UDK 629.5.023:004.421.2:519.863:519.24

OPTIMIZACIJA STRUKTURE S VIŠE FUNKCIJA CILJA – PREGLED POSTOJEĆIH METODA GENETSKOG ALGORITMA
MULTI-OBJECTIVE STRUCTURAL OPTIMIZATION – A REVIEW OF THE GENETIC ALGORITHM METHODS

Albert ZAMARIN – Jasmin JELOVICA – Marko HADJINA

Sažetak: *Mnogo je načina za poboljšanje brodske strukture, s ciljem zadovoljenja zahtjeva brodogradilišta i brodovlasnika. Načini se odnose na karakteristike brodske strukture, koje su u osnovi sukobljene na način da se poboljšanje jedne karakteristike ne može postići a da se ne pogorša neka druga. Te su karakteristike uobičajeno: troškovi, masa, pouzdanost, sigurnost, stabilitet. Cilj je pronalaženje kompromisa s obzirom na važnost pojedine karakteristike. U tom su smislu evolucijski algoritmi, kao optimizacijski alati, kod većine inženjerskih problema dokazali svoju valjanost, budući da uspješno upravljaju velikim brojem varijabli i ograničenja, a u posljednje vrijeme i funkcija cilja. Proces usavršavanja tih alata ostvaruje se hibridnim rješenjima koja spajaju najbolje od nekoliko različitih pristupa. Stvaranje jednog algoritma koji bi bio primjenjiv na sve probleme je nemoguće. U ovom je radu prikazano nekoliko važnijih pristupa i metoda genetskog algoritma s više funkcija cilja s ciljem usmjeravanja stručnjaka koji ulaze u područje strukturne optimizacije na one metode koje su dokazale svoju valjanost u praktičnim primjenama.*

Ključne riječi: - optimizacija
- genetski algoritam
- brodska konstrukcija

Abstract: *There are numerous means of enhancing ship structure in an attempt to satisfy both the shipyard and the ship owner requirements. These means are addressed as a ship's attributes and they are regularly contrary in meaning, such that the improvement in one cannot be achieved without making some other worse. Typically these are e.g. cost, weight, reliability, safety and stability, which can be minimized or maximized to improve the final design. A compromise obviously has to be made between them, depending upon the importance of each attribute. Evolutionary algorithms have proven their worthiness in a great variety of practical engineering problems, handling many variables, constraints and objectives. However, the need to improve them is always present. This has recently been done by hybridization, combining different approaches in order to get the best of them. To devise an algorithm which will be applicable to all problems is impossible. In this review, for the purpose of directing the experts entering in the field of structural optimization, only a few of the most important approaches and multi-objective algorithms have been presented.*

Keywords: - optimization
- genetic algorithm
- ship structure

1. UVOD

Brodograđevna industrija neprestano se susreće s povećanjem zahtjeva za izradu konkurentnih i inovativnih brodova. Preliminarna faza projektiranja broda trebala bi dati veći broj kvalitetnih projektnih alternativa koje bi omogućile lakše donošenje važnih odluka o projektu. Mnogo je načina za poboljšanje brodske strukture, s ciljem zadovoljenja zahtjeva brodogradilišta i brodovlasnika. Načini se odnose na atribute strukture, koji su u osnovi sukobljeni na način da se poboljšanje jedne karakteristike ili atributa ne može

1. INTRODUCTION

The shipbuilding industry faces a constant increase in demand for more competitive and innovative ships. The early stage of ship design consequentially requires better design alternatives that allow for ease in making important decisions about the project. There are numerous means of enhancing the ship structure in an attempt to satisfy both the shipyard and the ship owner requirements. These means are addressed as a ship's attributes and are regularly contrary in meaning, such that the improvement in one cannot be achieved without

postići a da se ne pogorša neka druga. Te su karakteristike uobičajeno: troškovi, masa, pouzdanost, sigurnost, stabilitet. Cilj je pronalaženje kompromisa s obzirom na važnost pojedine karakteristike. Brodska konstrukcija mora izdržati
globalno i lokalno opterećenje koje određuje dimenzije strukturnih elemenata, te stoga vrijednosti tih karakteristika ne mogu biti slobodno odabrane. Ta ograničenja zatim određuju strukturu koja se uopće može izvesti. Pri istraživanju mogućega strukturnog rasporeda atributi se mogu minimizirati ili maksimizirati, uglavnom korištenjem metode matematičkog programiranja, tj. optimizacije, dok ograničenja moraju biti nenegativna tj. moraju biti zadovoljena. Ti se atributi zatim nazivaju ciljevima i proces optimizacije traži određeni projekt koji je najbolji upravo prema njima, te se naziva optimalnim.
Optimizacijske metode općenito se dijele prema broju ciljeva i prema pristupu na klasične, ili evolucijske, s jednom ili više funkcija cilja. Klasične metode pretpostavljaju jedno rješenje koje vodi optimizacijski proces prema optimumu. Evolucijski algoritmi [1] vrlo se brzo razvijaju proteklih dvadeset godina, od rada Goldberga [2] koji je primijenio ideju genetskog algoritma. Kada se optimiraju višestruki i sukobljeni ciljevi, obično se primjenjuje skupni pristup za formiranje skalarne funkcije koji dovodi do problema optimizacije s jednom funkcijom cilja. Evolucijski algoritmi rade sa skupom rješenja pa imaju mogućnost određivanja potencijalno optimalnih rješenja, tzv. Pareto-optimalnog skupa. Optimizacijski proces funkcionira unutar projektnog prostora **X** koji je određen graničnim vrijednostima varijabli, te se može napisati:

