An Enhanced Evolutionary Approach for Solving the Nodes Migration Scheduling Problem

Fatma Moalla[1(✉)], Ali Balma[2], and Mehdi Mrad[3]

[1] ISG Tunis, University of Tunis, Tunis, Tunisia
fatma.moalla.fen@gmail.com
[2] National Higher Engineering School of Tunis, University of Tunis, Tunis, Tunisia
alibalma05@yahoo.fr
[3] Department of Industrial Engineering, College of Engineering,
King Saud University, Riyadh, Saudi Arabia
mmrad@ksu.edu.sa

Abstract. This paper deals with a scheduling problem in the telecommunication field, namely the node migration scheduling for an access network. The problem consists of migrating nodes from a former network to a new one affording the required services. The migration procedure needs the installation of a bridge between the two networks without disrupting current services. Nodes are moved sequentially one by one. Our objective is to minimize the cost of the required bridge. We describe an enhanced genetic algorithm based on a good initial population. Numerical experiments show that our method has good performance.

1 Introduction

In the telecommunications filed, operators are faced to new challenges such as emergence of new services and the traffic evolution. Indeed, the shift from a service to another obliges the operator to adjust the actual network configuration to support the new equipment installation. Technological capabilities are in perpetual evolution. Formerly, circuit switching was well repented before the emergence of internet and video technology. Nowadays, The mobile technology become the trend in the telecommunication sector. The operators interest is on deploying the 5G technology. Therefore, all these changes emphasizes the importance of the migration optimization strategies and the more specifically the migration scheduling problem. A migration decision can be taken at the strategic or the tactical level. In fact, For a medium term horizon links and nodes may be concerned. In a medium-term horizon, only nodes should migrate from a former network to a new one.

In this context, we consider the node migration process as a gradual stepwise move of the nodes which should be well planified in order to prevent the services interruption. For this aim, a temporary bridge should be installed between two networks linking the migrated nodes to those remaining in the ancient network

N. Boudriga et al. (Eds.): UNet 2018, LNCS 11277, pp. 69–81, 2018.
https://doi.org/10.1007/978-3-030-02849-7_7

(Fig. 1). Once all nodes are displaced to the new network, the bridge should be removed.

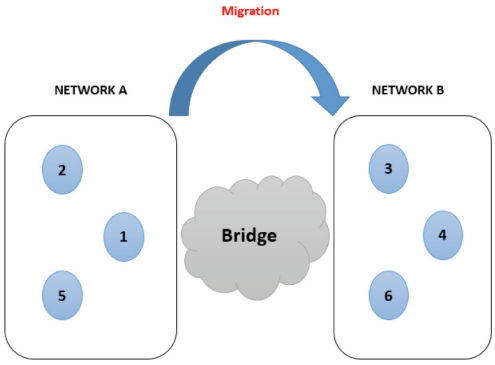

Fig. 1. A migration from ancient to new network

We extend the node migration scheduling problem addressed in [1] by proposing an enhanced genetic algorithm with a new strategy for generating the initial population. The paper is organized as follows. In Sect. 2, we present the migration process. Section 3 recall the problem formulation as presented in [1]. Section 4 presents some related works. Section 4 details the proposed method. In Sect. 5, we present computational results and finally we conclude.

2 Problem Description

Geographically spoken, three domains can be defined for the telecommunication networks which are the core, the metro and the access network. The access network which is connected to the end users consists of a set of transmission nodes connected to traffic processing nodes (Fig. 2). Two transmission nodes connect a processing nodes ensuring the continuity of services. In order to perform a migration, transmission nodes should be replaced by other nodes supporting the bandwidth growth.

