UDC 519.854

V.P. SHYLO, F. GLOVER, I.V. SERGIENKO

TEAMS OF GLOBAL EQUILIBRIUM SEARCH ALGORITHMS FOR SOLVING WEIGHTED MAXIMUM CUT PROBLEM IN PARALLEL

Abstract. In this paper, we investigate the impact of communication between optimization algorithms running in parallel. In particular we focus on the weighted maximum cut (WMAXCUT) problem and compare different communication strategies between teams of GES algorithms running in parallel. The results obtained by teams encourage the development of team algorithms. They were significantly better than the algorithmic portfolio (no communication) approach and suggest that the communication between algorithms running in parallel is a promising research direction.

Keywords: weighted maximum cut problem, global equilibrium search, path relinking, team of algorithms, parallel optimization.

INTRODUCTION

In this paper, we investigate the impact of communication between optimization algorithms running in parallel. In particular we focus on the weighted maximum cut (WMAXCUT) problem and compare different communication strategies between teams of global equilibrium search (GES) algorithms running in parallel. The impact of communication is analyzed by comparing the results to the algorithm portfolio approach, which does not include any communication between its constituents.

The WMAXCUT problem is *NP*-hard: the computational requirements grow exponentially as the problem size increases. This leads to computational intractability when dealing with large scale instances or in applications with strict time constraints. Parallel computing tools can accelerate optimization algorithms; however, this potential is difficult to unleash due to a variety of issues when parallelizing serial search procedures. Effective utilization of modern high performance computing requires specialized research in the area of parallel optimization algorithms.

The weighted maximum cut problem is a classical problem of discrete optimization, which recently gathered a lot interest due to a number of important practical applications. An overview of some of the algorithms for this problem is given in [1]. In recent years, the stochastic method of GES [2, 3] was successfully used for general discrete optimization problems, and various instances of the GES approach [1, 4, 5]) have been developed to solve the problem WMAXCUT.

WEIGHTED MAXIMUM CUT PROBLEM

The WMAXCUT problem is a classic discrete optimization problem with numerous practical applications [6, 7]. Given an undirected graph $G = G(V; E)$, where V is a set of nodes and E is a set of weighted edges, the objective is to partition the set V into two disjoint subsets, a so-called cut, in such a way that the sum of weights corresponding to the edges that connect the two subsets is maximized.

The problem is known to be *NP*-hard even for the case when all weights have the same value [8]. The polynomial-time algorithm for finding maximum cuts in planar graphs was proposed in [9]. In general, exact methods can handle only small to medium problem sizes [10]. The approximation algorithm [11] is using a semidefinite programming relaxation to generate a maximum cut with a cost that is on average at least 0.87856 times the optimal value. However, its CPU and memory requirements are not scalable for large problem sizes. Approximate heuristic methods are successfully used to solve large scale problems [12–15]. The GES method is an efficient meta-heuristic approach that demonstrates state-of-the-art performance in many applications, including the WMAXCUT [1–5].

Consider an undirected graph $G = G(V; E)$, where V is a set of vertices and E is the set of edges. Every edge $(i, j) \in E$ is characterized by a weight w_{ij}. Given a partition (V_1, V_2) of the set of vertices V, where V_1 and V_2 are non-overlapping and their union is equal to V, a cut is defined as a set of edges $(i, j) \in E$ such that $i \in V_1$ and $j \in V_2$. The weight of the cut is the sum of the weight of its edges:

$$w(V_1, V_2) = \sum_{i \in V_1,\, j \in V_2,\, (i,\, j) \in E} w_{ij} \quad .$$

In the WMAXCUT problem, the objective is to find a cut with the maximum weight.

The WMAXCUT problem can be formulated using a mixed integer programming model [16]:

$$\begin{gathered} \max \sum_{i,\, j=1,\, i<j} w_{ij}\, y_{ij}, \\ \text{s.t. } y_{ij} - x_i - x_j \le 0, \; i, j = 1, \ldots, n, \; i < j, \\ y_{ij} + x_i + x_j \le 2, \; i, j = 1, \ldots, n, \; i < j, \\ x \in \{0, 1\}^n. \end{gathered}$$

This model assumes that the weights are non-negative, and its solution defines a graph partition $\{V_1, V_2\}$(if $x_i = 1$ then $v_i \in V_1$, otherwise $v_i \in V_2$) that has the maximum cut value.

The WMAXCUT problem also can be formulated using the Unconstrained Binary Quadratic Programming model:

$$\max_{x_i \in \{0,1\}} \left\{ f(x) = \sum_{(i,\, j) \in E} w_{ij} (x_i - x_j)^2 \right\} .$$

PORTFOLIOS AND TEAMS OF OPTIMIZATION ALGORITHMS

In this paper, we focus on the parallel optimization framework that consists of a set of algorithms running on different processors in parallel. Each algorithm undertakes to solve the same problem, called the target problem.