making some other worse. Typically these are e.g. cost, weight, reliability, safety, stability, etc. A compromise obviously has to be made between them, depending upon the importance of each attribute. However, their values cannot be freely chosen, since the structure has to withstand the imposed local and global loads that restrict sizes of structural elements. Those limitations define the constraints that the structure has to satisfy in order to be feasible. To investigate the capabilities of suspected structural arrangement, attributes can be maximized or minimized using the methods of mathematical programming, also known as optimization, while the constraints have to be non-negative, i.e. satisfied. These attributes are then called the objectives and the optimization searches for particular designs which are the best according by those objectives. Optimization methods can be generally classified based on the number of objectives that they deal with and based on the approach as classical, evolutionary, single-objective or multi-objective. Classical methods are gradient-based, employing one solution to guide the search process towards optima. Evolutionary algorithms (EA) [1] were rapidly developed during the last two decades according to the work of Goldberg, who adopted the idea of the genetic algorithm [2]. When multiple, conflicting objectives are optimized, usually some aggregating approach is used to form a scalar function and to solve the resulting single-objective optimization problem. EA deals with a set of solutions and has the possibility of obtaining optimal trade-offs, the so-called Pareto-optimal set. The optimization problem operates within the design space **X** limited with variable bounds.

$$\begin{aligned} &\text{Min } \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),...,f_m(\mathbf{x}),...,f_M(\mathbf{x})),\\ &\text{s.t. } \mathrm{g}_j(\mathbf{x}) \geq 0, \quad j = 1,2,...,J;\\ &\quad \mathrm{h}_k(\mathbf{x}) = 0, \quad k = 1,2,...,K;\\ &\quad \mathrm{x}_i^{(L)} \leq \mathrm{x}_i \leq \mathrm{x}_i^{(U)}, \quad i = 1,2,...,I. \end{aligned} \tag{1}$$

gdje je *M* broj funkcija cilja, *J* broj nejednadžbi, *K* broj jednadžbi ograničenja, a *I* broj varijabli. Za rezultantni vektor za odluku $\mathbf{a} \in \mathbf{X}$ kaže se da je dominiran u odnosu na vektor $\mathbf{b} \in \mathbf{X}$ ($\mathbf{a} \prec \mathbf{b}$) ako i samo ako je [3];

having *M* objective functions, *J* inequality constraints, *K* equality constraints and *I* decision variables. A decision vector $\mathbf{a} \in \mathbf{X}$ is said to dominate a decision vector $\mathbf{b} \in \mathbf{X}$ (also written as $\mathbf{a} \prec \mathbf{b}$) if and only if [3];

$$\begin{aligned} &\forall i \in (1,M): f_i(\mathbf{a}) \leq f_i(\mathbf{b}) \wedge\\ &\exists j \in (1,M): f_j(\mathbf{a}) < f_j(\mathbf{b}) \end{aligned} \tag{2}$$

Za vektor odluke **a** se kaže da je nedominiran u odnosu na neki skup $\mathbf{X}' \subseteq \mathbf{X}$ ako i samo ako ne postoji vektor **a'** u $\mathbf{X}'$ koji dominira nad **a** ili $\nexists \mathbf{a}' \in \mathbf{X}' : \mathbf{a}' \prec \mathbf{a}$

Vektor **a** je Pareto-optimalan ako i samo ako je nedominiran u odnosu na **X**. Pareto-optimalni vektor ne može se poboljšati ni u jednom cilju bez pogoršanja barem jednoga drugog cilja.

The decision vector **a** is said to be non-dominated regarding a set $\mathbf{X}' \subseteq \mathbf{X}$ if and only if there is no vector **a**' in $\mathbf{X}'$ which dominates **a**, formally $\nexists \mathbf{a}' \in \mathbf{X}' : \mathbf{a}' \prec \mathbf{a}$

The decision vector **a** is Pareto-optimal if and only if **a** is non-dominated regarding **X**. Pareto-optimal decision vectors cannot be improved in any objective without causing a degradation in at least one other objective.

2. VIŠECILJNI EVOLUCIJSKI ALGORITMI

Jedna je od prednosti primjene genetskog algoritma (GA) pri rješavanju inženjerskih problema mogućnost pronalaska novoga projektnog principa koji je bio nepoznat upravo do te optimizacije, a nije se mogao otkriti iz same formulacije problema ili intuitivno. Stoga je moguće ponekad pronaći uzorak unutar varijabli za optimalno rješenje ili vidjeti neku vrijednost varijable koja se ponavlja u svim projektima bez obzira na cilj. U takvim su slučajevima klasične metode nekorisne, pogotovo ako se radi o diskretnim varijablama kao što je slučaj kod strukturne optimizacije broda gdje se koriste varijable kao što su debljina opločenja, dimenzije i razmak ukrepa. Upravo su u [4] na nekoliko optimizacija s dvije funkcije cilja autori pokazali mogućnost pronalaska novoga projektnog principa istražujući dobiveni spektar optimalnih rješenja.
Budući da mnogi praktični problemi uključuju nekoliko međusobno suprotstavljenih ciljeva, ne postoji idealna metoda koja bi omogućila dobivanje skupa optimalnih projekata. To upravo proizlazi iz prirode pojave višestruko suprotstavljenih interesa ili ciljeva, gdje ne postoji globalno optimalno rješenje, već čitav niz rješenja s Pareto-fronte koja čine jednako dobre projekte. Za dobivanje rješenja s Pareto-fronte u klasičnim se metodama koristi princip težinskih faktora. Bolji način rješenja višeciljnoga optimizacijskog problema je korištenje algoritama i principa koji će dati širu Pareto-frontu budući da evolucijski algoritmi koriste populaciju rješenja u svakom optimizacijskom prolazu unutar procesa optimizacije. Cilj je približiti se što više pravoj Pareto-fronti i istodobno dobiti što više rješenja. Na taj se način osigurava osobi koja će donijeti odluku veći izbor kvalitetnih rješenja pri čemu ima i bolji pregled na sve moguće optimalne projekte.