As mentioned previously in the introduction, installing a bridge along the migration process present many advantages. First, it guarantees the interoperability between the networks. Second, it prevents also from the service rupture. From a technical perspective, it represents a gateway for the traffic between the nodes connected to the ancient and the new network. Our objective is to find minimal capacity cost connecting the two networks. Finding a good node ordering for a migration sequence is the success key for achieving our goal. More practical applications for the migration problem include the Virtual machine migration as well as evolving to a more capacitated network using IP routers. The following example highlights the influence of changing the nodes order in

the migration sequence by considering two different orders. We note that the maximum value of the traffic on the bridge throughout the migration process is 15 for the first order. We now consider a second migration scenario. For this scenario, the value of the maximum traffic on the bridge is 12, entailing a capacity of the bridge less than the first scenario (Fig. 3).

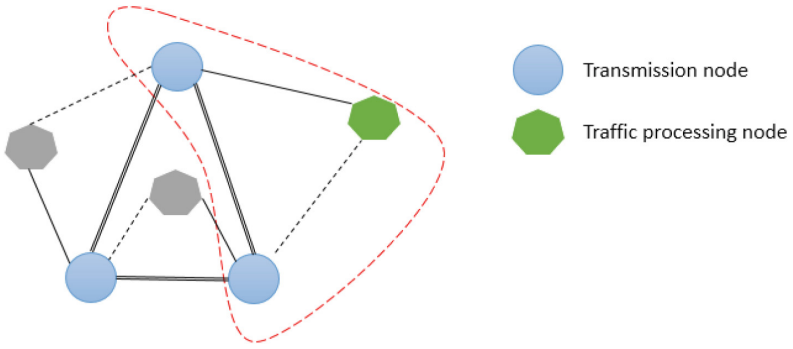

Fig. 2. Access network structure

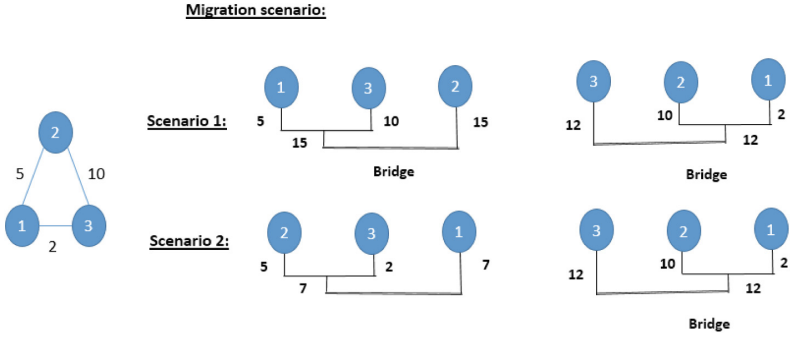

Fig. 3. Migration scenarios

3 Problem Formulation

Given an undirected graph G = (V, E) with $|v[$ = n nodes and E the set of valuated edges. An edge corresponding to the traffic amount between two nodes i and j and is denoted d_{ij}. The capacity on the bridge is ensured via boards. A board w which belongs to the boards set W is defined by its modular capacity λ_w and its cost k_w. We denote by $O_i = \sum_{j=1}^{n} d_{ij} \quad \forall i$ the total flow issued from the node i. The decision variables are:

- x_{ik}: binary variables which take 1 if node i is migrated before or at the stage k and 0 otherwise.
- f_{ik}: the traffic amount on the bridge emanating from i at the stage k.
- c_w: the number of boards of type w.

We recall the problem formulation as presented in [1]:

$$Min \sum_{w \in W} k_w c_w \tag{1}$$

$$f_{ik} \geq O_i x_{ik} - \sum_{j=1}^{n} d_{ij} x_{jk} \qquad \forall k = 1, \ldots, n \forall i = 1, \ldots, n \tag{2}$$

$$\sum_{i=1}^{n} f_{ik} \leq \sum_{w \in W} \lambda_w c_w \qquad \forall k = 1, \ldots, n \tag{3}$$

$$\sum_{i=1}^{n} x_{ik} = k \qquad \forall k = 1, \ldots, n \tag{4}$$

$$x_{in} = 1 \qquad \forall i = 1, \ldots, n \tag{5}$$

$$x_{ik-1} \leq x_{ik} \qquad \forall i = 1, \ldots, n \forall k = 2, \ldots, n \tag{6}$$

