Heterogeneous Multi-agent Evolutionary System for Solving Parametric Interval Linear Systems

Jerzy Duda and Iwona Skalna

AGH University of Science and Technology, Krakow, Poland
jduda@zarz.agh.edu.pl, skalna@agh.edu.pl

Abstract. The problem of computing the hull, that is the tightest interval enclosure of the solution set for linear systems with parameters being nonlinear functions of interval parameters, is an NP-hard problem. However, since the problem of computing the hull can be considered as a combinatorial or as a constrained optimisation problem, metaheuristic techniques might be helpful. Alas, experiments performed so far show that they are time consuming and their performance may depend on the problem size and structure, therefore some acceleration and stabilisation techniques are required. In this paper, a new approach which rely on a multi-agent system is proposed. The idea is to apply evolutionary method and differential evolution for different agents working together to solve constrained optimisation problems. The results obtained for several examples from structural mechanics involving many parameters with large uncertainty ranges show that some synergy effect of the metaheuristics can be achieved, especially for problems of a larger size.

1 Introduction

The paper addresses the problem of solving large-scale linear algebraic systems whose elements are nonlinear functions of parameters varying within prescribed intervals. Such systems arise, e.g., in reliability and risk analysis of engineering systems. The experiments done so far (see [14]) were designed to test the most popular metaheuristics such as evolutionary algorithm (EA), tabu search (TS), simulated annealing (SA) and differential evolution (DE) for their suitability to solve such problems, especially in the case of many parameters and large uncertainty. The experiments demonstrated that for relatively small problems the considered methods give very accurate results. However, for larger problems the computation time is significant, which limits their usage. To shorten the computation time, the algorithms were run in parallel. This allowed to reduce the total computation time, but further improvements are still required. During the experiments it was found that evolutionary method and differential evolution give significantly better results than the remaining algorithms and depending on the problem characteristic and size either EA or DE was the winning strategy. This suggested to the authors to employ both methods working together as agents and exchanging the best solutions between each other, instead of using parallelised methods independently. The proposed approach is presented in Section 3.

P. Manninen and P. Öster (Eds.): PARA 2012, LNCS 7782, pp. 465–472, 2013.

The rest of the paper is organised as follows. Parametric interval linear systems are described in Section 2. In Section 4, computational experiments for illustrative problem instances from structural mechanics with different number of intervals as well as different uncertainty ranges are presented in order to verify the usefulness of the proposed approach. The results for homogenous agents, using either evolutionary method or differential evolution and heterogeneous agents, where half of agents use one method, whereas other half of agents use the other one are provided and compared with the results obtained by single algorithms. The paper ends with concluding remarks.

2 Parametric Interval Linear Systems

Systems of parametric linear equations arise directly in various problems (e.g., balance of forces, of electric current, etc.) Systems of linear equations arise also indirectly in engineering problems through the use of numerical methods, e.g., by discrete solution of differential equations.

A parametric linear systems is a linear system of the form:

$$A(p)x(p) = b(p), \tag{1}$$

where $A(p) = [a_{ij}(p)]$ is an $n \times n$ matrix, $b(p) = [b_j(p)]$ is n-dimensional vector, and $a_{ij}(p)$, $b_i(p)$ are general nonlinear functions of $p = (p_1, \ldots, p_k)^T$ which is a k-dimensional vector of real parameters.

Often, the parameters p_i are unknown which stems, mainly, from the scarcity or lack of data. This kind of uncertainty is recognised as *epistemic* uncertainty and can (or [5] should) be modelled using interval approach, that is using only range information. In the interval approach ([1], [8], [9]), a true unknown value of a parameter p_i is enclosed by an interval $\boldsymbol{p}_i = [\check{p} - \Delta p, \check{p} + \Delta p]$, where $\check{p}$ is an approximation of p_i (e.g., resulting from an inexact measurement) and $\Delta p > 0$ is an upper bound of an approximation (measurement) error. Obviously, appropriate methods are required to propagate interval uncertainties through a calculation (see, e.g., [1], [8], [9]).