2.1. Nedominirani sortirajući genetski algoritam (NSGA i NSGA II)

Oba navedena algoritma NSGA i NSGA II razlikuju se od običnog GA [2] samo u načinu rada operatora selekcije, dok su operatori križanja i mutacije ostali isti. NSGA se koristi metodom rangiranja selekcije pri čemu se naglasak selektiranja stavlja na nedominirana rješenja. Uz tu metodu koristi se i metoda zajedničke funkcije, koja služi za održavanje raznolikosti unutar populacije. Prije nego se izvrši selekcija izvode se dvije procedure: najprije se rangira populacija na osnovi individualne nedominirnosti, a zatim se koristi zajednička funkcija za pridjeljivanje dobrote svakoj jedinki [5].

2. MULTI-OBJECTIVE EVOLUTIONARY ALGORITHMS (MOEA)

When using GAs for engineering design problems, one can benefit from the possible discovery of innovative design principles that were not known until a particular optimization was made and that were difficult to discover from problem formulation or from intuition. It is sometimes possible to find a "pattern" in the decision variables for the optimal solutions or to see if some variables take the same values for the whole range of best designs with respect to each objective. In such cases, the classical gradient-based methods are useless, and even more so if discrete values for variables are used, such as in the case of ship structural optimization where one is dealing with plate thicknesses, stiffener sizes or spacing. In [4] the author demonstrated that by using a GA it is possible to discover innovative design principles by investigating the spectrum of obtained optimal solutions.
Since many practical problems involve several conflicting objectives, there is no ideal classical method to obtain a satisfactory set of optimal designs. This is due to the natural behaviour of multiple conflicting objectives, where no solution is globally optimal, but instead, many solutions form a Pareto-front of "equally good" designs. To obtain a few solutions which belong to the Pareto-front using classical methods, several attempts should be taken using different e.g. weight factors. A better way to solve the multi-objective optimization problem is to use an algorithm which will obtain a wider part of the Pareto-front, since evolutionary algorithms use a population of solutions in a single optimization run. The goal is to come as close as possible to the true Pareto-front and to obtain a wide spread of solutions at the same time. This ensures greater opportunity for the decision-maker to choose the preferred solution and to have a better overview of all of the possible optimal designs.

2.1. Non-dominated sorting genetic algorithms (NSGA and NSGA-II)

Both NSGA and NSGA-II differ from Goldberg's simple GA [2] just in the way the selection operator is used. The crossover and mutation operator remain the same. NSGA uses a ranking selection method to create selection pressure on non-dominated solutions and the sharing function method is used to maintain diversity within the population. Before the selection is performed, two procedures are performed serially. First, the population is ranked on the basis of an individual's non-dominated level and then the sharing function method is used to assign fitness to each individual [5].

Da bi se pronašla nedominirana rješenja, svako se rješenje iz populacije uspoređuje sa svakim drugim rješenjem u smislu vrijednosti cilja. Nakon što je identificirana prva razina nedominiranosti, sva rješenja koja je čine privremeno se odbacuju radi identifikacije druge razine nedominiranosti. Procedura se ponavlja sve dok sva rješenja nisu svrstana u neku razinu nedominiranosti. To znači da postoji od jedan do N razina nedominiranosti (N je veličina populacije). Vrijednost dobrote dodjeljuje se svakoj jedinki unutar ranga nedominiranosti na način da niža razina dobiva veću vrijednost. Tako je težište selekcije stavljeno na jedinke niže razine budući da su one bliže pravom Pareto-optimalnom području. Dodjeljivanje dobrote podijeljeno je u dvije faze, od kojih svaka faza vodi računa o posebnim ciljevima unutar višeciljne optimizacije. Najprije je identična proizvoljna vrijednost dobrote dodijeljena svim rješenjima koja pripadaju istoj nedominiranoj razini. Nakon toga je izdvojeno pojedinačno rješenje na osnovi gomilanja rješenja na fronti pomoću strategije zajedničke funkcije.

Detaljnije, za vrijednost dobrote za sva rješenja prve nedominirane fronte postavlja se vrijednost veličine populacije, tako da niti jedno rješenje unutar čitave populacije ne može imati veću vrijednost. Ako rješenje ima mnogo susjednih točaka na istoj fronti, strategija zajedničke funkcije određuje smanjenje vrijednosti dobrote pomoću faktora redukcije. Taj faktor ovisi o broju i blizini susjednih rješenja. Proračunata reducirana zajednička dobrota rješenja iz određene nedominirane razine neće nikad biti manja od proizvoljno odabrane vrijednosti dobrote rješenja koja pripada višoj razini. Kada sva rješenja određene razine dobiju vrijednosti dobrote, razmatra se sljedeća razina koja također dobiva zajedničku dobrotu i tako dok sva rješenja svih razina ne dobiju takvu zajedničku vrijednost.

