Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi

Afyon Kocatepe University Journal of Science and Engineering

AKÜ FEMÜBİD **18** (2018) 015601 (256-264)

AKU J. Sci. Eng. **18** (2018) 015601 (256-264)

DOİ: 10.5578/fmbd.66584

A Study on Size Optimization of Trusses with BB-BC Algorithm: Review and Numerical Experiments

Hakan Özbaşaran

Department of Civil Engineering, Eskişehir Osmangazi University, 26480, Eskişehir, Turkey

e-mail: ozbasaran@ogu.edu.tr

Geliş Tarihi:27.05.2017 ; Kabul Tarihi:05.03.2018

Keywords

Truss;

Optimization;

Big Bang - Big Crunch;

Metaheuristic

Abstract

In the last decades, great attention has been paid to structural optimization with stochastic methods. The applications of metaheuristic algorithms have become popular, which mostly provide solutions with adequate precision for the structural optimization problems from an engineering point of view. However, these algorithms should be specifically tuned for the considered problem to obtain satisfactory results. Big Bang - Big Crunch is one of the efficient metaheuristic optimization algorithms that is based on the famous theory on the evolution of the universe. A considerable number of researchers presented applications of the Big Bang - Big Crunch algorithm for the size optimization of trusses, which is an inviting challenge in the structural optimization field. This study revisits the size optimization of trusses with continuous variables using the Big Bang - Big Crunch algorithm, discusses the previously introduced improvements and presents the results of a few experimental modifications.

Kafes Sistemlerin BB-BC Algoritması ile Boyut Optimizasyonu Üzerine bir Çalışma: İnceleme ve Sayısal Deneyler

Anahtar Kelimeler

Kafes sistem;

Optimizasyon;

Big Bang - Big Crunch;

Metasezgisel

Özet

Son birkaç on yılda, stokastik yöntemler ile yapısal optimizasyon konusuna büyük ilgi gösterilmiştir. Yapısal optimizasyon problemlerine mühendislik bakış açısına göre çoğunlukla yeterli hassasiyette çözümler sağlayan metasezgisel algoritmaların uygulamaları popüler hale gelmiştir. Ancak, tatmin edici sonuçlar elde edebilmek için, bu algoritmaların ele alınan probleme özel olarak düzenlenmeleri gerekmektedir. Evrenin oluşumu ile ilgili ünlü teori üzerine kurulu olan Big Bang - Big Crunch algoritması, etkili metasezgisel algoritmalardan biridir. Kayda değer sayıda araştırmacı, yapısal optimizasyon alanındaki çekici yarışlardan biri olan kafes sistemlerin boyut optimizasyonu için Big Bang - Big Crunch algoritmasının uygulamalarını sunmuşlardır. Bu çalışmada, Big Bang - Big Crunch algoritması ile kafes sistemlerin sürekli değişkenler kullanılarak optimizasyonu konusu tekrar incelenmiş, daha önce tanıtılan iyileştirmeler tartışılmış ve birkaç deneysel değişikliğin sonuçları sunulmuştur.

1. Introduction

There are three optimization types for truss structures as size, shape and topology optimization. In shape optimization, the optimal design is searched by moving the joints within the design space while the number of the bars used in the structure and the connection information stay the same. In addition to joint positions, bar count and connection information can be changed in a topology optimization application. Size optimization, which is the main concern of the study, only takes the cross sections of the bars into account as design variables. The objective of the size optimization procedure is to minimize the weight of the structure as;

$$\min w = \sum_{i=1}^{n} \gamma L_i A_i,\ \ (A_i \in \mathbb{R}_{\geq 0}) \tag{1}$$

for the design variables,

$$X_j = \{A_1, A_2, \dots, A_{n-1}, A_n\},\ \ (A^L \leq A_i \leq A^U) \tag{2}$$

In Eq. 1, w is the weight of the structure, n is the total number of the bars, γ is the unit weight of the structural material, L_i and A_i are the length and section area of the i^{th} bar, respectively. In Eq. 2, X_j is the j^{th} candidate solution. Finally, A^L and A^U are the lower and upper allowed limits for the section area, respectively. Size optimization procedure can be carried out by controlling a variety of constraints such as stress, displacement and frequency.