$$x_{ik} \in \{0, 1\} \qquad \forall i, k = 1, \ldots, n \tag{7}$$

$$f_{ik} \geq 0 \qquad \forall i, k = 1, \ldots, n \tag{8}$$

$$c_w \in Z \qquad \forall w \in W \tag{9}$$

The objective function (1) aims to minimize the total boards cost. The constraint (2) expresses the flow amount circulating on the bridge if we migrate the node i at or before the stage k. The constraint (3) imposes that the total flow is limited by the capacity of the bridge at any stage. The constraint (4) forces k nodes to be moved in the new network after k steps. The constraint (5) ensure that all nodes should be migrated the final stage. The constraint (6) fixes the variable at 1 for stages superior to k, once we migrate a node i at a given stage k. The constraints (7), (8) and (9) are the integrity and non-negativity constraints.

4 Related Work

The network planning problem has attracted more attention in the telecommunications field. One of the most studied problems in the planning context is the migration problem. This problem encompasses a variety of network technologies such as traffic routing and transmission technology [3] and mobile access network technology [4]. The authors of [5] and [6] tackle the migration problem from an SDH to Ethernet network. The proposed migration consider both nodes and links. The Problem is solved using metaheuristics namely the ant colony [5] and a genetic algorithm in [6]. The authors of [7] suggests a mixed integer

linear program to reduce the costs in a passive Optical Network. These costs are related to the capacity evolution. An empirical approach and mathematical model were proposed in [8] to address the migration in a mobile network with load balancing. The authors of [9] propose two algorithms for a virtual network in order to minimize the whole migration time. The authors of [10] take into account the virtual machines dependencies and the network structure to get a scheme. The objective of such migration is to decrease the traffic amount in the network. We study in this work the node migration scheduling problem for an access network in order to migrate from a 4G network to 5G network. In [1], a time-staged formulation and lower bound model were proposed to solve this problem. We should mention that exact method were not able to solve large sized instances. An approximate method was developed in [2] to solve this problem in virtual networks. Therefore, the objective of this work is to provide another approximate method combining the partition problem concepts and the standard elements of a genetic algorithm.

5 Proposed Method

Genetic algorithms are population based search techniques. They are among the most important class of evolutionary algorithms. The concept behind this technique introduced by Holland [11] is inspired by biological natural selection. This work is motivated by the fact that the use of the classical form of genetic algorithm (GA) may consume much time and could not converge to a global optimum. To remedy this weaknesses, many works focus on improvement on GA operators such as crossover and mutation. Other works consider that evolutionary algorithms with automatic parameter tuning is more performant than setting manually these parameters. Another important feature of the genetic algorithm is the population initialization since it accelerates the convergence of the algorithm. Population initialization has not received the attention it deserves despite its importance. In order to solve node migration scheduling problem, we propose a new genetic algorithm with an initial population based on problem knowledge.

Different approaches for generating initial population are suggested. An alternative random initialization in genetic algorithm was discussed by [12], authors of [13] proposed an opposition based learning initialization approach. Selective initialization is proposed in [14] to deliver better results. A state-of the-art population initialization techniques was proposed in [15]. The classification of these techniques is based on three categories which are randomness, compositionality and generality.

We propose a knowledge based procedure for our GA. The procedure starts by generating the first chromosome C1 in the initial population. This chromosome represents a bi-partition solution of the problem.

The pseudo-code of the overall proposed genetic algorithm is given in Algorithm 1.

We introduce quickly the bi-partitioning problem in Sect. 5.1 related to the graph theory and optimization problems.