The algorithms in the set can be either independent, or they may communicate certain information about the search space of the target problem during their execution. To distinguish these two cases, we will call a set of independent algorithms an algorithm portfolio, and will call the set of algorithms that communicate a team of algorithms.

There may be multiple portfolios of algorithms that work independently of each other (yet which are classified as a single unit) just as there may be multiple teams whose members share information with each other. Consider some set of available algorithms $A = \{A_1, \ldots, A_m\}$ that can be executed by any of the P available processors. Each processor should be assigned to an algorithm from A, while an algorithm may be assigned to multiple processors. The latter case makes sense when dealing with randomized algorithms, where each copy of the same algorithm uses a distinct seed value for initializing its pseudorandom number generator. In this case, although the algorithm deployed on different processors is the same, the difference in seed values leads to different search trajectories.

Teams of algorithms, as well as algorithm portfolios, can be described by a list that specifies how many processors are assigned to each algorithm from A: $\{(n_1 A_1), (n_2 A_2), \ldots, (n_m A_m)\}$, where $\sum_{i=1}^{m} n_i = P$.

When solving an optimization problem by a team or portfolio of algorithms, performance is measured with respect to the best algorithm in the team or portfolio, that is, the one that takes the least time to solve the target problem. In other words, if t_i is the computational time elapsed before finding solution by algorithm on ith processor, then the corresponding computational time for the portfolio or the team, which we designate as a *unit*, is measured by $t_{unit} = \min_{i=1,\dots,P} t_i$.

The theoretical properties of optimal portfolios of restart algorithms were considered in [17]. In [18] the authors provided a sharp upper bound on the maximum speedup coefficient of the portfolios consisting of different algorithms when compared to a single algorithm portfolio. The work of [19] proposes and investigates various algorithms portfolios.

Treatment of same objective values. GES algorithm keeps track of the best found solution, x_{best}, and the best solution after the last restart of the search procedure or the current best, $x_{\max}$. During the execution of a given algorithm, upon finding a new solution x with the same objective value as $x_{\max}$ obtained by that algorithm, we have three options:

(St) stay with (retain) the current best solution (without replacing $x_{\max}$);

(Mv) switch (update) the current best solution to x (by setting $x_{\max} = x$);

(Mc) switch (update) the current best solution to x (by setting $x_{\max} = x$) under some condition.

Similar choices are appropriate for updating x_{best}. By choosing different update rules for $x_{\max}$ and x_{best}, we can define nine different variations of the search algorithm. For some problems, the number of solutions with the same objective value is measured by hundreds of thousands, thus these strategies can behave very differently. These variations together with different search methods used by GES procedures provide a wide spectrum of algorithms for composing optimization teams.

Figure 1 shows a typical example of computational acceleration achieved by portfolios of four identical algorithms. If a time to solve a problem by a serial algorithm is t_{serial}, and the time to solve a problem by a portfolio of algorithms on P processors is $t_{\text{portfolio}}$, then the coefficient of acceleration is $t_{\text{serial}} / (t_{\text{portfolio}} * P)$. In particular, this graph shows the coefficient of computational acceleration for the set of instances G1-G54 achieved by 4 different algorithm portfolios considered in [19]: MvSt1GES, MvSt2GES, StSt1GES, and StSt2GES. The naming convention of theses algorithms is based on the choice of search strategies, the first two letters refer to the update rule for $x_{\max}$, the third and fourth letters refer to the update rule for x_{best}, and the following number identifies which type of tabu search procedure is used. The difference between the tabu procedure 1 and the tabu procedure 2 is following: the first tabu procedure performs a fixed number of iterations when attempting to improve $x_{\max}$, and in the second tabu procedure the number of iterations depends on the previous search results (increased if there was a recent improvement). For example, MvSt1GES always updates the current best solution, $x_{\max}$, always retains the best solution, x_{best}, and uses the first type of tabu search procedure. To calculate the acceleration coefficient for each instance, we used computational times required to find target solution objectives (the target objectives were set to the objectives of the known high quality solutions).

In the current paper, we will focus on the GES based algorithm, which in addition to the common GES structure, performs oscillations around the best record, x_{best}, based on the path-relinking (PR) methodology [20]. Throughout this paper we will refer to this method as GESPR algorithm. In our studies we will focus on the variation that always updates the best solutions when locating solutions with the same objectives.