The first paper that author could find in the literature on size optimization of trusses with the BB-BC algorithm (Erol and Eksin 2006) is written by Camp (2007). Camp proposed a multiphase search procedure and an alternative equation to calculate the next positions of the particles. Kaveh and Talatahari (2009) had presented a hybrid BB-BC algorithm that utilizes a sub-optimization mechanism and uses a more extensive equation for the redistribution. Then, they extended their work to discrete size optimization of skeletal structures (Kaveh and Talatahari 2010a) and topology optimization of Schwedler and ribbed domes (Kaveh and Talatahari 2010b). Casavola et al. (2012) powered the BB-BC algorithm with pseudo-gradient information and presented the results for the size optimization of two benchmark problems. Kaveh and Zolghadr (2012) developed a hybrid method with trap recognition capabilities by using the Charged System Search and BB-BC algorithms and evaluated their method with truss size and shape optimization examples considering natural frequency constraints. Hasançebi and Azad (2013) presented a new redistribution equation for size optimization of skeletal structures. Azad et al. (2013) proposed the Upper Bound Strategy (UBS) to reduce the number of analyses and validated the performance of their strategy with various truss-sizing problems. Another hybrid algorithm is presented by Kaveh and Mahdavi (2013) for size optimization of trusses with multiple frequency constraints. Their work installs the Quasi-Newton method to the original BB-BC algorithm as an additional step. Milajić et al. (2014) presented the size optimization of spatial trusses with the BB-BC algorithm. Hasançebi and Azad (2014) presented an application of their previously proposed modification (Hasançebi and Azad 2013) for discrete size optimization of steel trusses. Lotfi and Ghoddosian (2015) presented the size optimization of two-dimensional trusses by Kaveh and Talatahari's (2009) hybrid algorithm. Milajić and Beljaković (2016) compared four BB-BC variants for the multi-objective (size and displacement) optimization of trusses. Jain et al. (2016) developed a Visual Basic interface for the optimization of trusses with BB-BC algorithm, which uses STAAD.Pro for analysis and AutoCAD for drafting.

This paper revisits the size optimization of trusses with the BB-BC algorithm using continuous variables and presents the contribution of a few experimental modifications to the algorithm on one of the famous truss-sizing optimization benchmark problems. The first thoughts on this paper were presented by the author in ICOME2017 (Özbaşaran 2017).

2. The Algorithm

The BB-BC algorithm simulates the well-known theory on the evolution of the universe. It consists of two main phases as big bang and big crunch. First, the particles are scattered to the design space (big bang). Then, they shrank to a single point, which is called the center of mass (big crunch). The analogy of the BB-BC algorithm is simple. The particles are the candidates. Each design variable represents a coordinate in the design space. Therefore, location of a particle is a candidate solution. And, the mass of a candidate is its objective value.

Step 1: Initialization

The particles are randomly scattered to the design space. It will be proper to draw the initial coordinates from a uniform distribution. The coordinates (design variables of the candidate solution) of the j^{th} particle (candidate) X_j are stored in the form $\{x_{j,1}, x_{j,2}, \dots, x_{j,n}\}$.

Step 2: Calculate the masses (objective values) of the particles

The objective value of the j^{th} candidate is represented by $f(X_j)$, which corresponds to the penalized weight of the structure in most of the size optimization problems.

Step 3: Calculate the center of the mass

The algorithm offers two ways to calculate the center of mass (X_c). First one is as follows:

$$X_c = \frac{\sum_{j=1}^{t} \frac{1}{f(X_j)} X_j}{\sum_{j=1}^{t} \frac{1}{f(X_j)}} \tag{3}$$

In Eq. 3, t is the particle count. An important note is Eq. 1 is for minimization problems and calculates the "inverse" center of mass. Since truss size optimization is a minimization problem, Eq. 3 assumes that the center of mass is closer to the lighter particles. Note that the terms $1/f(X_j)$ should be replaced by $f(X_j)$ for maximization problems. The second alternative is assuming the best candidate in the candidates list (the local best solution: X_{lb}) as the center of mass (Eq. 4).

$$X_c = X_{lb} \tag{4}$$

Step 4: Redistribute the particles around the center of mass

This step simulates the big bang and recalculates the new positions of the particles as:

$$X_{j+1} = X_c + \frac{r x^U}{k} \tag{5}$$

where, x^U is the upper limit of the parameter, r is a normal random number and k is the iteration step.

Step 5: Closing the loop

The new positions of the particles calculated by Eq. 5 may go out of the design space. Any method can be used to keep the design variables in the allowed limits. The algorithm repeats by returning to Step 2 until the stopping criteria have been met.