Algorithm 1. Pseudo code of the genetic approach

```
Input: GA parameters:
Output: cost of the best migration order;
Begin
generate the initial population with knowledge based initialization procedure
While termination criterion is not met do
    select parents from the current population pop_k
    apply crossover
    apply mutation
    apply replacement
EndWhile
    compute the cost of the best migration order
End
```

5.1 The Graph-Partitioning Problem GPP

The graph-partitioning problem (GPP) is a combinatorial optimization problem, which belongs to the class of NP-hard. Cutting a graph into smaller parts is relevant since it presents many advantages for parallelization or complexity reduction. This problem concerns both the weighted and unweighted graph. Formally, Given an undirected graph G = (V, E), where V represents the set of vertices and E refers the set of edges, a graph partitioning could be defined as division of V into k disjoint subsets $V_1, V_2, \ldots, V_k$ in order to minimize the cut value. Each disjoint subset is named a partition. The cut edges are those edges whose extremities belong to different partitions. For a weighted graph, the cut value C_i defines the cut edges weight sum. Considering k = 2 is a special case of partitioning problem where we divide the vertices set in only two subsets. This problem is called graph bi-partitioning or graph bisection. The GPP problem has proven its ability to solve real life such parallel processing, complex network, image processing, road network and VLSI design [16]. Much effort, as for [17], have been made to solve this problem exactly. When exact methods fail to deliver results, approximative methods are used such as lin Kernighan algorithm and more recently multilevel algorithms [16]. A fast running heuristic for the bi-partitionning problem from [18] is used to generate the chromosome C1.

5.2 Solution Representation

The sequence representation can be considered as permutation which refers to a sequence of integers where each number designates a migration order. Possible values are from 1 to n where n = $|N|$. The vector Perm(x) is the permutation order of a solution x as shown in Fig. 4.

5.3 Initial Population

A population is composed of a set of chromosomes as illustrated in Fig. 5. NP denotes the population size. As mentioned earlier, in order to generate the first

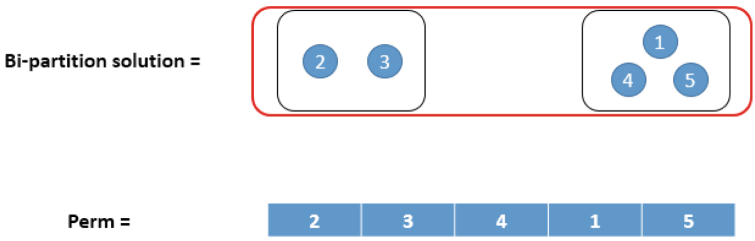

Fig. 4. A chromosome encoding

individual, we refer to the heuristic from [18]. The remaining population individuals ($|NP| - 1$) are generated following the Algorithm 2.

Algorithm 2. Knowledge-based algorithm for generating the initial population

1: **Input**: NP: population size, N: chromosome length
2: **Output**: initial population pop1;
3: **Begin**
4: solution ⟵ RunHeuristic (from [18])
5: generate the first chromosome C1 in the initial population
6: For all remaining individuals in this population
7: swap two nodes position x and y to get the current individual Ck (with $x, y \in [1, \frac{N}{2}]$ or $x, y \in [\frac{N}{2} + 1, N]$)
8: EndFor
9: **End**

5.4 Selection

We use the classical roulette selection wheel. In this method, all the chromosomes in the population are placed on the roulette wheel according to their fitness values. Each individual is assigned a segment of roulette wheel. The size of each segment in the roulette wheel is proportional to the value of the fitness of the individual.

5.5 Crossover

One-point crossover works by selecting a common crossover point in the parent chromosome and then swapping the corresponding parts. We check the offspring chromosome feasibility in order to get a feasible potential solution.

5.6 Mutation

The purpose of mutation is to preserve and introduce diversity. It alters one or more gene values in a chromosome from its initial state.