Procedura zajedničke dobrote za skup od n_k rješenja unutar k-te nedominirane fronte, od kojih svako ima proizvoljnu vrijednost dobrote f_k za svako se rješenje $i = 1,2,\ldots, n_k$ odvija kroz sljedeće korake:

Korak 1: Proračun normalizirane euklidske mjere udaljenosti i-tog i j-tog rješenja unutar k-te nedominirane fronte:

Every solution from the population is compared with all others regarding their objective values in order to find those which are non-dominated. Once the first level of non-dominated solutions is identified, solutions that constitute it are temporarily disregarded in the search for the second non-dominated level. This procedure is continued until all solutions are classified into a certain level of non-dominance. There can be from one to N different non-dominated levels (N being population size). Fitness value is assigned to each individual based on its non-domination level. An individual in a lower level gets higher fitness. This way, selection pressure is put on lower-level individuals, since they are closer to the true Pareto-optimal region. Thus, fitness assignment is divided into two stages, each taking care of separate multi-objective optimization goals. First, an identical dummy fitness is assigned to all solutions belonging to the same non-domination level. Thereafter, based on crowding of solutions in the front, single solutions are emphasized by using a sharing function strategy.

In more detail, fitness equal to the population size is assigned to all solutions from the first non-dominated front. No solution within a whole population can have higher fitness than this. Sharing function strategy determines that if a solution has many neighbouring points in the same front, its dummy fitness will be reduced by a factor called *niche count*, which depends on number and proximity of neighbouring solutions. Nevertheless, this shared fitness of a solution from a certain non-dominated level will never be smaller than any dummy fitness value from the solution belonging to a higher non-dominated level. Once all the solutions from a certain non-dominated level are assigned to sharing fitness values, the next level is considered, also receiving shared fitness. This procedure is continued until all solutions are assigned a shared fitness. Sharing procedure for a set of n_k solutions in the k-th non-dominated front, each having a dummy fitness value f_k, is performed in the following way for each solution $i = 1,2,\ldots, n_k$.

Step 1: Compute a normalized Euclidean distance measure with another solution j in the k-th non-dominated front, as follows:

$$d_{ij} = \sqrt{\sum_{p=1}^{P} \left(\frac{x_p^{(i)} - x_p^{(j)}}{x_p^{u} - x_p^{l}} \right)^2}, \tag{3}$$

gdje P označava ukupan broj promatranih varijabli. Parametri x_p^u and x_p^l predstavljaju gornju i donju granicu varijable x_p.

where P denotes the number of variables in the problem. The parameters x_p^u and x_p^l are the upper and lower bounds of variable x_p

Korak 2: Udaljenost d_{ij} uspoređuje se s prethodno definiranim parametrom σ_{share}, i proračunava se sljedeća zajednička funkcija:

$$Sh\left(d_{ij}\right)=\begin{cases}1-\left(\dfrac{d_{ij}}{\sigma_{share}}\right)^2, & \text{if } d_{ij}\le\sigma_{share}\\ 0, & \text{otherwise.}\end{cases} \tag{4}$$

Step 2: The distance d_{ij} is compared to a pre-specified parameter σ_{share}, and the following *sharing function* is computed:

Korak 3: Povećaj *j*.
Ako je $j\le n_k$, idi na Korak 1 i proračunaj $Sh\left(d_{ij}\right)$, a ako je $j>n_k$,proračunaj redukcijski faktor *i*-tog rješenja:

$$m_i=\sum_{j=1}^{n_k}Sh\left(d_{ij}\right). \tag{5}$$

Step 3: Increment *j*.
If $j\le n_k$, go to Step 1 and calculate $Sh\left(d_{ij}\right)$.
If $j>n_k$, calculate niche count for *i*-th solution as:

Korak 4: Umanji proizvoljnu vrijednost dobrote f_k *k*-te nedominantne fronte za *i*-to rješenje i proračunaj reduciranu zajedničku dobrotu kao:

$$f_i'=\frac{f_k}{m_i}. \tag{6}$$

Step 4: Degrade the dummy fitness value f_k of the *k*-th non-dominated front for the *i*-th solution to calculate the shared fitness:

Najmanja zajednička vrijednost dobrote od svih rješenja koje pripada *k*-toj fronti f_k^{min} još se smanjuje za mali pozitivni broj ε_k pa je tako definirana proizvoljna dobrota za sljedeću frontu kao $f_{k+1}=f_k^{\min}-\varepsilon_k$

The smallest share fitness value from all solutions belonging to the *k*-th front, f_k^{min}, is reduced by the small positive number ε_k and a dummy fitness for the following front is obtained as $f_{k+1}=f_k^{\min}-\varepsilon_k$.

Navedena procedura zahtijeva i parametar σ_{share}, ,[5] koji zapravo predstavlja najveću udaljenost između bilo kojih dvaju rješenja koja dijele svoje dobrote:

$$\sigma_{share}\approx\frac{0,5}{\sqrt[P]{q}}, \tag{7}$$

The sharing procedure described above requires a parameter σ_{share}, [5] that denotes the largest value of that distance metric within which any two solutions share each other's fitness:

gdje je *q* broj željenih različitih Pareto-optimalnih rješenja na kraju optimizacijskog procesa. Mnogi višeciljni evolucijski algoritmi, među kojima i NSGA, imaju nedostatke u smislu proračunske složenosti, pristupa koji ne uključuje elitizam i potrebu za određivanjem zajedničkog parametra σ_{share}, koji se u tradicionalnim procedurama koristi za osiguranje raznolikosti populacije, ali se mora odrediti unaprijed. Otklanjanje nedostataka rezultiralo je novijom poboljšanom verzijom algoritma NSGA-II. Proračunsko vrijeme smanjeno je s $O(MN^3)$ na $O(MN^2)$ na osnovi boljeg čuvanja podataka nedominantnih projekata. Koncept elitizma također može znatno ubrzati performanse GA [3]. Zajednički parametar potreban u NSGA zamijenjen je novijom metodom procjene gustoće. NSGA uspoređuje svako rješenje sa svim drugim rješenjima unutar populacije za sve *M* funkcije cilja. Kada je svih *N* članova populacije provjereno, može biti maksimalno $O(MN^3)$ usporedbi da bi se pronašao nedominirani projekt [6]. Kod NSGA-II svako rješenje donosi nedominirani skup rješenja *P'* koji je sastavljen od članova tekuće populacije *P*. Ako je neki član te skupine dominiran, uklanja se iz *P'*, inače su

where *q* is the desired number of different Pareto-optimal solutions at the end of the optimization process. NSGA and other multi-objective evolutionary algorithms that use non-dominated sorting and sharing have been mainly criticized [6] for their computational complexity, non-elitist approach and the need for determining a sharing parameter which is used in traditional procedures that ensure diversity in a population, but it is required that it be determined before the optimization process starts.
The criticism of NSGA has lead to the development of a new and improved version of this algorithm, called NSGA-II. Computational time is decreased from $O(MN^3)$ to $O(MN^2)$, based on better bookkeeping of non-dominated designs. The concept of elitism can significantly speed up the performance of the GA [3]. The sharing parameter required by NSGA is bypassed by the new density estimation. NSGA compares each solution with every other solution in the population, for all *M* objective functions. When all *N* population members are checked, there can be up to $O(MN^3)$ comparisons to find the non-dominated designs [6]. In NSGA-II, each solution is entering non-dominated set of solutions *P'* composed of members from the current

rješenja dio nedominirane fronte i zadržavaju se u skupu. Procedura započinje tako što se prvi projekt u populaciji proglašava članom nedominiranog skupa. Tako se drugi projekt uspoređuje samo s jednim rješenjem, treći projekt s najviše dva rješenja itd. Radi izbjegavanja problema koji nastaje upotrebom parametra σ_{share}, umjesto pristupa zajedničke funkcije primijenjen je pristup usporedbe gustoće. Taj novi pristup procjenjuje gustoću rješenja koja okružuju određeno rješenje preko izračuna prosječne udaljenosti dviju točaka na objema stranama promatranog rješenja, a za sve funkcije cilja, slika 1. Rezultat te procjene naziva se udaljenost gustoće i daje veličinu najvećeg kuboida koji zatvara točku i bez uključivanja bilo koje druge točke populacije. Sva rješenja sortirana su uzlaznim redom u odnosu na vrijednost funkcije cilja. Pretpostavljajući da ima l rješenja u određenoj nedominiranoj fronti, procedura za proračun udaljenosti gustoće $L_{distance}$ je:

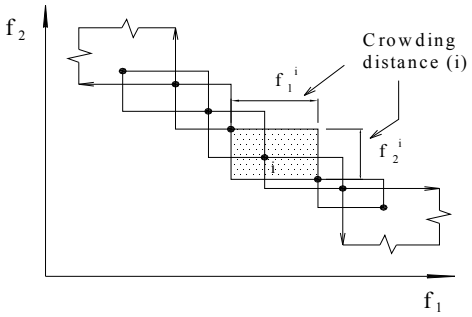
Korak 1: $I_{distance}^{(1)} = I_{distance}^{(l)} = \infty$

Korak 2:
$$\text{for } i = 2, l-1$$
$$\text{for } m = 1, M$$
$$I_{distance}^{(i)} = I_{distance}^{(i)} + \left(I_m^{(i+1)} - I_m^{(i-1)} \right)$$

population P. If any member of this group dominates it, it is removed from P'. Otherwise, it is also a member of the non-dominated front and is kept in this set, while the solution or solutions, which it dominates, are removed from P'. The procedure starts by assigning the first design in a population to be a member of the non-dominated set. Thus, the second design is compared only with one solution, the third with at most two designs, etc. To avoid problems which come through usage, the parameter σ_{share}, crowded comparison approach is implemented instead of the sharing function approach. This new approach estimates the density of solutions surrounding a particular solution by calculating the average distance of two points on either side of the observed solution for all M objective functions, as seen in Figure 1. The value of this estimation, called the *crowding distance*, gives the size of the largest cuboids enclosing the point i without including any other point in the population. Presuming there are l solutions in a certain non-dominated front, the crowding distance assignment $L_{distance}$ computation procedure is:

Step 1: $I_{distance}^{(1)} = I_{distance}^{(l)} = \infty$

Step 2:
$$\text{for } i = 2, l-1$$
$$\text{for } m = 1, M$$
$$I_{distance}^{(i)} = I_{distance}^{(i)} + \left(I_m^{(i+1)} - I_m^{(i-1)} \right)$$



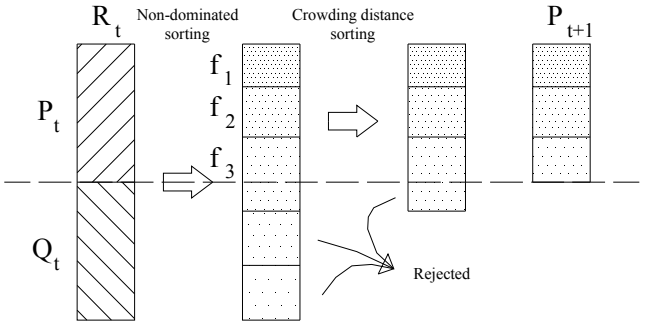
Slika 1 Proračun udaljenosti gustoće [6]
Figure 1. The crowding distance calculation [6]