3. Literature Review

The improvements presented for the BB-BC algorithm mostly modify the redistribution equation and/or create hybrids with other algorithms. Table 1 presents the modifications made on redistribution equations in the studies that are given in the references list.

Table 1. Redistribution equations

Study	Redistribution Equation
(Erol and Eksin 2006)	$X_j = X_c + \frac{r x^U}{k}$
(Camp, 2007)	$X_j = \alpha_2 X_c + (1-\alpha_2) X_{gb} + \alpha_1 \frac{r(x^U - x^L)}{k}$
(Kaveh and Talatahari 2009)	$X_j = \alpha_2 X_c + (1-\alpha_2)\left(\alpha_3 X_{gb} + (1-\alpha_3) X_{lb}\right) + \alpha_1 \frac{r(x^U - x^L)}{k+1}$
(Casavola et al. 2012)	$X_j = \frac{X_c + X_{gb}}{2} + r_{[0,1]}(X_f - X_c) + r_{[0,1]}(X_{gb-1} - X_c) + r_{[0,1]}(X_{gb} - X_c)\mu$
(Kaveh and Zolghadr 2012)	$X_j = X_c + \frac{r L_u}{k}$
(Hasançebi and Azad 2013)	$X_j = X_c + \mathrm{Round}\left[\alpha_1 r^{\alpha_2} \frac{(x^U - x^L)}{k}\right]$
(Kaveh and Mahdavi 2013)	$X_j = X_c + \alpha_1 \frac{r(x^U - x^L)}{1 + \frac{k}{\alpha_2}}$

In Table 1, notation of the original papers are changed in order to clearly indicate the differences between the redistribution equations. The equation presented by Camp (2007) considers the contribution of the global best solution (X_{gb}) obtained so far with an α_2 adjusting parameter. The α_1 parameter regulates the balance between the random part and the rest of the equation. Kaveh and Talatahari's (2009) equation keeps the balance between the contribution of the local best (X_{lb}) solution and the global best solution with the α_3 parameter. The methods introduced in Casavola et al. (2012), Kaveh and Mahdavi (2013) and Kaveh and Zolghadr (2012) are hybrid algorithms. Casavola et al. (2012) benefitted from the largest and steepest descent. In their equation, $r_{[0,1]}$ is a random number drawn from a uniform distribution between 0 and 1. The symbol X_f is for the solution that has the steepest descent with respect to X_c and X_{gb-1} is the second global best solution. Kaveh and Zolghadr (2012) used a Charged System Search powered algorithm and Kaveh and Mahdavi (2013) performed local searches with a quasi-Newton method called Broyden-Fletcher-Goldfarb-Shannon (BFGS) method between the loops. The redistribution equation presented by Hasançebi and Azad (2013) is for combinatorial optimization. However, it can be used in optimization with continuous variables by removing the rounding operator. The objective functions used in the cited studies are presented in Table 2.

Table 2. Objective functions

Study	Objective Function
(Camp, 2007)	$W_j = w_j\left(1+\phi_{j,\sigma}+\phi_{j,\delta}\right)^{\varepsilon_1}$
(Kaveh and Talatahari 2009)	$W_j = \varepsilon_3 w_j + \varepsilon_2\left(\phi_{j,\sigma}+\phi_{j,\delta}\right)^{\varepsilon_1}$
(Casavola et al. 2012)	N/A
(Kaveh and Zolghadr 2012)	$W_j = w_j\left(1+\varepsilon_2\phi_{j,\omega}\right)^{\varepsilon_1}$
(Hasançebi and Azad 2013)	$W_j = w_j\left[1+\varepsilon_2\left(\phi_{j,\sigma}+\phi_{j,\delta}\right)\right]$
(Kaveh and Mahdavi 2013)	$W_j = w_j\left[1+\varepsilon_2\left(\phi_{j,\sigma}+\phi_{j,\delta}\right)^2\right]$

In Table 2, W_j and w_j are the objective function and the weight of the j^{th} candidate, respectively. $\phi_{j,\sigma}$, $\phi_{j,\delta}$ and $\phi_{j,\omega}$ represent the stress, displacement and frequency constraint violations, respectively. Finally, ε_1, ε_2 and ε_3 are the weighting coefficients. The efficiency of the optimization methods are measured by benchmark problems. A spatial truss structure called "25-Bar transmission tower" is used for benchmarking in most of the truss size optimization papers (Figure 1).