Thus, if some of the parameters are assumed to be unknown, ranging within prescribed intervals, $p_i \in \boldsymbol{p}_i$ $(i = 1, \ldots, k)$, the following family of parametric linear system, usually called *parametric interval linear system (PILS)*,

$$A(p)x(p) = b(p),\ p \in \boldsymbol{p} \tag{2}$$

is obtained, where $\boldsymbol{p} = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_k)^T$.

The set of all solutions to the point linear systems from the family (2) is called a *parametric solution set* and is defined as

$$S_{\boldsymbol{p}} = \{x \in \mathbb{R}^n \mid \exists\, p \in \boldsymbol{p}\ \ A(p)x = b(p)\}\,. \tag{3}$$

This set is generally of a complicated non-convex structure [2]. In practise, therefore, an interval vector $\boldsymbol{x}^*$, called the *outer solution*, satisfying $S_{\boldsymbol{p}} \subseteq \boldsymbol{x}^*$ is computed. The tightest outer solution, with respect to the inclusion property, is called a *hull solution* (or simply a hull) and is denoted by $\square S_{\boldsymbol{p}}$:

$$\square S_{\boldsymbol{p}} = \bigcap \{\boldsymbol{Y} \mid S_{\boldsymbol{p}} \subseteq \boldsymbol{Y}\}\,.$$

Computing the hull solution is in general case NP-hard [12]. However, the problem of computing the hull can be considered as the family of the following $2n$ constrained optimisation problems:

$$\begin{aligned} \underline{x}_i &= \min\{x(p)_i \mid A(p)x(p) = b(p),\, p \in \boldsymbol{p}\}, \\ \overline{x}_i &= \max\{x(p)_i \mid A(p)x(p) = b(p),\, p \in \boldsymbol{p}\}, \end{aligned} \tag{4}$$

and, therefore, heuristic approach can be used to find very good approximations of the required optima while minimising the computation overhead. Additionally, in this paper it is claimed that the usage of metaheuristic agents strategy allows additional reduction in the computational time.

Theorem 1. *Let $\underline{x}_i$ and $\overline{x}_i$ denote, respectively, the solution of the i-th minimisation and maximisation problem (4). Then, the hull solution*

$$\square S_{\mathbf{p}} = \square\{x(p) : A(p)x(p) = b(p), p \in \boldsymbol{p}\} = [\underline{x}_1, \overline{x}_1] \times ... \times [\underline{x}_n, \overline{x}_n]. \tag{5}$$

3 Methodology

3.1 Evolutionary Multi-agent System

Different strategies can be used in order to compute population-based metaheuristics in parallel. In the so called *global parallelisation model* there is one population, and computation of objective function are done in parallel on slave units [15]. This approach is particularly useful for multicore or multiprocessor architectures where communication cost is almost negligible. In the *island model* the whole population is divided into subpopulations that can be run on different heterogeneous machines. Since in this case communication time is significant, thus the subpopulations are run independently and they occasionally exchange solutions. Finally, in the *master-slave model* there is one central (master) population that communicates with other subpopulations to collect (and use) their best solutions.

When solving parametric interval linear systems, the time spent for computing the objective function significantly dominates the time spent for communication between algorithms, so the island model approach seems to be the most suitable. Agents are run independently and communicate with each other after a given time has elapsed (1-3 seconds). In the preliminary experiments, agents communicate by exchanging their best so far solutions stored in auxiliary files, but in the future more effective communication methods is planned. Three variants of island model have been considered. Two of them were homogeneous multi-agent systems based either on evolutionary method or differential evolution, while the third one was a heterogeneous system with half agents based on one method and the other half based on the other one. In the following sections metaheuristics used by agents are briefly described.