Glavna procedura NSGA-II (slika 2) je sljedeća; početna slučajno odabrana populacija P_0 veličine N sortira se prema nedominiranosti. Prva generacija Q_0 kreira se korištenjem binarnoga normalnog odabira, rekombinacije i mutacije. Nakon toga procedura se mijenja. Uvodi se elitizam tako da se uspoređuje tekuća populacija s prethodno pronađenim najboljim nedominiranim rješenjem. Nakon prve generacije procedura je sljedeća: stvara se unija $R_t = P_t \cup Q_t$ veličine $2N$ i sortira na osnovi nedominiranosti. Najbolja nedominirana fronta transformira se u novu dječju populaciju dok ne dostigne N članova. Ako je slučajno određena fronta veća nego što ima mjesta u populaciji, uzimaju se oni članovi koji imaju veći $L_{distance}$. Za odabir rješenja za novu populaciju Q_{t+1} koristi se binarni odabir zasnovan na rangu

The main procedure of NSGA-II, Figure 2, is as follows: initial random population P_0 of size N is sorted according to non-domination. Child population Q_0 is created using normal binary tournament, recombination and mutation. After this, the procedure is changed. Elitism is introduced by comparing the current population with the previously determined best non-dominated solutions. The procedure after the first child population has been made is as follows: create union $R_t = P_t \cup Q_t$ of size $2N$ and sort it based on non-domination. The best non-dominated fronts are transferred to the new child population P_{t+1}, until it has N members. If a certain front is bigger than the required population slots, only members having the greatest $L_{distance}$ are taken. The binary tournament selection based on non-domination rank and crowding

nedominiranosti i udaljenosti gustoće, a zatim slijedi jednostavno križanje i mutacija.

distance is used for selecting solutions for a new population Q_{t+1}, followed by simple crossover and mutation.



Slika 2 NSGA-II [6] shema
Figure 2. A sketch of NSGA-II [6]

2.2. Čvrsti Pareto evolucijski algoritam 2 (SPEA2)

Zitzler i ostali [7] pokazali su znatno poboljšanje rada višeciljnih evolucijskih algoritama u kojima se primjenjuje pristup elitizma. Proces selekcije kod SPEA provodi se korištenjem pristupa klasterizacije. Budući da ukupna veličina populacije $\left|P_t \cup \overline{P_t}\right| = N'$ nije konstantna, formira se klaster $\left|\overline{P_t}\right| = N$ unutar svake generacije *t*.
Pretpostavlja se da su članovi populacije odvojeni klaster, a zatim se računaju sve euklidske udaljenosti. Dva klastera s najmanjom udaljenosti se spajaju, a koristi se prosječna udaljenost tako sparenih udaljenosti između rješenja dvaju klastera. Procedura se koristi dok veličina populacije ne dođe na *N*. Zatim se za parenje odabire jedan slučajno odabran projekt iz svakoga klastera. SPEA2 koristi poboljšanu shemu dodjeljivanja dobrote uzimajući u obzir za svaku jedinku je li dominantna ili dominirana, što govori o snazi jedinke. Nadalje, različitost rješenja garantira se tehnikom procjene najbliže gustoće koja izračunava udaljenost do *k*-te najbliže točke u ciljnom prostoru, čime se određuje pripadnost gušćem ili rjeđem području. Očuvanje graničnih rješenja postiže se metodom arhivskog reza.
SPEA2, za razliku od SPEA, za određivanje nedominiranih rješenja koristi fiksnu veličinu arhiva. Kada je broj nedominiranih projekata manji od zahtijevane veličine arhiva, on se puni dominiranim rješenjima na osnovi funkcije dobrote. U suprotnome poziva se funkcija arhivskog reza koja iterativno uklanja jedinke iz arhiva dokle god arhiv ne dođe na svoju prethodno definiranu veličinu [7]. U određenoj generaciji samo najbolja rješenja koja se nazivaju *arhivi* sudjeluju u procesu odabira za parenje. Nakon što se primijene GA operatori, uspoređuje se nova i roditeljska populacija kako bi se stvorio novi arhiv.
NSGA, NSGA-II, SPEA ili SPEA2 imaju nedostatke: dobro raspoređene skupove rješenja mogu odrediti ili uz velik utrošak vremena ili uz premalu raznolikost rješenja.

2.2. Strength Pareto Evolutionary Algorithm 2 (SPEA2)

Zitzler *et al.* [7] demonstrated that using the elitism approach causes MOEA to perform significantly better. The selection process in SPEA is performed using the clustering approach. Since the overall population size $\left|P_t \cup \overline{P_t}\right| = N'$ is not constant, $\left|\overline{P_t}\right| = N$ clusters are formed in each generation *t*. Population members are assumed to be a separate cluster, and all Euclidean distances are computed. Two clusters with the smallest distance are merged together and the average distance of all pair-wise distances between solutions of the two clusters is used. This procedure is used until population size reduces to *N*. From each cluster, one random design is selected for pairing. SPEA2 uses the improved fitness assignment scheme that takes into account for each individual the number of individuals that it dominates and that it is dominated by, which is said to be the strength of an individual. Furthermore, diversity among solutions is guaranteed by the nearest density estimation technique that calculates distance to the *k*-th nearest data point in the objective space, showing whether it is in the high or low-density region. Preservation of boundary solutions is accomplished by the archive truncation method. Contrary to SPEA, SPEA2 uses fixed archive size for non-dominated solutions. Whenever the number of non-dominated designs is smaller than the required archive size, it is filled up by dominated solutions, based on their fitness value. Otherwise, the archive truncation procedure is invoked, which iteratively removes individuals from the archive until it reaches its pre-defined size [7]. In certain generation of optimization, only the best solutions, called the archive, participate in the mating selection process. After the GA operators have been applied, offspring and parent populations are compared to form a new archive. NSGA, NSGA-II, SPEA or SPEA2, are either able to find a well-distributed set of solutions at the expense of a large computational