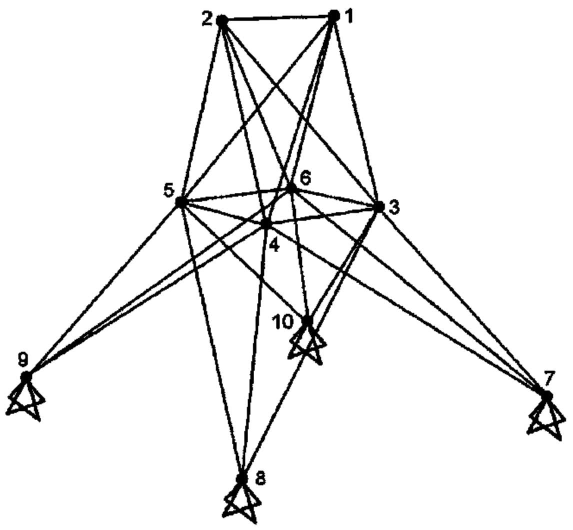

Figure 1. Configuration of the 25-Bar transmission tower

The structure given in Figure 1 has 25 bars that are classified into 8 groups. Tables 3 and 4 present the node coordinates and connectivity information of the transmission tower, respectively.

Table 3. Node coordinates of the transmission tower

Node	x (in.)	y (in.)	z (in.)
1	-37.5	0.0	200.0
2	37.5	0.0	200.0
3	-37.5	37.5	100.0
4	37.5	37.5	100.0
5	37.5	-37.5	100.0
6	-37.5	-37.5	100.0
7	-100.0	100.0	0.0
8	100.0	100.0	0.0
9	100.0	-100.0	0.0
10	-100.0	-100.0	0.0

Table 4. Element connectivity information of the transmission tower

Element Group			
1	2	3	4
1: (1,2)	2: (1,4)	6: (2,4)	10: (3,6)
	3: (2,3)	7: (2,5)	11: (4,5)
	4: (1,5)	8: (1,3)	
	5: (2,6)	9: (1,6)	
Element Group			
5	6	7	8
12: (3,4)	14: (3,10)	18: (4,7)	22: (6,10)
13: (5,6)	15: (6,7)	19: (3,8)	23: (3,7)
	16: (4,9)	20: (5,10)	24: (4,8)
	17: (5,8)	21: (6,9)	25: (5,9)

In Table 3, the dimensions are given in inches. In Table 4, $i:(a,b)$ notation is for the ith bar that connects the node a to node b. This tower is optimized for both single and multiple loading cases in the literature. In this study, the multiple loading cases (Table 5) are considered.

Table 5. Loading conditions for the transmission tower

Case	Node	P_x (kips)	P_y (kips)	P_z (kips)
1	1	0.0	20.0	-5.0
	2	0.0	-20.0	-5.0
2	1	1.0	10.0	-5.0
	2	0.0	10.0	-5.0
	3	0.5	0.0	0.0
	6	0.5	0.0	0.0

The nodal displacements are limited to ± 0.35 in all global directions. The allowed compressive and tensile stress values of each group are as given in Table 6 for the introduced problem.

Table 6. Stress limits for the transmission tower

Element Group		Compressive Stress Limit (ksi)	Tensile Stress Limit (ksi)
1	A_1	35.092	40.0

2	A_2~A_5	11.590	40.0
3	A_6~A_9	17.305	40.0
4	A_{10}~A_{11}	35.092	40.0
5	A_{12}~A_{13}	35.092	40.0
6	A_{14}~A_{17}	6.759	40.0
7	A_{18}~A_{21}	6.959	40.0
8	A_{22}~A_{25}	11.082	40.0

Since this study compares the results of the single objective optimization (weight) with only stress and deflection constraints using continuous variables, there are four studies to place into the comparison tables in the references list (Table 7).