3.2 Evolutionary Optimisation

Population P consists of pop_{size} individuals characterised by k-dimensional vectors of parameters $p_i = (p_{i1}, \ldots, p_{ik})^T$, where $p_{ij} \in \boldsymbol{p}_j$, $i = 1, \ldots, pop_{size}$, $j = 1, \ldots, k$. Elements of the initial population are generated at random based on the uniform distribution. The 10% of the best individuals pass to the next generation and the rest of the population is created by the *non-uniform mutation*

$$p'_j = \begin{cases} p_j + (\overline{p}_j - p_j)\, r^{(1-t/n)^b}, & \text{if } q < 0.5 \\ p_j + (p_j - \underline{p}_j)\, r^{(1-t/n)^b}, & \text{if } q \geqslant 0.5 \end{cases} \tag{6}$$

and *arithmetic crossover*

$$p^{1\prime} = rp^1 + (1-r)p^2, \quad p^{2\prime} = rp^2 + (1-r)p^1 \tag{7}$$

It turned out from numerical experiments [14] that mutation rate r_{mut} should be close to 1 (r_{mut}=0.95), and the crossover rate r_{crs} should be less than 0.3 (r_{crs}=0.25). Population size and the number of generations n depend strongly on the problem size (usually pop_{size} should be set to at least 16 and n to 30). General outline of the EO algorithm is shown in Fig. 1.

Initialise P of pop_{size} at random
$j = 0$ /* number of generation */
while $(j < n)$ **do**
  Select P' from P; Choose parents p_1 and p_2 from P'
  if $(r_{[0,1]} < r_{crs})$ **then** Offspring o_1 and $o_2 \longleftarrow$ Recombine p_1 and p_2
  if $(r_{[0,1]} < r_{mut})$ **then** Mutate o_1 and o_2
end while

Fig. 1. Outline of an evolutionary algorithm

3.3 Differential Evolution

Differential evolution (DE) has been found to be a very effective optimisation method for continuous problems [3]. DE itself can be treated as a variation of evolutionary algorithm, as the method is founded on the same principles such as selection, crossover, and mutation. However, in DE the main optimisation process is focused on the way the new individuals are created. Several strategies for constructing new individuals [11] have been defined. Basic strategy described as $/rand/1/bin$ (which means that vectors for a trial solution are selected in a random way and binomial crossover is then used) creates a mutated individual p_m as follows

$$p_m = p_1 + s \cdot (p_2 - p_3) \ , \tag{8}$$

where s is a *scale parameter* called also an *amplification factor*. After a series of experiments, the best strategy for the problem of solving large PILS appeared to be the strategy described as $/best/2/bin$ (compare [4]). In this strategy a mutated individual (trial vector) is created on the basis of the best solution p_{best} found so far and four other randomly chosen individuals

$$p_m = p_{best} - s \cdot (p_1 + p_2 - p_3 - p_4) \ . \tag{9}$$

The mutated individual p_m is then mixed with the original individual p with a probability C_R using the following *binomial crossover*

$$p'_j = \begin{cases} p_{mj}, & \text{if } r \geqslant C_R \text{ or } j = r_n \\ p_j, & \text{if } r > C_R \text{ and } j \neq r_n \end{cases} , \tag{10}$$

where $r \in [0, 1]$ is a random number and $r_n \in (0, 1, 2, ..., D)$ is a random index ensuring that p'_j is a at least an element obtained by p_{mj}. The following parameters values were taken $s = 0.8$ and $C_R = 0.9$, as the most efficient.

```
Initialise P of pop_size at random
while (i < n) do
   Do
      Choose at random 4 individuals p1, p2, p3, p4
      Generate mutant pm from pbest and from p1, p2, p3, p4
   While (pm is not valid)
   p' ⟵ Crossover(p, pm)
   if (f(p') > f(pi)) then p_{i+1} ⟵ p' else p_{i+1} ⟵ pi
end while
```

Fig. 2. Outline of a differential evolution algorithm

4 Numerical Experiments

The multi-agent system proposed by the authors has been tested for the three exemplary truss structures, each of different size: four bay two floor truss, five bay six floor truss and ten bay eleven floor truss. Additionally, different levels of uncertainty for the parameters and the load have been considered: 40% for the first truss, 10% for the second, and 6% for the third truss.