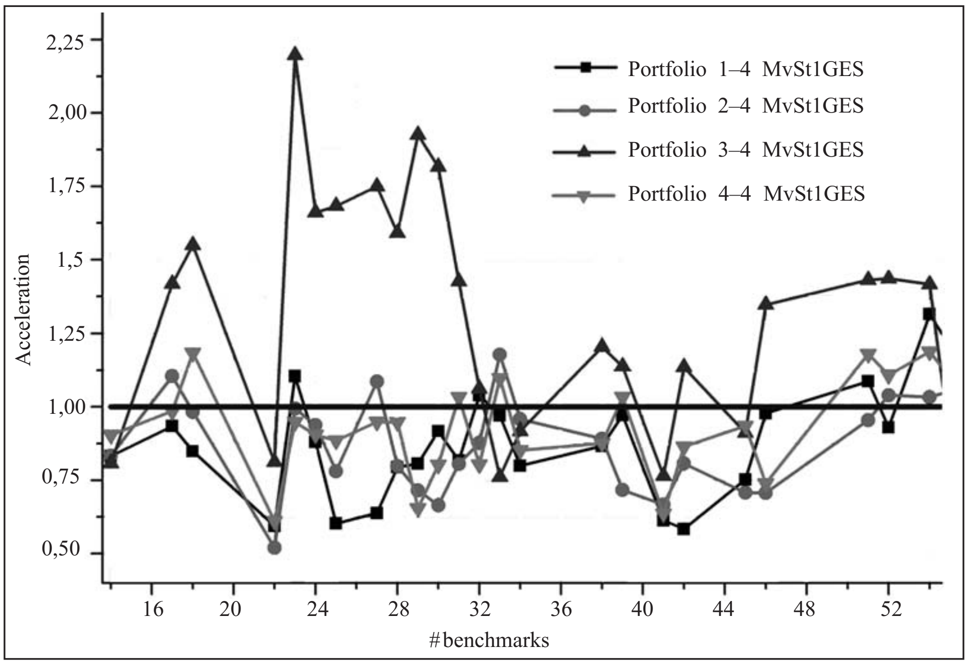


Fig. 1. Parallel acceleration achieved by the algorithm portfolios consisting of four GES algorithms

The exact notion of a team of algorithms is difficult to formalize, thus prohibiting the exact analysis. The algorithms within a team not only communicate with each other but may include a meta-algorithm that can restructure the team, modify its communication based on the outcomes of the optimization process. In order to be efficient, a team of algorithms should consist of procedures that can utilize the external information that is being communicated within the team. For example, if the problem is solved by a team of standard local search algorithms that are initiated by independently generated starting solutions, then the exchange of information is useless, unless the information that is being communicated can adjust the local search steps.

The Global Equilibrium Search Method provides excellent computational efficiency in many applications, but one of its most important qualities is its ability to be an efficient team algorithm. This is achieved through embedded memory structures that can process solutions obtained by any other algorithm. Exchanging solutions between different copies of the GES algorithm affords an opportunity to strengthen their performance by decreasing the computational time to yield a high quality solution. With respect to such exchange, the key question is: what information should be exchanged and how often should it be communicated.

The determination of a good communication protocol is not trivial. It is wrong to think that any exchange of information is useful. Some algorithms need to forget, or restrict the use of certain information. In some cases communication can even be detrimental to the team performance and should be avoided.

COMPUTATIONAL EXPERIMENT

In this section, we compare the algorithm portfolio approach to the team approach, where the algorithms communicate with each other. In both settings, we will use a set of identical GES algorithms running on different processors. In particular, we will consider an extremely simple communication strategy, where the best found solution and its objective value is the only thing that is being communicated.

For this experiment, we use the recent family of GES algorithms developed for the large WMAXCUT instances. These algorithms implement an oscillation around the best found solution based on the path-relinking methodology [20]. Like in [21], our experiments were conducted on a set of 71 benchmark instances that has been widely used to evaluate Max-Cut algorithms. These instances can be downloaded from http://www.stanford.edu/~yyye/yyye/Gset/ and include toroidal, planar and random graphs, with the number of vertices ranging from $|V| = 800$ to 20,000, and edge weights of values 1, 0, or -1.

Communication strategies. Let $x^i(t)$ denote the best solution found by the ith algorithm of the team, and be $x^*(t)$ the best overall solution that is stored in the shared memory and can be accessed by all members of the team. Thus $f(x^*(t)) = \max_{i \in A} \{f(x^i(t))\}$, where A is the set of team members. The members of team communicate with each other by reading and updating $x^*(t)$ while solving optimization problems. The following two rules define the communication protocol used by Team1:

(A) Update $x^*(t)$ whenever some algorithm A_i yields $f(x^i(t)) > f(x^*(t))$ by setting $x^*(t) = x^i(t)$.

(B) Periodically, check all algorithms A_i to see if $f(x^i(t)) < f(x^*(t))$, and if so, then redefine $x^i(t) = x^*(t)$.

