linear Programming (ILP) model (1)-(3) [2,3]. The objective is to choose the most profitable subset of items (1), respecting m capacity constraints (2) and not considering fractional items (3). The MKP generalizes the simpler knapsack problem, in which there is only $m = 1$ constraint.

$$\max \quad f(\mathbf{x}) = \sum_{j=1}^{\ell} p_j \cdot x_j, \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{\ell} w_{ij} \cdot x_j \leq c_i, \qquad i = 1, \ldots, m, \tag{2}$$

$$x_j \in \{0, 1\}, \qquad j = 1, \ldots, \ell. \tag{3}$$

By definition, there are $\ell \geq 1$ items available, with profits $p_j > 0$ $(j = 1, \ldots, \ell)$, and $m \geq 1$ resources available in amounts $c_i > 0$ $(i = 1, \ldots, m)$. For each item chosen, p_j is incremented in the objective value (total profit) and the quantity w_{ij} is decremented from the i^{th} resource. Some items are more profitable than others, some consume less resources than others, those with a high ratio between their profit and resource consumption (called *efficiency*) are likely to be in optimal solutions. On the other hand, those items with low efficiencies are likely to be left outside the knapsack $(x_j = 0)$ in high-quality solutions. We may say that all the other items compose the *Knapsack Core* (KC).

The *Chu & Beasley Genetic Algorithm* (CBGA) [1] uses such efficiencies to guide the search [4]. However, at some point of the evolutionary process efficiencies lose their power, from that point, the quality of solutions will improve slowly due to the NP-hardness of the KC (first assumption). Fortunately, during the search the population accumulates information about the KC, and we can expect the KC being estimated from items' marginal probabilities – those with $0 < p_{X_i}(x = 1) < 1$ – (second assumption).

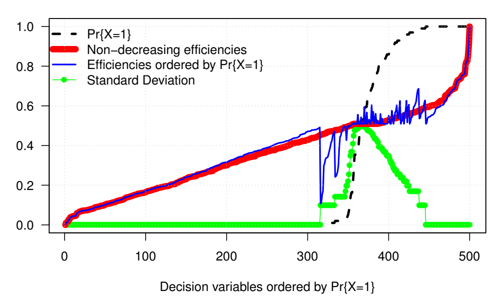

Figure 1: Marginal probabilities in non-decreasing order, after $100,000$ **fitness evaluations.**

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598477 .

Table 1: Results for instances `10.500.*`

Alpha	Instance	FH time(h)	CBGA Time(h)	Stag. Time(%)	f(x)	f(x*)	CBGA Gap	|C|	CPLEX Gap	CPLEX Time(h)
0.25	10.500-00	0.49	6.72	92.64	117790	117821	31	72	**0**	4
0.25	10.500-01	0.16	3.45	95.5	119208	119249	41	82	**17**	4
0.25	10.500-02	2.64	6.84	61.32	119194	119215	21	72	**4**	4
0.50	10.500-10	2.24	4.43	49.34	217365	217377	12	73	12	3.09
0.50	10.500-11	5.63	6.79	17.08	219053	219077	24	72	**2**	4
0.50	10.500-12	1.8	5.05	64.35	217792	217847	55	68	**0**	0.31
0.75	10.500-20	0	4.35	99.97	304344	304387	43	75	**17**	1.19
0.75	10.500-21	0.37	6.07	93.97	302370	302379	9	77	**8**	1.13
0.75	10.500-22	1.93	4.88	60.41	302408	302417	9	74	**3**	1.46
–	–	2.3±1.9	5.7±1.1	62.5±28.6	–	–	27.4±17.2	–	10.6±10.1	2.3±1.6

Figure 1(a) shows in red (thicker line) the efficiencies e_j^{dual} sorted in non-decreasing order and scaled between $[0,1]$. The dotted line (in black), shows the marginal probabilities also sorted in non-decreasing order. In blue (thin line) we show the efficiencies ordered according two criteria: 1) the marginal probabilities, 2) the efficiencies itself. At last, we see in green dots that the KC is most probably composed of items with larger standard deviation in the population.

2. EXPERIMENTS

The experiments were performed with all MKP instances provided by Chu and Beasley [1]. For each instance the CBGA was run once and the last population was used to estimate the KCs. The best solution in the population was set as initial solution and the subproblem defined by the KC items solved using CPLEX. The solver was allowed to run for a maximum of 4 hours in a multi-core processor[1].

Table 1 shows the results for some instances with $\ell = 500$ and $m = 10$. CBGA was only able to find the best known solution in instance `10.500-13`. After applying `CPLEX` to the KC (of sizes $|C|$) most of the gaps (CPLEX Gap) were decreased, as shown in bold, and the best known solution was found for more instances. However, the stagnation time (62% in average) represents most of CBGA's running time.

These results agree with all of our previous hypothesis. It shows that the original CBGA cannot keep its effectiveness in long-runs, due to the decrease of the information provided by items' efficiencies along the generations. This fact is shown by high stagnation times, from which we can conclude that CBGA will spend most of its time without improvements in a long-run. On the other hand, we also showed that information about the KC can be extracted from statistics measured from CBGA's population. The improvements obtained by `CPLEX` when solving the subproblems defined by the estimated KCs proves the validity of such approach.

3. CONCLUSIONS

This study analyzed the effectiveness of CBGA in solving MKPs. CBGA was one of the first successful *Genetic Algorithms* (GAs) for the MKP and various new methods were developed based in its basic principles. In general, CBGA is a simple steady-state GA, however, it originally proposed the use of items' efficiencies to implement a repair-operator that was able to rapidly bias the search towards high-quality solutions.

[1] `Intel(R) Core(TM) i7-2600 CPU, 3.40GHz`

We showed that by using efficiencies CBGA can only be effective by a certain amount of time. After some generations, when most of highly/lowly efficient items have already converged, the search stagnates. This occurs because the remaining items' efficiencies are too much similar and the repair-operator is not able to bias the search anymore. As a consequence of this fact, even after a very long-run CBGA is not able to find the best known solution for harder instances. In fact, we showed that the best solution found in a long-run appears in the beginning of the search, followed by very long stagnation times after that.

Another important characteristic in solving MKPs is the GA's ability to converge non-KC decision variables very fast, in order to focus the search in the harder subproblem defined by the KC. We showed that the CBGA is not able perform this task, after its stagnation the algorithm also stops the convergence of non-KC variables, beginning a drift-like behavior. Therefore, when considering larger population sizes, a smaller subsets of variables converge presenting some difficulties in focusing the appropriate regions.

Although we used a two-step procedure, running the CBGA first then extracting information from the population. We believe that in a general context, the analysis provided in this study can help to understand and develop more effective GAs for the MKP.

Acknowledgments

The authors are indebted to FAPESP (São Paulo Research Foundation) for the financial support provided for this research (process number 2011/07792-4).

4. REFERENCES

[1] P. Chu and J. Beasley. A Genetic Algorithm for the Multidimensional Knapsack Problem. *Journal of Heuristics*, 4:63–86, 1998.

[2] A. Fréville. The multidimensional 0–1 knapsack problem: An overview. *European Journal of Operational Research*, 155(1):1–21, 2004.

[3] J. Puchinger, G. R. Raidl, and U. Pferschy. The multidimensional knapsack problem: Structure and algorithms. *INFORMS Journal on Computing*, 22(2):250–265, 2010.

[4] J. Tavares, F. Pereira, and E. Costa. Multidimensional Knapsack Problem: A Fitness Landscape Analysis. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(3):604–616, june 2008.