A Parallel Evolutionary Algorithm for Unconstrained Binary Quadratic Problems

István Borgulya
Faculty of Business and Economics
University of Pécs, Hungary
Tel.: 36 72 501599
borgulya@ktk.pte.hu

ABSTRACT
In this paper an island model is described for the unconstrained Binary Quadratic Problem (BQP), which can be used with up to 2500 binary variables. Our island model uses a master-slave structure and the migration is centralized. In the model a basic evolutionary algorithm (EA) runs which is a hybrid, steady-state EA. The basic EA uses a new mutation operator that is composed of two parts and based on a modified version of an explicit collective memory method (EC-memory), the Virtual Loser [2].We tested our island model on the benchmark problems from the OR-Library. Comparing the results with other heuristic methods, we can conclude that our algorithm is highly effective in solving large instances of the BQP; it has a high probability of finding the best-known solutions.

Categories and Subject Descriptors
12.8 [**Artificial Intelligence**]: Problem Solving, Control Methods and Search – heuristic methods

General Terms
Algorithms.

Keywords
Binary quadratic programming; evolutionary algorithm; island model; EC-memory.

1. INTRODUCTION
In the BQP, a symmetric *n x n* matrix $Q=(q_{ij})$ is given, and a binary vector *X* of length *n* is desired, which maximizes the objective function

$$f(X) = X^tQX = \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}X_iX_j, \qquad X_i \in \{0,1\}\forall i$$

BQP has a central role in combinatorial optimization. A large number of problems can be formulated as maximization of quadratic real values function in 0-1 variables.

In this paper, we present now a new EA for the BQP. First we develop a memetic algorithm, a "basic EA", next we organize a parallel, island model with the help of the basic EA. The basic algorithm is a hybrid, steady-state EA. It uses a truncation selection, a new mutation operation, and the used local search method is the randomized k-opt procedure. The mutation operation is composed of two parts and based on a new modified version of the Virtual Loser method.

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

2. THE ALGORITHM
The main attributes of the basic EA are the following:

- The basic EA uses 2-stage algorithm structure. The first stage is a quick "preparatory" stage; the second is a hybrid EA.
- We do not use recombination operator.
- We use different mutation operator by the Virtual Loser method that was defined in [2]. In our operation the first part of the mutation operator makes random bit mutation, and the second part makes random bit mutation based on the Virtual Loser. The algorithm has maximum n/5 bit mutation in every generation in both parts of the mutation (it decreases a little bit during the evolutionary process). The number of the bit mutations was chosen based on the fitness landscape analysis in [1].
- We modified the update technique of the Virtual Loser method. We break periodically the update of the memory of the Virtual Loser and continue the evolution process with an empty memory.
- We speed up the convergence by a restart technique and by two special procedures. A *Filter* procedure selects only the best of the individuals close to each other, the other ones are deleted; and a *Delete* procedure deletes some worst individuals based on the fitness function periodically.

To improve the quality of the results and the running time we developed a parallel EA, an island model. In our island model:

- The basic EAs run on 2, 4, 8 or 16 parallel processors.
- The model uses a master-slave structure. It uses a centralized scheme in which slave processors execute the basic EA on their population and periodically send their best partial results to a master process. The master process stores the partial results in a common migration set (*MS*), and chooses random individuals from MS one after the other for every slave and sends them to the slaves.

(With this island model we wanted to examine the performance of the island model's structure, so we simulated the island model in one processor).

Table 1. Average solution values for large scale problems without the island model

Task	Glov500				B1000				B2500			
	Best known	AD	AT/ gen	b/ 20	Best known	AD	AT/ gen	b/ 20	Best known	AD	AT/ gen	b/ 20
1	61194	0	9/51	20	371438	0	133/113	20	1515944	0	761/118	20
2	100161	0	10/48	20	354932	0	73/76	20	1471392	0.0053	2243/252	12
3	138135	0	17/62	20	371236	0	53/58	20	1414192	0	827/129	20
4	172771	0	30/97	20	370675	0	75/83	20	1507701	0	232/54	20
5	190507	0	23/75	20	352760	0	106/98	20	1491816	0	1071/155	20
6					359629	0	89/87	20	1469162	0	1159/166	20
7					371193	0	159/126	20	1479040	0	1832/237	20
8					351994	0	183/139	20	1484199	0	1407/186	20
9					349337	0	223/168	20	1482413	0	1394/229	20
10					351415	0	107/92	20	1483355	0	1238/174	20
Aver.		0	18/67			0	120/104			0.0005	1216/170	

3. COMPUTATION EXPERIENCES

We tested the basic EA with the benchmark set from the OR-Library. We show only three benchmark sets (the large scale problems): two sets with 1000 and 2500 dimensions and the set with 500 dimensions (notation: B1000, B2500, glov500). In each set there are 5 or 10 instances. (The algorithm was implemented in Visual Basic and ran on a Pentium 4 1.8 GHz with 256 MB RAM).

Table 1 and table 2 present mean values calculated from a number of runs without and with parallel computing; all problems were run 20 times (The running was finished if the number of the generations was exceeded 400). In the tables we give the problem name (task), the average relative percentage deviation of the solution from the best known solution (AD) and the average running time in seconds (AT) with the average generation numbers (gen) to the best solutions. In table 1 we give how many best-known solutions could find the algorithm within the predefined number of generation (b/20).

We can say that our island model is successful; we could improve both the quality and the running times of the basic EA. The island model found the best-known solutions in all runs and the average running times are 2, 3 or 4 times shorter that on one processor without the island model.

We compared our results with one of the best method's results [1]. We can conclude that our model has similar results and it can find the best-known solutions with a high probability.

Table 2. Average solution values for large scale problems on 16 parallel processors

Task	Glov500		B1000		B2500	
	AD	AT/gen	AD	AT/gen	AD	AT/gen
1	0	7/49	0	26/49	0	233/58
2	0	4/71	0	22/44	0	1012/132
3	0	6/47	0	27/49	0	295/64
4	0	7/48	0	30/51	0	136/47
5	0	8/47	0	27/48	0	173/50
6			0	26/50	0	379/73
7			0	38/53	0	574/96
8			0	40/64	0	616/91
9			0	38/55	0	374/73
10			0	37/55	0	290/64
Average	0	6/46	0	31/52	0	408/75

4. ACKNOWLEDGEMENT

The Hungarian Research Foundation OTKA K 68137 supported the study.

5. REFERENCES

[1] Merz, P. and Katayama, K. A Hybrid Evolutionary Local Search Approach for the Unconstrained Binary Quadratic Programming Problem. *Bio Systems*. Vol. 78. No. 1-3. pp 99-118.

[2] Sebag, M., Schoenauer, M. and Ravisé, C. Toward Civilized Evolution: Developing Inhibitions. In: *Bäck T (ed): Proc. of the 7th International Conference on Genetic Algorithm.* Morgan Kaufmann Pub. San Francisco, 1997. pp 291-298.