Three variants of the island model have been tested for each of the test problems. Each system consisted of 8 independent agents. Number of generations n and population size pop_{size} for both evolutionary computation and differential evolution were set to the same values. For the first problem $n = 300$, $pop_{size} = 30$, for the second problem $n = 100$, $pop_{size} = 20$, and for the third problem n=10, $pop_{size} = 10$.

In order to compare the proposed variants, a measure similar to the overestimation measure described by Popova [10] was used. This time, however, as the algorithms computed the inner interval of the hull solution, the overestimation measure was calculated in the relation to the tightest inner solution, i.e. the worst estimation of the hull solution. The measure can be treated as a relative increase over the tightest inner solution and will be marked as RITIS. For each variant of the multi-agent system the result of the best agent, average result of all agents and the worst result obtained by a given system or algorithm are provided. The same data are listed for the evolutionary optimisation and differential evolution ran as a single algorithm. The overestimation is computed over the worst solution coming from all experiments for a given problem size. Comparison of the results for the first test problem consisting of 15 nodes and 38 elements is presented in Table 1.

Table 1. RITIS measure for four bay two floor truss (higher is better)

Multi-agent system	Largest RITIS	Average RITIS	Smallest RITIS	Computation time [sec.]
HomEO	1.9%	1.8%	1.7%	274
HomDE	1.5%	1.4%	1.4%	289
Heter	2.0%	1.8%	1.6%	298
SingleEO	0.1%	0.0%	0.0%	318
SingleDE	1.4%	1.3%	1.3%	232

Hull approximations obtained by the homogenous agent system based only on the evolutionary optimisation method (HomEO) and the heterogeneous system (Heter) with both EO and DE agents were on average better than the approximations obtained by the other systems and algorithms. The system with heterogeneous agents achieved the best approximation of the hull solution, but on average those two systems performed the same. Algorithms running alone with the same parameters as for the multi-agent systems obtained the worst results, however, the approximations generated by differential evolution were only slightly worse than the results obtained by DE agents working together. Contrary to the agents based on the evolutionary method, the solutions provided by the agents based on DE do not sum up in a simple way.

The results for the second test problem that consisted of 36 nodes and 110 elements are collected in Table 2.

Table 2. RITIS measure for five bay six floor truss (higher is better)

Multi-agent system	Largest RITIS	Average RITIS	Smallest RITIS	Computation time [min.]
HomEO	18.3%	17.0%	14.9%	31.7
HomDE	20.7%	20.1%	19.5%	31.5
Heter	22.9%	19.6%	16.9%	34.0
SingleEO	0.1%	0.0%	0.0%	43.1
SingleDE	19.4%	19.2%	19.0%	21.8

This time the results obtained by the agents in the heterogeneous mutli-agent system (Heter) were similar to the results obtained by the homogenous system with the agents using differential evolution (HomDE). The synergy effect of EO and DE allowed to achieve the best hull approximation, however, on average the heterogeneous system performed a little worse than HomDE.

Finally, Table 3 gathers the results for the largest of all test problems consisting of 120 nodes and 420 elements.

Table 3. RITIS measure for ten bay eleven floor truss (higher is better)

Multi-agent system	Largest RITIS	Average RITIS	Smallest RITIS	Computation time [min.]
HomEO	10.4%	5.7%	1.4%	288
HomDE	31.7%	30.7%	29.8%	291
Heter	33.1%	25.3%	5.1%	289
SingleEO	0.1%	0.1%	0.0%	356
SingleDE	18.7%	18.5%	18.2%	267

For the largest problem considered the homogenous multi-agent system based on differential evolution (HomDE) and the single DE algorithm performed on average better than others systems. Evolutionary optimisation method gave much worse results than differential evolution, thus the agents based on EO could not go hand in hand with the agents using DE method and it caused that heterogeneous agents were on average worse than homogenous DE agents. It is also wort to notice that, unlike previous experiments, approximations obtained by the DE mutli-agent system were significantly better (by 66%) that those generated by the single DE algorithm.