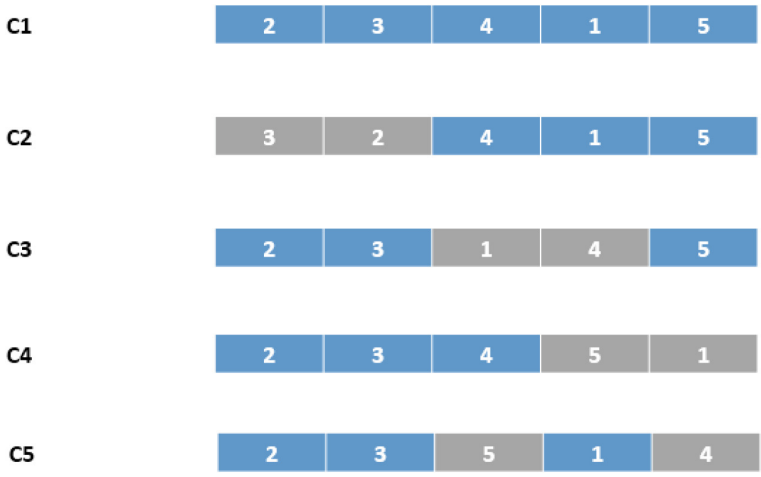

Fig. 5. Generating initial population

5.7 Replacement

The replacement operator consists of choosing the appropriate individual for removal. Replacement strategies have been studied in the literature. Several methods were discussed among them Steady State and elitism.

6 Experimental Tests

The proposed approach has been coded in C language and executed on a i7 processor machine with 1.8 GHz. Memory and 8 GB Ram. In this section, two benchmark sets are considered. The first set A refer to small size instances up to 40 nodes proposed firstly in [1]. The second set B deals with medium and large size instances from [2]. For all the instances a complete graph is considered. The overall Instances sets consist of a traffic matrix randomly generated between 100 Mb/s and 1 Gb/s which consider only the symmetric traffic.

Different versions of the genetic algorithm are considered:

1. SGA: simple genetic algorithm (random initial population).
2. ICPGA: genetic algorithm with combined initial population($\frac{1}{2}$ random, $\frac{1}{2}$ based bi-partitioning approach).
3. IPBBGA: genetic algorithm with initial population based bi-partitioning.

Defining the quality of this proposed evolutionary approach is crucial. In order to evaluate the results, we use the following performance measures which are the gap and ARPD metrics:

$$G = 100 * \frac{(GA_i - OPT)}{OPT}. \tag{10}$$

$$ARPD = \frac{1}{s}\sum_{i=1}^{s}\frac{A_i - bA_i}{bA_i} * 100 \quad (11)$$

where GA_i denotes the solution value obtained by a version of genetic algorithm for its i^{th} instance. bGA_i is the best solution value obtained for that instance among the three versions of GA (SGA, ICPGA and IPBBGA).A_i refer the solution value obtained by an approximative method (heuristic, genetic algorithm) for its i^{th} instance. bA_i is the best solution value obtained for that instance among the fours methds of GA (SGA, ICPGA and IPBBGA and the heuristic). We denote by s the number of problem instances for a problem size. We should note that OPT is the solution of mathematical formulation f2 presented in [1].

Table 1. Average gap values for instances of set A

Size	G_{IPBBGA}	G_{SGA}
20	5.71	17.14
25	2.77	11.11
30	0	1.85
35	0	6.94
40	0	10.75

Table 2. Average ARPD values for instances of set B

Size	$ARPD_{IPBBGA}$	$ARPD_{ICPGA}$	$ARPD_{SGA}$	$ARPD_H$
50	0	0	7.73	2.77
60	0	0	2.62	0.76
70	0	0	5.99	0.28
80	0	0	4.78	0.21
90	0.16	0	5.89	2.2
100	0	0.15	5.5	2.03
200	0.04	0.06	4.34	1.17

The first column of Table 1 cites instances of type A. Column 2 and 3 report the average gap values between optimal solution and two versions of genetic algorithm namely the IPBBGA and the SGA. It is clear that the proposed approach outperforms the classical version of genetic algorithm. Indeed, the average gap value has decreased from 17.14 to 5.71 for $|v| = 20$ and from 11.11 to 2.77 for $|v| = 25$. The optimality is reached for all the instances of size 30, 35, 40. Mathematical formulation from [1] was not able to deliver results for instances