In the beginning, we tested the portfolio consisting of four copies of GESPR, where none of the algorithms communicated with each other. Additionally, we considered three teams consisting of four GESPR algorithms. The algorithms in Team1 exchanged the best solutions using the file in the shared memory. The file was used to store the best objective $f(x^*(t))$ and the best current solution $x^*(t)$ obtained by the team. Whenever one of the team members improved its current best solution $x^i(t)$, it read the file and compared its best solution to the best solution from the file. If $f(x^i(t)) > f(x^*(t))$, then the file was updated with the new record value and solution $x^*(t) = x^i(t)$. Furthermore, each algorithm periodically read the record file to update its current best solution $x^i(t)$, setting $x^i(t) = x^*(t)$, whenever $f(x^i(t)) f(x^*(t))$.

The second team, Team2, used a communication scheme similar to Team1 with one distinction. From the computational results of the portfolio approach, for every trial we knew which algorithm would find the best solution. We call this algorithm a leader. The algorithms in Team2 used the following scheme in each trial. All members of the team read and write to the shared memory file similarly to the algorithms in Team1, with the exception of the leader of the trial. The trial leader only writes to the file, but never reads from it. Obviously, the results of Team2 were always at least as good as the result of the portfolio approach.

Despite of the unrealistic assumption of this scheme (knowing the leader in advance), the results of Team2 can provide some insight into what can be achieved by improving the exchange of information. Furthermore, if there is some guessing mechanism to predict the leader, then this scheme becomes implementable. For example, when solving problem G81 we've noticed that usually the algorithm that is first to find a solution x_{best} with $f(x_{\text{best}}) = 14030$, eventually becomes a leader, and this takes less than an hour. Tables 1 and 2 present the results of 20 trial runs by algorithm portfolio (where port f and port t correspond to the best found solution in a given trial and the corresponding computational time), and the results of Team1, Team2 and Team3 (where Team1_f, Team2_f, and Team3_f represent the best found objectives, while Team1_t, Team2_t, and Team3_t present the corresponding computational times). Every trial was limited to 3600 sec.

Table 1. Computational results for the instance G77

# trial	**Results of 20 trial runs**							
	Port_f	Port_t	Team1_f	Team1_t	Team2_f	Team2_t	Team3_f	Team3_t
1	9930	3216,63	9928	3293,67	**9934**	2899,35	9930	2887,94
2	9926	3504,89	9932	2426,67	9930	2727,25	9934	2506,52
3	9926	1074,06	9930	1486,31	9928	1482,94	9930	2576,57
4	9926	1245,83	9930	2827,41	9926	1211,97	9928	1910,21
5	9926	2896,41	9930	2580,06	9926	2913,74	9926	3619,86
6	9926	2974,16	9926	3761,95	9928	3163,91	9930	2885,93
7	9930	2654,45	9932	2183,11	9930	1446,33	9932	3616,34
8	9926	3576,21	9932	1999,03	9930	3580,24	9932	3305,58
9	9930	3385,39	9928	766,46	9930	3379,75	9928	1988,41
10	9926	2458,83	**9934**	2027,45	9930	2262,94	9930	2232,4
11	9930	2602,61	9930	3713,95	9930	2657,11	9930	3522,78
12	9928	3107,28	9930	1953,98	9930	2384,53	9930	3209,05
13	9926	1532,79	9928	1063,14	9928	2162,46	**9936**	3293,55
14	**9932**	2161,77	9932	2098,63	**9934**	3670,48	9930	1732,96
15	9930	2467,04	9928	2933,07	9930	2685,22	9932	2811,37
16	9926	1201,13	9930	2898,75	9926	1326,44	9928	2815,68
17	9928	3421,4	9932	2001,59	9930	1683,2	9932	3213,52
18	9930	1809,27	9928	1538,2	9932	2709,57	9932	2393,6
19	9930	3271,39	9930	3576,82	9930	2992,98	9930	3420,75
20	9928	931,82	9930	2909,12	9930	1408,25	9932	3274,53
Mean	9928	2474,67	9930	2401,97	9929,6	2437,433	9930,6	2860,88

The computational experiment was conducted using the PC Intel CoreTM i7-3770 CPU @ 3.40 GHz 3.40 GHz and 8.0GB RAM in real time. In other words, all algorithms were running at the same time.

Computationally, the GESPR algorithm reveals strong intensification properties, but less satisfactory diversification behavior. The algorithms can be trapped in a basin of attraction populated by “bad” solutions leading to unnecessary computations (such attractors will be called traps). The communication between GESPR algorithms in turn can lead to an entrapment of the whole team. Therefore, it is necessary to introduce special measures to prevent such situations and to enable escape from such traps.