5 Conclusions

Heterogeneous multi-agent evolutionary system for solving parametric interval linear systems has been proposed in the paper. Although some examples of evolutionary multi-agent systems can be found in literature ([7],[6]), the system proposed by the authors can use two different methods that are based on the idea of evolution: evolutionary algorithm and differential evolution. Numerical experiments performed by the authors have shown that the proposed approach can bring a synergy effect of those two metaheuristics. Despite the experiments were computed on a single multiprocessor machine the proposed muti-agent system can be easily applied in distributed computing. This would allow to use more than 8 agents and the differences in the hull approximation between multi-agent systems and single algorithms would be more significant.

Future studies should focus on finding more efficient metaheuristic algorithms for heterogeneous agents, capable to provide good results for the problems of large size, like the third test problem presented in the paper. The authors also plan to test such metaheuristics like ant colony optimisation (ACO) and artificial bee colony (ABC). Also evolutionary method might be improved by introducing some local search algorithms based e.g. on iterated local search (ILS) or variable neighbourhood search (VNS).

Acknowledgements. This work was partly financed within the statutory activity no 11/11.200.232.

References

1. Alefeld, G., Herzberger, J.: Introduction to Interval Computations (transl. by J. Rokne from the original German 'Einführung In Die Intervallrechnung'), pp. xviii–333. Academic Press Inc., New York (1983)
2. Alefeld, G., Kreinovich, V., Mayer, G.: The Shape of the Solution Set for Systems of Interval Linear Equations with Dependent Coefficients. Mathematische Nachrichten 192(1), 23–36 (2006)
3. Dréo, J., Pétrowski, A., Siarry, P., Taillard, E.: Metaheuristics for Hard Optimization. Springer (2006)
4. Duda, J., Skalna, I.: Differential Evolution Applied to Large Scale Parametric Interval Linear Systems. In: Lirkov, I., Margenov, S., Waśniewski, J. (eds.) LSSC 2011. LNCS, vol. 7116, pp. 206–213. Springer, Heidelberg (2012)
5. Ferson, S., Ginzburg, L.R.: Different methods are needed to propagate ignorance and variability. Reliability Engineering and Systems Safety 54, 133–144 (1996)
6. Hanna, L., Cagan, J.: Evolutionary Multi-Agent Systems: An Adaptive and Dynamic Approach to Optimization. Journal of Mechanical Design 131(1), 479–487 (2009)
7. 't Hoen, P.J., de Jong, E.D.: Evolutionary Multi-agent Systems. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 872–881. Springer, Heidelberg (2004)
8. Moore, R.E.: Interval Analysis. Prentice-Hall, Inc., Englewood Cliffs (1966)
9. Neumaier, A.: Interval Methods for Systems of Equations, pp. xvi–255. Cambridge University Press, Cambridge (1990)
10. Popova, E., Iankov, R., Bonev, Z.: Bounding the Response of Mechanical Structures with Uncertainties in all the Parameters. In: Muhannah, R.L., Mullen, R.L. (eds.) Proceedings of the NSF Workshop on Reliable Engineering Computing (REC), pp. 245–265 (2006)
11. Price, K.S., Rainer, M., Lampinen, J.A.: Differential Evolution. A Practical Approach to Global Optimization. Springer (2005)
12. Rohn, J., Kreinovich, V.: Computing exact componentwise bounds on solutions of linear systems with interval data is NP-hard. SIAM Journal on Matrix Analysis and Application, SIMAX 16, 415–420 (1995)
13. Rump, S.M.: Verification Methods for Dense and Sparse Systems of Equations. In: Herzberger, J. (ed.) Topics in Validated Computations, pp. 63–135. Elsevier Science B. V. (1994)
14. Skalna, I., Duda, J.: A Comparison of Metaheurisitics for the Problem of Solving Parametric Interval Linear Systems. In: Dimov, I., Dimova, S., Kolkovska, N. (eds.) NMA 2010. LNCS, vol. 6046, pp. 305–312. Springer, Heidelberg (2011)
15. Talbi, E.G.: Parallel combinatorial optimization. John Wiley and Sons (2006)