Table 3. Comparaison between initial population results

Size	IPBBGA		SGA	
	Min	Max	Min	Max
20	491.7	502.6	519.9	580.6
25	758.5	783.2	821.3	901.1
30	1098	1121.8	1187.3	1284
35	1515.4	1546.7	1636.5	1733
40	1992.5	2018.2	2153.7	2256
50	3191.1	3194.8	3414	3518.7
60	4577.9	4595.4	4868	5006.6
70	6229.1	6244.9	6660.7	6839.2
80	8176.5	8194.6	8730.2	8894.5
90	10404	10425.7	11059.8	11288.7
100	12849	12863.2	13642	13858.4
200	52312.8	52344.8	54698.4	54981.7

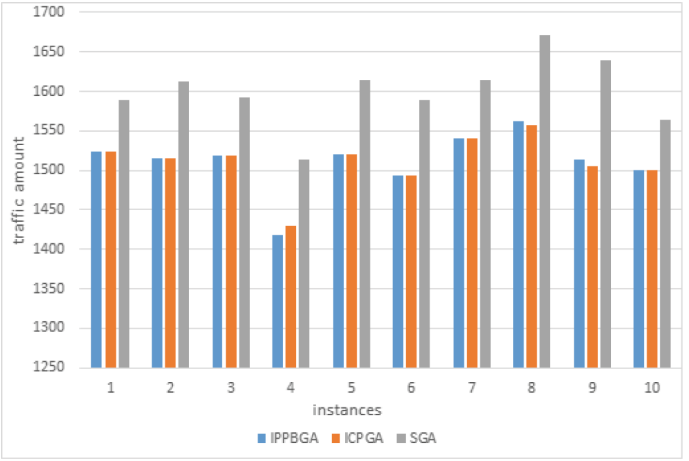

Fig. 6. Total traffic on the bridge for 10 instances ($|V| = 35$ nodes)

up to $|v| = 40$ nodes. Therefore, to assess the efficiency of the IPBBGA algorithm for medium and large size instances, we compare our results to Heuristic H from [2] which address the nodes migration problem for virtual machines. Table 2 presents the ARPD values for the instances set B. Results show that for the two versions using the initialization procedure (IPBBGA and ICPGA) perform better compared to basic version of genetic algorithm (SGA) and the heuristic H. Indeed, the proposed results in column 2 and 3 have small average

Table 4. Running time for IPBBGA

SEQ	t_{IPBBGA}
20	0.05
25	0.08
30	0.12
35	0.18
40	0.28
50	0.5
60	0.89
70	1.3
80	1.75
90	2.4
100	3.2
200	24.1

ARPD values near to zero for almost the cases. The SGA has the worst results among the considered methods i.e. an average ARPD from 2.62 to 7.73.

In order to confirm the robustness of the initialization procedure, average Best (Min) and worst (Max) chromosomes for the initial population are listed in Table 3 for both IPBBGA and SGA. The reported values disclose the competitivity of our initial population since for the overall considered sizes, the average worst chromosomes fitness for IPBBGA is always better to the average best chromosomes fitness for the SGA. The running time is reported in Table 4. For similar cost values, we should compare the total traffic values. Figure 6 reports the total traffic values for 10 instances of size $|V| = 35$ nodes. Based on these results, the initialization procedure proves its efficiency to deliver better results.

7 Conclusion

In this paper, the node migration scheduling problem for an access network is discussed. A genetic algorithm with initialization procedure is proposed. The originality of this proposed approach resides in using initial population based on a fast heuristic of the bi-partition problem. This problem was used in prior works to get effective bounding for our migration scheduling problem. The results show that our proposed approach is effective. It provides better results than other methods in a reasonable time. To extend this work, we will investigate more tight bounds for this problem as well as others application context.