The analysis of the computational experiments leads to the conclusion that the optimization process is similar to a chess game. It also can be divided into three phases: a search over poor solutions, a search over average solutions and the search for good solutions, which in chess corresponds (roughly) to the opening game, middle game and end game. In the first phase, when there is no danger of falling into the trap of poor quality solutions, we can promote an active exchange of information between the team members. Such exchange will help to transition rapidly into the second phase, where there is a higher probability of getting trapped. To avoid this and to provide better diversification, on the second stage we reduce the exchange of information. Finally, in the third phase we again promote frequent communication to enable intensification of the search for high quality solutions.

An attempt to incorporate the foregoing entrapment considerations was implemented in the team of algorithms Team3. The algorithm GESPR quickly moves from poor solutions to good solutions entering the middle game stage. Therefore, the time interval $[0, 3600]$ was divided according to the golden ratio. The smaller part of it (1368 sec) was allocated to the middle game stage, where we prohibited any communication with one exception: the exchange is allowed only if the solution significantly improves the best record: $f(x^*(t)) \geq f(x^i_{\text{best}}(t)) + 10$. In the third stage (time interval $[1368, 3600]$, the communication patterns followed the same rules as in Team1. Since the algorithms check for the stopping criterion only periodically, some trials report the time to find the best solution that is larger than 3600 sec. In the last row, we present average values for the data in each column.

Considering these computational results, it is necessary to point out the high computational efficiency of the GESPR algorithm: the portfolio and teams of GESPR algorithms found solutions that are better than previously known records (9926 for problem G77 and 14030 for problem G81) [21]. More importantly, all 3 versions of team algorithms outperformed the portfolio approach in terms of solution quality. As mentioned earlier, in the worst case Team2 would show same results as the portfolio approach. However, if we look at trials 4, 9, 11, 15, and 16 when solving G77 and trials 11, 12, 16, and 19 when solving G81, Team2 improved the best found objective values. In some trials, when Team2 found the same solution as the portfolio, the reported computational times are different, since all the processors belong to the same computational node and might interfere with each other. Because of this interference the results of the team make it difficult to replicate the trials by fixing the seeds of random number generator.

In our experiments, the teams consisted of four algorithms running in parallel. In such a setup, Team2 discarded 25 % of the solutions it generated, whenever the best solution of the leader was improved by the rest of the team. With more algorithms, the disposal of the leader would have a much smaller impact on the performance.

The results obtained by Team1 and Team3 encourage the development of team algorithms. They were not only better compared to the no communication approach, but also surpassed the results of Team2 on G77.

Table 2. Computational results for the instance G81

# trial	Results of 20 trial runs							
	Port_f	Port_t	Team1_f	Team1_t	Team2_f	Team2_t	Team3_f	Team3_t
1	14038	3251,84	14042	3103,72	14038	2178,55	14040	3432,88
2	14030	3480,69	14032	3373,46	14040	4712,58	14032	3438,01
3	14030	2131,45	14036	3528,37	14036	2728,44	14036	2456,63
4	14038	2267,64	14044	2324,39	14044	2224,24	14038	1948,64
5	14034	3535,53	14038	3724,05	14038	3608,4	14040	3698,1
6	14036	3841,05	14036	2910,7	14036	2753,26	14038	3593,83
7	14038	3509,7	14040	4220,32	14038	2338,49	14038	3148,18
8	**14044**	2905,24	14044	3303,9	14044	2612,46	**14046**	3323,57
9	14030	1371,69	14038	3548,84	14038	3605,76	14036	3149,75
10	14036	3942,49	14042	3130,61	14040	2350,18	14038	2606,26
11	14038	2511,59	14034	2382,79	14038	2569,45	14040	3774,32
12	14038	3313,16	14036	4074,08	14038	3336,2	14034	2483,22
13	14034	3441,66	14034	2286,94	14036	2742,39	14034	3352,15
14	14038	3185,49	14040	2834,76	14038	2679,3	14040	2885,26
15	14036	2732,64	14042	3562,49	14040	2688,35	14040	3141,32
16	14040	3413,63	14034	2911,64	14040	2968,99	14038	3517,64
17	14038	3441,18	14036	3462,01	14040	4119,84	14040	3054,89
18	14038	3358,61	14040	2127,68	14044	2697,95	14044	2777,57
19	**14044**	2885,46	14042	3208,98	14044	3021,08	14038	2673,94
20	14032	3117,96	**14046**	3082,6	**14046**	3988,78	14044	2986,29
Mean	14036.5	3081,935	14038.8	3155,117	14039.8	2996,24	14038,7	3072,123