Table 7. Optimization results as given in the references

Study	Best Weight (lb)	Average Weight (lb)	Number of Analyses	C_v (%)	Particle Count	Runs
(Camp, 2007)	545.48	546.40	9746	0.00	100	100
(Kaveh and Talatahari 2009)	545.16	545.66	12500	0.01	50	50
(Casavola et al. 2012)	545.11	N/A	593	0.19	20	N/A
(Milajić et al. 2014)	545.16	N/A	N/A	0.00	N/A	N/A

It should be noted that Camp (2007) obtained better solutions by implementing an additional phase (Phase 2). However, results of Phase 1 are considered in this paper. Kaveh and Talatahari (2009) obtained a better average weight with less runs compared to Camp's study. Casavola et al. (2012) improved the computation time by significantly decreasing the number of analyses utilizing a hybrid procedure. Unfortunately, Milajić et al. (2014) did not provide detailed information about the optimization process.

There are 55 constraints of the introduced problem (3×10 node displacement components and 25 stress values). Each node has three displacement components in the global directions x, y and z. For example, 12th displacement component represents the displacement of 4th node in the global z direction. Total constraint violation ratio is calculated by the equation given below.

$$C_v = \sum_{p=1}^{25} \phi_{p,\sigma} + \sum_{m=1}^{30} \phi_{m,\delta} \tag{6}$$

where $\phi_{p,\sigma}$ is the stress constraint violation ratio of the p^{th} bar and $\phi_{m,\delta}$ is the constraint violation ratio of the m^{th} deflection component, which are calculated as follows.

$$\phi_{p,\sigma} = \left| \frac{\sigma_p - \sigma^{lim}}{\sigma^L} \right| \tag{7}$$

$$\phi_{m,\delta} = \left| \frac{\delta_m - \delta^{lim}}{\delta^L} \right| \tag{8}$$

In Eqs. 7 and 8, the superscript lim indicates the limiting value, which may be the upper or lower limit. σ is for bar stresses and δ is for node displacements. Constraint violation is taken 0 if the design variable is in the allowed limits. Note that there are constraint violations in the designs presented by Kaveh and Talatahari (2009) and Casavola et al. (2012) (see Table 7).

4. Numerical Experiments

The first experiment involves the bounding methods. Due to the nature of the algorithm, particles may go out of the design space after applying the movement operator. Therefore, the last step of the loop is "Then new point is upper and lower bounded" in the own words of the author (Erol and Eksin 2006). One can use different bounding methods such as;

- *CC*: Clip the exceeding part
- *CR*: Clip if smaller than the lower limit, regenerate if greater than the upper limit
- *RC*: Regenerate if smaller than the lower limit, clip if greater than the upper limit
- *RR*: Regenerate the coordinate

Clipping is assigning the lower or upper limit to the design variable if its value exceeds the defined limits. Regeneration is applying the same redistribution equation with different random numbers until a value that is in the design space is obtained. Table 8 summarizes the optimization results of the 25-Bar transmission tower obtained by Camp's equation using various bounding methods. It should be reminded that Phase 2 of

Camp's work is not considered. The parameters α_1 and α_2 are taken 1 and 0.2, respectively.

In the further tables (including Table 8), each row represents the results of 100 consecutive runs. The stopping criterion is "No improvement is observed in the last 20 iterations". The number of the particles (population count) is set to 100 and the simple penalty function mentioned in the Camp's work is used unless indicated otherwise.

Table 8. The results obtained by using Camp's equation and the introduced bounding methods

Bounding Method	Best Weight (lb)	Average Weight (lb)	Worst Weight (lb)	Avg. No. of Analyses	C_v (%)
(CC)	545.44	549.34	569.41	10371	0.00
(CR)	545.48	549.84	564.78	10047	0.00
(RC)	545.63	550.30	566.77	12375	0.00
(RR)	545.76	549.99	564.73	11860	0.00

It can be seen from Table 8 that the introduced bounding methods do not have a significant effect on the best, average and worst solutions. However, the algorithm with CR bounding converged to near-optimal solutions with less than about 18.8% and 15.3% structural analyses compared to RC and RR methods, respectively. The convergence history of the best solutions are presented in Figure 2.

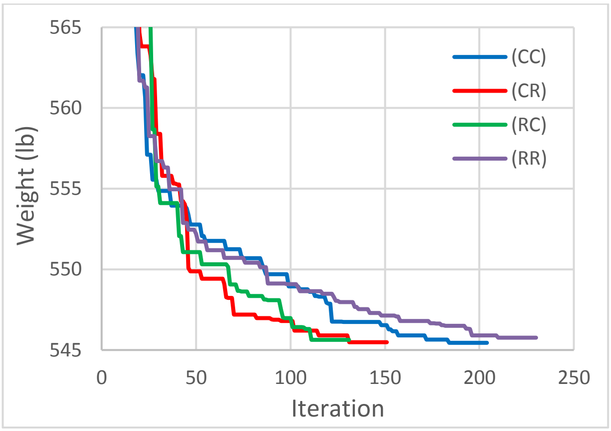

Figure 2. Convergence history of the best solutions for the introduced bounding methods

In Figure 2, the limits of the vertical axis are selected adequately close to each other to clearly show the difference between different bounding methods. It is seen from Figure 2 that the number of structural analyses of the best solutions do not reflect the average behavior of bounding methods.