Rješenje problema ponudio je Deb i ostali [8] s prijedlogom MOEA na osnovi koncepta ε-dominiranosti.

2.3. ε-MOEA

Ova metoda koja se zasniva na ε-dominiranosti dijeli cijeli projektni prostor na određen broj hiperkutija ili mrežu, gdje se raznolikost održava na način da svaka hiperkutija može sadržavati samo jedno rješenje. Populaciji arhiva $\overline{P}_t$ pridružuje se ε-nedominirano rješenje iz populacije P_t veličine $2N$, pa se svaki par sastoji od člana iz $\overline{P}_t$ i P_t. Za odabir rješenja iz P_t, slučajno se izvlače dvije jedinke a i b, pa ako je $a \prec b$ odabire se prva, a ako je $a \not\prec b \wedge b \not\prec a$ tada se jedinka odabire slučajno. Budući da se nadalje može koristiti nekoliko strategija, rješenje iz $\overline{P}_t$ također se slučajno odabire [8]. Novo rješenje o_n uspoređuje se sa svakim članom p_n iz $\overline{P}_t$. Svakom se rješenju pridružuje identifikacijski red $\mathbf{B} = (B_1, B_2, ..., B_M)^{\mathrm{T}}$ kako slijedi:

effort, or have less diversity. Deb *et al.* [8] suggested MOEA based on the ε-dominance concept.

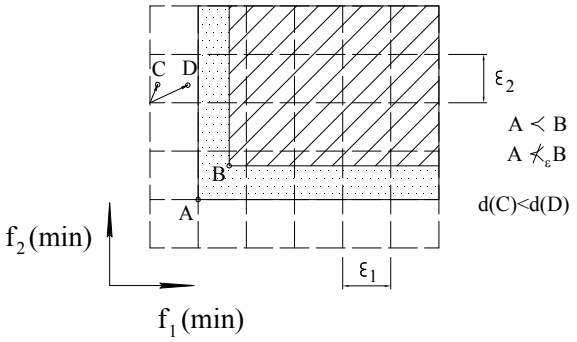
2.3. ε-MOEA

The proposed ε-dominance concept divides the search space into a number of hyper-boxes or grids and the diversity is maintained by ensuring that a hyper-box can be occupied by only one solution. The archive population $\overline{P}_t$ is assigned with ε-non-dominated solutions from P_t of the size $2N$ and each mating pair is composed of a member from $\overline{P}_t$ and P_t. To choose a solution from P_t, two individuals a and b are picked at random, and if $a \prec b$ the former one is selected, or if $a \not\prec b \wedge b \not\prec a$ then one is picked at random. Although several strategies can be used, the required solution from $\overline{P}_t$ is also randomly chosen [8]. An offspring o_n is compared with each member p_n of $\overline{P}_t$ for ε-dominance. Each solution is assigned an identification array $\mathbf{B} = (B_1, B_2, ..., B_M)^{\mathrm{T}}$:

$$B_m(\mathrm{f}) = (f_m - f_m^{\min}) / \varepsilon_m \tag{8}$$

gdje je f_m^{min} minimalna dobivena vrijednost, a ε_m je dozvoljena tolerancija m-tog cilja, slika 3. Korisnik definira vektor $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_M)^{\mathrm{T}}$ koji određuje korak u smjeru svakog cilja za željenu razdiobu rješenja. Naprimjer, kod optimizacije mase strukture broda i vertikalnog položaja težišta može se postaviti $\boldsymbol{\varepsilon} = (100, 0.05)^{\mathrm{T}}$, što pretpostavlja da će željena rješenja mase biti raspodijeljena u koraku od po 100t, a položaj težišta po 5 cm.

where f_m^{min} is the minimum obtained value of the m-th objective and ε_m is the allowable tolerance in the m-th objective, as shown in Figure 3. The user defines a vector $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_M)^{\mathrm{T}}$ that gives a step in the direction of each objective for the desired distribution of solutions. For instance, if the minimization of ship structural weight and VCG is performed, one can set $\boldsymbol{\varepsilon} = (100, 0.05)^{\mathrm{T}}$, denoting that desired solutions should be distributed in the step of 100 t and the position of VCG of 5 cm.