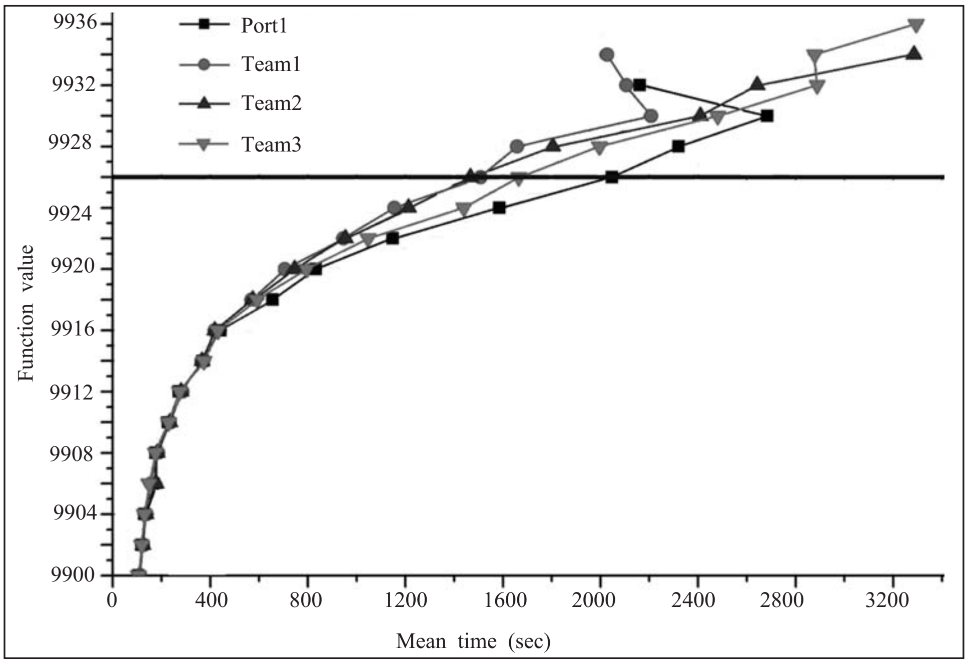


Fig. 2. Average dynamics of the solving process of the benchmark G77

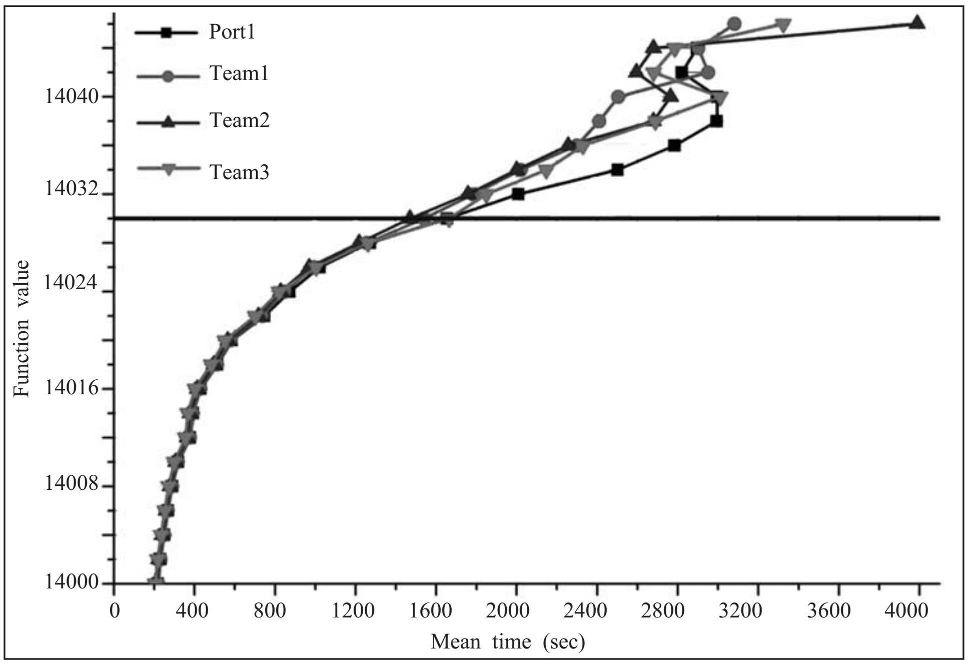


Fig. 3. Average dynamics of the solving process of the benchmark G81

Figures 2 and 3 show the average performance when solving G77 and G81, respectively. Each point on these graphs shows the average computational time until finding a solution with the given objective value or better. The bold horizontal lines show the previously known records [21]. These results show an impressive boost in performance when introducing communication to the portfolio approach.

