Naive and Heuristic Permutation-Coded Genetic Algorithms for the Quadratic Knapsack Problem

Bryant A. Julstrom
Computer Science and Information Technology
St. Cloud State University
St. Cloud, MN 56301 USA
julstrom@stcloudstate.edu

ABSTRACT

The Quadratic Knapsack Problem extends the 0-1 Knapsack Problem by associating values not only with individual objects but also with pairs of objects. Two genetic algorithms encode candidate solutions to this problem as permutations of objects. One GA applies no heuristic steps while a second seeds its population and performs crossover by considering the values of objects relative to the objects already in the knapsack. Both algorithms perform well, and competitively with two earlier binary-coded GAs. The heuristic measures improve performance significantly.

Categories and Subject Descriptors

G.2.1 [**Mathematics of Computing**]: Discrete Mathematics—*Combinatorics*; I.2.8 [**Problem Solving, Control Methods, and Search**]: Heuristic Methods

General Terms

Algorithms

Keywords

Quadratic Knapsack Problem, permutation coding, heuristic operators, genetic algorithm

1. INTRODUCTION

The Quadratic Knapsack Problem (QKP) [3] extends the familiar 0-1 Knapsack Problem with values attached to pairs of objects. The goal remains maximizing the total value of objects in the knapsack without overfilling it, but the value of a pair accrues to the knapsack when it contains both objects. Two genetic algorithms for the QKP encode candidate selections of objects as permutations.

The first GA applies no heuristic techniques, while the second considers objects' values relative to the objects already chosen for the knapsack in seeding the population and in crossover. The two GAs compete effectively with two earlier binary-coded algorithms, and the heuristic steps in the second algorithm improve performance markedly.

2. A PERMUTATION CODING

Candidate selections of objects for the knapsack can be represented by permutations of integers that represent the objects. A decoder scans the objects in their order in a permutation and places in the knapsack every object that fits. The sum of the included objects' individual and joint values is the permutation's fitness.

A mutation operator independently swaps values at random positions in a chromosome; the maximum number of swaps is a parameter of the operator. Crossover is alternation [5], which interleaves two parent permutations, preserving only the first appearance of each integer.

3. HEURISTIC STEPS

An object's value density r_i relative to a set S of objects is the sum of its value and the values it shares with objects in S, divided by its weight:

$$r_i = \frac{v_i + \sum_{j \in S} v_{i,j}}{w_i}.$$

The second genetic algorithm applies two heuristic steps based on objects' relative value densities.

Heuristic initialization seeds the GA's population with one chromosome generated by placing a random object in the knapsack, then repeatedly appending to the permutation the object the knapsack can accommodate that has the largest value density relative to the objects already chosen.

Similarly, *heuristic alternation* builds one offspring from two parents, beginning with the first object in the first parent. It the repeatedly examines the next unexamined objects in both parents. If only one fits, it joins the offspring. If neither fits, neither joins the offspring. If both fit, the next one to join the offspring is the object with the larger value density relative to the objects already listed in the offspring.

In both operations, when the knapsack can accommodate no more objects, the chromosome is completed by listing the unused objects in random order.

4. TWO GENETIC ALGORITHMS

Two generational genetic algorithms for the QKP encode candidate selections of objects as permutations and apply the operators just described. Each GA initializes its population with randomly-generated permutations and selects parents in k-tournaments without replacement. They generate each offspring via either crossover or mutation, are 1-elitist, and run through fixed numbers of generations.

When executed on a QKP instance with n objects, the GAs' populations contain n chromosomes. The probability that crossover generates an offspring is $P[Cr] = 0.35$, that of mutation therefore $P[Mu] = 1 - 0.35 = 0.65$. The algorithms run through $10n$ generations.

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

Table 1: Results of the trials of the naive and heuristic permutation-coded genetic algorithms on the set of QKP instances of Billionnet and Soutif. For each instance, the table lists its optimum solution value and the ratio of its knapsack's capacity C to the sum of its objects' weights. For each set of trials by each GA, it lists the number of trials (out of 50) that found optimum solutions (hits), the maximum solution value found, the mean solution value, the mean error of the solutions, and their standard deviation. The bottom line lists the total numbers of hits and the mean overall error in the two sets of 500 trials.