References

1. Mrad, M., Balma, A., Moalla, F., Ladhari, T.: Nodes migration scheduling of access networks. IEEE Trans. Netw. Serv. Manag. **14**(1), 77–90 (2017)
2. Moalla, F., Balma, A., Mrad, M.: A rapid heuristic for the virtual machines migration scheduling problem. In: Proceeding of Engineering and Technology PET of The Fifth International Conference on Control and Signal Processing (CSP 2017), vol. 25, pp. 44–47 (2017)
3. Leiva, A., Machuca, C.M., Beghelli, A., Olivares, R.: Migration cost analysis for upgrading WDM networks. IEEE Commun. Mag. **51**(11), 87–93 (2013)
4. Haidine, A., Aqqal, A., Ouahmane, H.: Modeling the migration of mobile networks towards 4G and beyond. In: El Oualkadi, A., Choubani, F., El Moussati, A. (eds.) Proceedings of the Mediterranean Conference on Information Communication Technologies 2015, vol. 380, pp. 355–363. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30301-7_37
5. Türk, S., Radeke, R., Lehnert, R.: Network migration using ant colony optimization. In: 2010 9th Conference on Telecommunications Internet and Media Techno Economics (CTTE), pp. 1–6, June 2010
6. Türk, S., Radeke, R., Lehnert, R.: Improving network migration optimization utilizing memetic algorithms. In: 2013 Global Information Infrastructure Symposium, pp. 1–8, October 2013
7. Andrade, M.D., Tornatore, M., Sallent, S., Mukherjee, B.: Optimizing the migration to future-generation passive optical networks (PON). IEEE Syst. J. **4**(4), 413423 (2010)
8. Chardy, M., Yahia, M.B., Bao, Y.: 3G/4G load-balancing optimization for mobile network planning. In: 2016 17th International Conference on Telecommunications Network Strategy and Planning Symposium (Networks), September 2016
9. Ammar, M., Lo, S., Zegura, E.: Design and analysis of schedules for virtual network migration. In: 2013 IFIP Networking Conference, pp. 1–9 (2013)
10. Shrivastava, V., Zerfos, P., Lee, K.-W., Jamjoom, H., Liu, Y.-H., Banerjee, S.: Application aware virtual machine migration in data centers. In: 2011 Proceedings of INFOCOM, pp. 66–70 (2011)
11. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge (1992)
12. Kallel, L., Schoenauer, M.: Alternative random initialization in genetic algorithms, In Bäeck, Th. (ed.) Proceedings of the 7th International Conference on Genetic Algorithms, pp. 268–275. Morgan Kaufmann, Burlington (1997)
13. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.: A novel population initialization method for accelerating evolutionary algorithms. Comput. Math. Appl. **53**(10), 1605–1614 (2007)
14. Sivaraj, R., Ravichandran, T., Priya, R.D.: Boosting performance of genetic algorithm through selective intialization. Eur. J. Sci. Res. **68**(1), 93–100 (2012). ISSN 1450–216x
15. Kazimipour, B., Li, X., Qin, A.K.: A review of population initialization techniques for evolutionary algorithms. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 2585–2592. IEEE (2014)
16. Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., Schulz, C.: Recent advances in graph partitioning. In: Kliemann, L., Sanders, P. (eds.) Algorithm Engineering. LNCS, vol. 9220, pp. 117–158. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49487-6_4

17. Liberti, L.: Compact linearization for binary quadratic problems. 4OR **5**(3), 231–245 (2007)
18. Wu, J., Jiang, G., Zheng, L., Zhou, S.: Algorithms for balanced graph bi-partitioning. In: 2014 IEEE International Conference on High Performance Computing and Communications, 2014 IEEE 6th International Symposium on Cyberspace Safety and Security, 2014 IEEE 11th International Conference on Embedded Software and Systems (HPCC, CSS, ICESS), pp. 185–188. IEEE (2014)