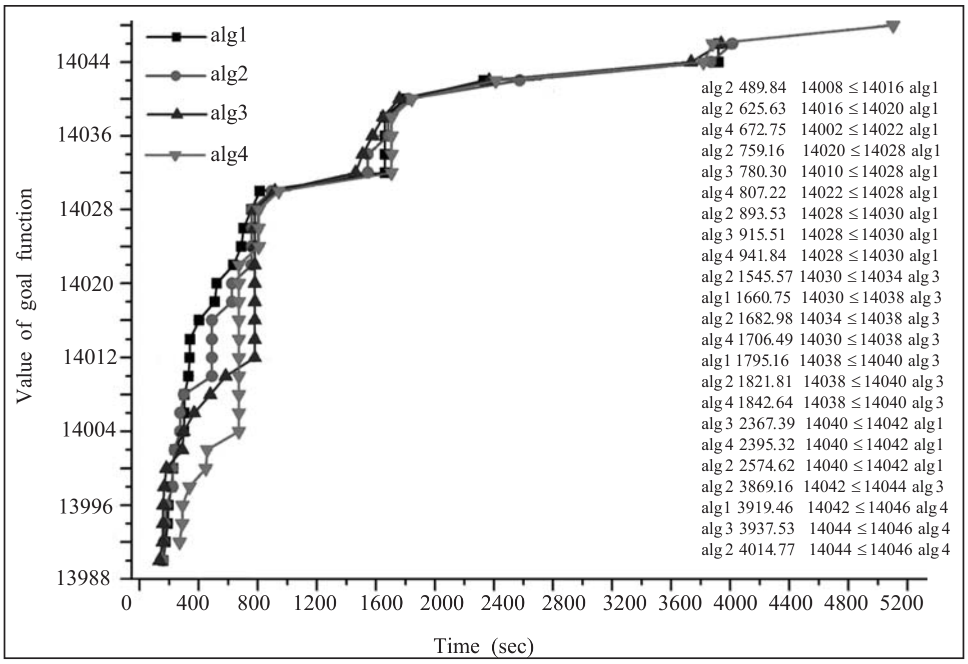
While in the current paper we focused on comparing different communication strategies, our experiments have resulted in solutions that improve the current best known

T a b l e 3. Computational results for instances G55–G81

Name task	$\|V\|$	BLS	GESPR
G55	5000	10294	**10299**
G56	5000	4012	**4017**
G57	5000	3492	**3494**
G58	5000	19263	**19293**
G59	5000	6078	**6086**
G60	7000	14176	**14188**
G61	7000	5789	**5796**
G62	7000	4868	**4870**
G63	7000	26997	**27045**
G64	7000	8735	**8751**
G65	8000	5558	**5562**
G66	9000	6360	**6364**
G67	10000	6940	**6950**
G70	10000	9541	**9591**
G72	10000	6998	**7006**
G77	14000	9926	**9938**
G81	20000	14030	**14048**

record for G55–G81 benchmark instances, which are summarized in Table 3. In addition to the new records found by GESPR, we also present the best solutions found by the Breakout Local Search (BLS) approach [21] which provides excellent computational performance on the set of standard benchmarks compared to other approaches in the literature. The first and second columns in Table 3 provide problem names and the number of vertexes in the corresponding graphs, while the third and fourth columns provide the best solution out of 20 trials for the BLS. The interacting algorithms GESPR found new records for all of the instances in Table 3, suggesting that this form of interaction is a good choice for studying the potential of the team approach to algorithm design.

Figure 4 shows a run of Team1 that resulted in the record (14048) for the problem G81. This trial is not included in Table 2, since the record solution was found after 3600 sec. In addition, we show a full protocol of information exchange between algorithms in Team1.



```
alg 2 489.84   14008 ≤ 14016 alg1
alg 2 625.63   14016 ≤ 14020 alg1
alg 4 672.75   14002 ≤ 14022 alg1
alg 2 759.16   14020 ≤ 14028 alg1
alg 3 780.30   14010 ≤ 14028 alg1
alg 4 807.22   14022 ≤ 14028 alg1
alg 2 893.53   14028 ≤ 14030 alg1
alg 3 915.51   14028 ≤ 14030 alg1
alg 4 941.84   14028 ≤ 14030 alg1
alg 2 1545.57  14030 ≤ 14034 alg 3
alg1 1660.75   14030 ≤ 14038 alg 3
alg 2 1682.98  14034 ≤ 14038 alg 3
alg 4 1706.49  14030 ≤ 14038 alg 3
alg1 1795.16   14038 ≤ 14040 alg 3
alg 2 1821.81  14038 ≤ 14040 alg 3
alg 4 1842.64  14038 ≤ 14040 alg 3
alg 3 2367.39  14040 ≤ 14042 alg1
alg 4 2395.32  14040 ≤ 14042 alg1
alg 2 2574.62  14040 ≤ 14042 alg1
alg 2 3869.16  14042 ≤ 14044 alg 3
alg1 3919.46   14042 ≤ 14046 alg 4
alg 3 3937.53  14044 ≤ 14046 alg 4
alg 2 4014.77  14044 ≤ 14046 alg 4
```

Fig. 4. Trial of Team1 that resulted in the record for the instance G81

CONCLUSIONS AND FUTURE TRENDS

The results suggest that the communication between algorithms running in parallel is a promising research direction. Our algorithms produced new best solutions for the classical benchmark problems from G55 to G81, and in just 1 hour, the teams of algorithms were able to obtain solutions whose quality established new records for the large scale instances G77 and G81 (14000 and 20000 vertices, respectively). In addition to improving the algorithms in the team, future research can beneficially make use of large scale computing systems to address communication patterns, communication protocols, content of information exchange, and communication management. The results in this paper suggest that in the near future we will be able to solve WMAXCUT problems with up to 50,000 vertices through the capabilities offered by parallel computing — a prospect which seemed impossible in the recent past.