The effect of UBS on various redistribution equations are studied. The UBS scheme of Azad et al. (2013) is simple. After generation of the candidates, the unpenalized weights are calculated before performing the structural analyses. If the unpenalized weight of a candidate is greater than the penalized weight of the global best solution, it is thought that the mentioned candidate is not capable of improving the final result and removed from the candidates list. In this study, an additional rule is applied: Another candidate that meets the above mentioned criterion fills the place of the removed one. In the further text, UBS refers to this modified version. Table 9 presents the contribution of the UBS to the procedures that use Camp and Kaveh-Talatahari redistributions and CR bounding. It should be noted that the sub-optimization mechanism (SOM) of Kaveh and Talatahari's work is not considered. The α_1, α_2 and α_3 parameters are taken 1, 0.4 and 0.8, respectively.

Table 9. Effect of the UBS on the optimization results

Eq.	Best Weight (lb)	Average Weight (lb)	Worst Weight (lb)	Avg. No. of Analyses	C_v (%)
Camp (2007)	545.48	549.84	564.78	10047	0.00
Camp (2007) + UBS	545.40	547.31	560.99	9913	0.00
Kaveh and Talatahari (2009)	545.32	548.78	561.25	10874	0.00
Kaveh and Talatahari (2009) + UBS	545.43	547.61	565.30	9200	0.00

By utilizing the UBS, structural analysis count of the Camp's procedure is improved by 1.3%, which is a negligible ratio. However there is a 15.4% improvement in the Kaveh-Talatahari algorithm. Figure 3 presents the convergence history of the best solutions.

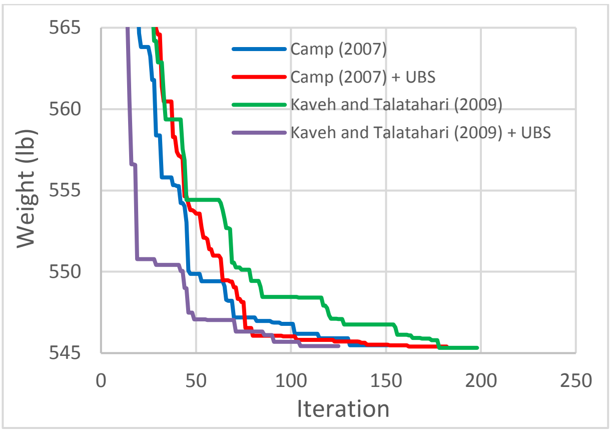

Figure 3. Contribution of the UBS to convergence

It is seen from Figure 3 that the Kaveh-Talatahari algorithm with UBS precedes the other procedures for this special case. The last test involves the definition of a new objective function as given in Eq. 9.

$$W_j = w_j\left(1 + \phi_{j,\sigma} + \phi_{j,\delta}\right)^{\frac{\eta(k+1)}{t}} \tag{9}$$

where η is a positive real number. Eq. 9 is a dynamic penalty function that gradually increases the penalty amount by the iteration count. This gives the infeasible solutions a "chance" to contribute to the exploration phase. Table 10 presents the results obtained by Camp's equation and CR bounding for the values of η from 1 to 5.

Table 10. Variation of the results obtained by Camp's equation and CR bounding with respect to η parameter.

η	Best Weight (lb)	Average Weight (lb)	Worst Weight (lb)	Avg. No. of Analyses	C_v (%)
1	545.38	550.62	576.06	9861	0.00
2	545.42	549.41	567.04	9539	0.00
3	545.39	549.50	563.72	9855	0.00
4	545.32	549.38	566.96	10818	0.00
5	545.52	549.14	569.05	10569	0.00

Table 10 shows that the best, average and worst values do not change considerably with η. However, $\eta = 2$ provides the best performance in the average number of structural analyses, which is less than the worst result by 11.8%. In addition, the presented dynamic penalty function requires 5.1% less calculations than the penalty function used by Camp to converge to a near-optimal design. The convergence histories for various η values are presented in Figure 4. Note that the member section areas of the best designs given in Tables 8, 9 and 10 are presented in Tables 11, 12 and 13 of the Appendix part, respectively.