Instance			Naive GA					Heuristic GA				
Num	Opt	$C/\sum w_i$	Hits	Max	Mean	%E	StdDev	Hits	Max	Mean	%E	StdDev
1	18558	0.26	9	18558	18492.3	-0.35	61.35	43	18558	18556.9	-0.01	3.05
2	56525	0.85	50	56525	56525.0	0.00	0.00	50	56525	56525.0	0.00	0.00
3	3752	0.06	23	3752	3731.5	-0.55	19.60	32	3752	3739.4	-0.34	16.97
4	50382	0.76	24	50382	50353.6	-0.06	63.89	50	50382	50382.0	0.00	0.00
5	61494	0.93	50	61494	61494.0	0.00	0.00	50	61494	61494.0	0.00	0.00
6	36360	0.54	25	36360	36276.9	-0.23	97.06	50	36360	36360.0	0.00	0.00
7	14657	0.19	40	14657	14645.6	-0.08	23.14	48	14657	14655.6	-0.01	8.54
8	20452	0.26	28	20452	20406.4	-0.22	55.64	50	20452	20452.0	0.00	0.00
9	35438	0.57	3	35438	35344.4	-0.26	53.06	50	35438	35438.0	0.00	0.00
10	24930	0.40	10	24930	24878.7	-0.21	55.40	47	24930	24929.9	-0.00	0.48
			262			-0.20		470			-0.04	

In the naive GA, selection tournaments have $k = 3$ contestants, and mutation performs one, two, or three swaps. In the heuristic GA, selection tournaments have $k = 2$ contestants, and mutation performs up to five swaps.

The GAs were implemented and executed on an HP dc7600 with a P4 processor running at 3.2GHZ with 1Gbyte of RAM under Fedora 13 Linux.

5. TESTS

Billionnet and Soutif have posted a collection of randomly-generated QKP instances[1]. The two GAs were run 50 independent times on ten of these, with $n = 100$ objects. In these instances, 25% of the objects' values, individual and joint, are non-zero, and they have ratios $C/\sum w_i$ that range from 0.06 to 0.93. These instances have been solved to optimality [2] [1]. Table 1 summarizes the results of these trials.

The naive GA performed well. It found an optimum solution at least three times on every instance, and every trial hit on two instances. The mean error over all the trials was -0.20%, and the total number of hits was 262 out of 500.

The heuristic GA improved decisively on the performance of the naive GA. It identified an optimum solution at least 32 times on every test instance, and every trial hit on six of the ten instances. The mean error over all the trials was only -0.04%, and the total number of hits was 470 in the 500 trials.

These results compare favorably with those of two earlier binary-coded GAs for the QKP [4], again one naive and one with heuristic enhancements. On the same test instances, those algorithms found optimum solutions on 281 trials out of 500 with average error -0.18%, and on 446 trials out of 500 with average error -0.03%, respectively.

6. CONCLUSION

Two genetic algorithms for the QKP encode candidate selections of objects as permutations. A decoding algorithm identifies the selection of objects such a chromosome represents by scanning the objects in chromosome order and including all that fit. The value of the resulting set of objects is the chromosome's fitness.

One GA applies no heuristic devices. A second seeds its population with one permutation generated by examining the objects in order of their value densities relative to the objects already in the knapsack, and it extends alternation crossover by choosing each next object from the parental candidates according to objects' value densities relative to the objects already in the offspring's knapsack.

In tests on ten QKP instances whose optimum knapsack values are known, both versions of the GA performed well, and competitively with two binary-coded GAs. The heuristic operators improved the GA's performance significantly.

[1] http://cedric.cnam.fr/~soutif/QKP/

7. REFERENCES

[1] A. Billionnet and E. Soutif. An exact method based on Lagrangian decomposition for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 157(3):565–575, 2004.

[2] A. Billionnet and E. Soutif. Using a mixed integer programming tool for solving the 0-1 quadratic knapsack problem. *INFORMS Journal on Computing*, 16(2):188–197, 2004.

[3] G. Gallo, P. L. Hammer, and B. Simeone. Quadratic knapsack problems. *Mathematical Programming*, 12:132–149, 1980.

[4] B. A. Julstrom. Greedy, genetic, and greedy genetic algorithms for the Quadratic Knapsack Problem. In H.-G. Beyer et al., editors, *Proceedings of the 2005 Genetic and Evolutionary Computation Conference, GECCO-2005*, volume 1, pages 607–614, New York, 2005. ACM Press.

[5] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic. Genetic algorithms for the Traveling Salesman Problem: A review of representations and operators. *Artificial Intelligence Review*, 13:129–170, 1999.