REFERENCES

1. V. P. Shylo and O. V. Shylo, "Solving the maxcut problem by the global equilibrium search," Cybernetics and Systems Analysis, **46**, No. 5, 744–754 (2010).
2. I. V. Sergienko and V. P. Shylo, Discrete Optimization Problems: Challenges, Solution Techniques, and Analysis [in Russian], Naukova Dumka, Kiev (2003).
3. V. P. Shylo, "The method of global equilibrium search," Cybernetics and Systems Analysis, **35**, No. 1, 68–74 (1999).
4. V. P. Shylo, O. V. Shylo, and V. A. Roschyn, "Solving weighted max-cut problem by global equilibrium search," Cybernetics and Systems Analysis, **48**, No. 4, 563–567 (2012).
5. V. P. Shylo and O. V. Shylo, "Path relinking scheme for the Max-Cut Problem within global equilibrium search," International J. of Swarm Intelligence Research (IJSIR), **2**, No. 2, 42–51 (2011).
6. F. Barahona, M. Grotschel, M. Juger, and G. Reinelt, "An application of combinatorial optimization to statistical physics and circuit layout design," Operations Research, **36**, 493–513 (1988).
7. C. Helmberg and F. Rendl, "A spectral bundle method for semidefinite programming," SIAM J. on Optimization, **10**, No. 3, 673–696 (2000).
8. R. Karp, "Reducibility among combinatorial problems" in: R. Miller and J. Thatcher (eds.), Complexity of Computer Computations, Plenum Press (1972), pp. 85–103.
9. F. Hadlock, "Finding a maximum cut of a planar graph in polynomial time," SIAM J. Comput., **4**, No. 3, 221–225 (1975).
10. K. Krishnan and J. E. Mitchell, "A semidefinite programming based polyhedral cut and price approach for the maxcut problem," Comput. Optim. Appl., **33**, No. 1, 51–71 (2006).
11. M. X. Goemans and D. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," Journal of the ACM, **42**, No. 6, 1115–1145 (1995).
12. R. Marti, A. Duarte, and M. Laguna, "Advanced scatter search for the max-cut problem," INFORMS J. on Computing, **21**, 26–38 (2009).
13. S. Burer, R. D. C. Monteiro, and Y. Zhang, "Rank-two relaxation heuristics for max-cut and other binary quadratic programs," SIAM J. on Optimization, **12**, 503–521 (2002).
14. P. Festa, P. M. Pardalos, M. G. C. Resende, and C. C. Ribeiro, "Randomized heuristics for the maxcut problem," Optimization Methods and Software, **7**, 1033–1058 (2002).
15. G. Palubeckis and V. Krivickiene, "Application of multistart tabu search to the max-cut problem," Information Technology and Control, **31**, No. 2, 29–35 (2004).
16. S. Kahruman, E. Kolotoglu, S. Butenko, and I. V. Hicks, "On greedy construction heuristics for the max-cut problem.," Int. J. Comput. Sci. Eng., **3**, No. 3, 211–218 (2007).
17. O. V. Shylo, O. A. Prokopyev, and J. Rajgopal, "On algorithm portfolios and restart strategies," Operations Research Letters. **39**, No.1, 49–52 (2011).
18. O. Mostovyi, O. A. Prokopyev, and O. V. Shylo, "On maximum speedup ratio of restart algorithm portfolios," INFORMS Journal on Computing, **25**, No. 2, 222–229 (2013).
19. V. P. Shylo, V. O. Roschyn, and P. V. Shylo, "Construction of algorithm portfolio for parallelization the solving process of WMAXCUT problem," in: Computer Mathematics [in Russian], No. 2, 163–170, V. M. Glushkov Inst. Cybern., NAS Ukraine, Kiev (2014).
20. F. Glover, M. Laguna, and R. Marti, "Fundamentals of scatter search and path relinking," Control and Cybernetics, **39**, 653–684 (2000).
21. U. Benlic and J. K. Hao, "Breakout local search for the max-cut problem," J. Engineering Applications of Artificial Intelligence, **26**, No. 3, 1162–1173 (2013).

Поступила 04.09.2014