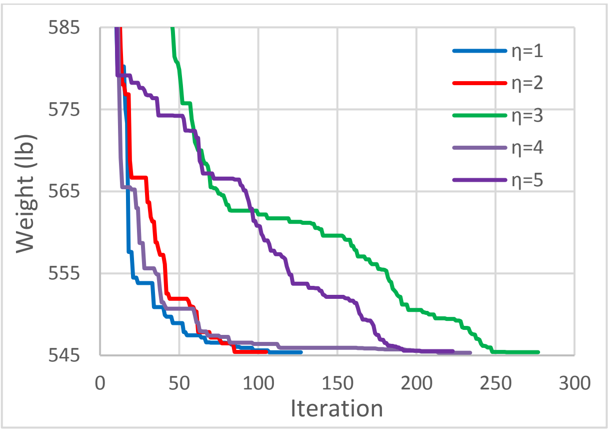

Figure 4. Effect of the η parameter on the best designs

It should be remembered that this study evaluates the contribution of the mentioned experimental modifications over a numerical example. It is not possible to certainly determine the superiority of one method over other methods in stochastic optimization. However, their convergence behavior can be assessed by comparing the average results of multiple runs.

5. Conclusions

In this study, size optimization of trusses with the BB-BC algorithm is revisited. First, various redistribution equations from the literature are summarized and findings of the authors are discussed. Then, the contributions of three experimental modifications to the BB-BC algorithm are presented over a spatial truss structure example. The effect of four introduced bounding methods as CC, CR, RC and RR are presented. It is found that the CR bounding requires about 18.8% less structural analyses than the worst bounding method to converge to a near-optimal solution. The modified UBS provides up to 15.4% decrease in the number of structural analyses. As the last improvement, presented dynamic penalty function provides 5.1% less structural analyses than the penalty function used by Camp. The ideal penalty parameter value for the dynamic function is determined as 2. It should be noted that all introduced procedures found nearly the same best, average and worst values. The significant difference is in the number of the structural analyses, which is an important measure in evaluating the performance of an optimization procedure.

References

Azad, S., Hasançebi, O., Azad S. and Erol, O., 2013. Upper Bound Strategy in Optimum Design of Truss Structures: A Big Bang-Big Crunch Algorithm Based Application. *Advances in Structural Engineering*, **16(6)**, 1035–1046.

Camp, C. V., 2007. Design of Space Trusses Using Big Bang–Big Crunch Optimization. *Journal of Structural Engineering*, **133(7)**, 999–1008.

Casavola, C., Lamberti, L. and Pruncu, C. I., 2012. Weight Minimization of Truss Structures with Big Bang-Big Crunch. *The 4th International Conference "Advanced Composite Materials Engineering", COMAT*, Brasov.

Erol, O. K. and Eksin, I., 2006. A New Optimization Method: Big Bang-Big Crunch. *Advances in Engineering Software*, **37(2)**, 106–111.

Hasançebi, O. and Azad, S. K., 2013. Reformulations of Big Bang-Big Crunch Algorithm for Discrete Structural Design Optimization. *International Journal of Civil, Environmental, Structural, Construction and Architectural Engineering*, **7(2)**, 139–150.

Hasançebi, O. and Azad, S. K., 2014. Discrete Size Optimization of Steel Trusses Using a Refined Big Bang–big Crunch Algorithm. *Engineering Optimization*, **46(1)**, 61–83.

Jain, R., Arun, L. and Ananthan, H., 2016. Visual Basic Interface for Size & Shape Optimization of Space Trusses Using BB-BC Algorithm. *International Journal of Engineering Research & Technology*, **5(3)**, 669–674.

Kaveh, A. and Talatahari, S., 2009. Size Optimization of Space Trusses Using Big Bang–Big Crunch Algorithm. *Computers & Structures*, **87(17–18)**, 1129–1140.

Kaveh, A. and Talatahari, S., 2010a. A Discrete Big Bang - Big Crunch Algorithm for Optimal Design of Skeletal Structures. *Asian Journal of Civil Engineering*, **11(1)**, 103–122.

Kaveh, A. and Talatahari, S., 2010b. Optimal Design of Schwedler and Ribbed Domes via Hybrid Big Bang–Big Crunch Algorithm. *Journal of Constructional Steel Research*, **66(3)**, 412–419.

Kaveh, A. and Zolghadr, A., 2012. Truss Optimization with Natural Frequency Constraints Using a Hybridized CSS-BBBC Algorithm with Trap Recognition Capability. *Computers & Structures*, **102–103**, 14–27.

Kaveh, A. and Mahdavi, V. R., 2013. Optimal Design of Structures with Multiple Natural Frequency Constraints Using a Hybridized BB-BC/Quasi-Newton Algorithm. *Periodica Polytechnica Civil Engineering*, **57(1)**, 27.

Lotfi, H. and Ghoddosian, A., 2015. Size and Shape Optimization of Two-Dimensional Trusses Using Hybrid Big Bang-Big Crunch Algorithm. *International Journal of Mechatronics, Electrical and Computer Technology*, **5(14)**, 1987–1998.

Milajić, A. and Beljaković, D., 2016. Multi-Objective Truss Optimization Using Different Types of the BB-BC Algorithm. *MATEC Web of Conferences*, **73**, 4012.

Milajić, A., Beljaković, D. and Barović, D., 2014. Optimum Truss Design Using Big Bang - Big Crunch Algorithm. *International Conference of Contemporary Achievements in Civil Engineering*, 447-453.

Özbaşaran, H., 2017. Experiments on Size Optimization of Trusses with BB-BC Algorithm. *International Conference on Mathematics and Engineering*, 130.

Appendix

Table 11. Section areas (in^2) of the designs presented in Table 8.

	Element Group	CC	CR	RC	RR
1	A_1	0.01000	0.02556	0.03588	0.02592
2	A_2~A_5	1.92878	2.03565	2.01196	2.03431
3	A_6~A_9	3.08260	2.90631	2.94261	2.90374
4	A_{10}~A_{11}	0.01645	0.01000	0.01055	0.01373
5	A_{12}~A_{13}	0.01000	0.01000	0.01235	0.01323
6	A_{14}~A_{17}	0.68065	0.67562	0.68762	0.67588
7	A_{18}~A_{21}	1.68883	1.67926	1.68091	1.68388
8	A_{22}~A_{25}	2.63960	2.69628	2.67241	2.69629
Weight (lb)		545.45	545.48	545.63	545.76

Table 12. Section areas (in^2) of the designs presented in Table 9.

	Element Group	Camp (2007)	Camp (2007) + UBS	Kaveh and Talatahari (2009)	Kaveh and Talatahari (2009) + UBS
1	A_1	0.02556	0.01000	0.01000	0.01000
2	A_2~A_5	2.03565	1.98440	1.98684	1.97498
3	A_6~A_9	2.90631	2.98593	2.96871	2.95763
4	A_{10}~A_{11}	0.01000	0.01000	0.01000	0.01000
5	A_{12}~A_{13}	0.01000	0.01000	0.01000	0.01000
6	A_{14}~A_{17}	0.67562	0.66585	0.70041	0.67894
7	A_{18}~A_{21}	1.67926	1.68743	1.68889	1.70875
8	A_{22}~A_{25}	2.69628	2.68556	2.64658	2.67123
Weight (lb)		545.48	545.40	545.32	545.43

Table 13. Section areas (in^2) of the designs presented in Table 10.

	Element Group	$\eta = 1$	$\eta = 2$	$\eta = 3$	$\eta = 4$	$\eta = 5$
1	A_1	0.01000	0.01000	0.01000	0.01000	0.01000
2	A_2~A_5	2.01873	1.99212	1.95109	1.97146	1.98653
3	A_6~A_9	2.94637	2.98586	3.02739	3.00861	2.96421
4	A_{10}~A_{11}	0.01000	0.01000	0.01000	0.01000	0.01000
5	A_{12}~A_{13}	0.01000	0.01000	0.01880	0.01000	0.02081
6	A_{14}~A_{17}	0.70226	0.67670	0.68369	0.66167	0.66792
7	A_{18}~A_{21}	1.67442	1.67896	1.69393	1.68639	1.69437
8	A_{22}~A_{25}	2.65145	2.67518	2.64918	2.68562	2.68772
Weight (lb)		545.38	545.42	545.39	545.32	545.52