Slika 3. Koncept ε-dominacije
Figure 3. The ε-dominance concept

Ako je u slučaju minimizacije $\mathbf{B}_{p_n} < \mathbf{B}_{o_n}$ onda $p_n \prec_\varepsilon o_n$. Znači da su nova rješenja o_n ε-dominirana od nekog člana $\overline{P}_t$, te se stoga odbacuju. Inače, ako je $o_n \prec_\varepsilon p_n$ onda o_n zamjenjuje ε-dominirano rješenje iz $\overline{P}_t$. U trećem slučaju, kada je $o_n \not\prec_\varepsilon p_n \wedge p_n \not\prec_\varepsilon o_n$, onda se provjerava uobičajena nedominiranost i usvaja se ono rješenje koje ima manju euklidsku udaljenost prema vektoru **B**. Tako procedura osigurava postojanje samo jednog rješenja u nekoj hiperkutiji. Opći je zaključak,

If for minimization cases $\mathbf{B}_{p_n} < \mathbf{B}_{o_n}$, then $p_n \prec_\varepsilon o_n$, meaning that the offspring o_n is ε-dominated by some member of $\overline{P}_t$ and is rejected. Otherwise if $o_n \prec_\varepsilon p_n$ than o_n replaces ε-dominated solution from $\overline{P}_t$. In the third case, when $o_n \not\prec_\varepsilon p_n \wedge p_n \not\prec_\varepsilon o_n$ then they are checked for the usual non-dominance and are processed accordingly. If they are non-dominated in relation to each other even in the usual sense, then the one having smaller Euclidean distance to the **B** vector is accepted. This

nakon provedenog testiranja, da ε-MOEA procedura dobro konvergira i proizvodi dobru raznolikost i to za barem jedan red veličine manje utrošenoga proračunskog vremena u odnosu na ostale MOEA procedure. SPEA2 izvodi se najbolje u smislu raznovrsnosti, ali na račun velikog utroška proračunskog vremena i to za tri do četiri puta veće od ε-MOEA. Raznolikost rješenja kod ε-MOEA lošija je negoli kod SPEA2 jer ne obuhvaća rubna rješenja, a što proizlazi iz same definicije ε-dominiranosti.

procedure ensures that only one solution exists in some hyper-box. The general conclusion after performing all the tests is that ε-MOEA produces a good convergence and diversity with at least an order of magnitude smaller computational time than the rest of MOEAs. SPEA2 performs the best in terms of diversity, but at the expense of large computational requirements, three to four orders of magnitude larger than the ε-MOEA. The diversity of solutions from ε-MOEA is somewhat worse than SPEA2 because it does not obtain boundary solutions, due to the definition of ε-dominance.

2.4. Hibridni pristup lokalna pretraga

Evolucijski algoritmi rade s populacijom rješenja i u svakoj generaciji iskorištavaju dobivena nedominirana rješenja. U završnoj fazi optimizacije populacija se sastoji od rješenja koja su blizu Pareto-fronte. To je princip rada NSGA-II [6], SPEA2 [7], ε-MOEA [8]. Kako bi se omogućilo korisniku približavanje stvarnoj Pareto-optimalnoj fronti, a istodobno smanjila veličina dobivenog skupa nedominiranih rješenja, hibridni pristup [9] predlaže kombinaciju evolucijskog algoritma i lokalne pretrage. Glavna je ideja da MOEA pronađe skup rješenja
blizu stvarne Pareto-fronte, a da zatim tehnika lokalne pretrage pomogne bržoj konvergenciji prema pravoj Pareto-optimalnoj fronti, koristeći se različitim naglaskom na funkcije cilja. Za lokalnu pretragu uvodi se težinska funkcija cilja:

2.4. A hybrid approach for local search

Evolutionary optimization algorithms work with a population of solutions and in each generation exploits the obtained non-dominated solutions.
In the final stage of optimization, the population consists of solutions which are close to Pareto-front. That is the principle of NSGA-II [6], SPEA2 [7], and ε-MOEA [8]. To enable a user to move closer to the true Pareto-optimal front and simultaneously reduce the size of the obtained non-dominated solution set, [9] proposes a hybrid method that combines the multi- objective evolutionary algorithm and a local search. The overall idea is that once MOEA finds a set of solutions close to the true Pareto-optimal region, a local search technique is used from each of these solutions with a different emphasis of objective functions in the hope of better converging to the true Pareto-optimal front. To perform a local search, a weighted objective function is employed:

$$F(\mathbf{x}) = \sum_{m=1}^{M} w_m{}^{\mathbf{x}} \cdot f_m(\mathbf{x}), \tag{9}$$

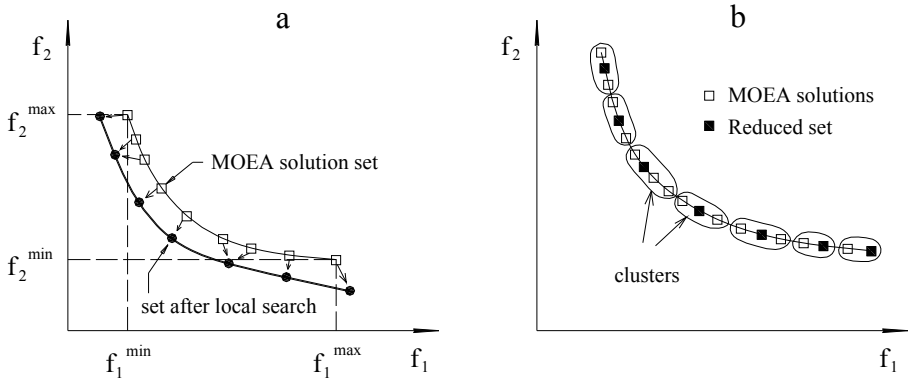
za koju se težina može izračunati nakon određivanja miminimalne $f_m{}^{min}$ i maksimalne $f_m{}^{max}$ vrijednosti svake funkcije cilja f_m.

for which the weights can be calculated after determining the minimum $f_m{}^{min}$ and the maximum $f_m{}^{max}$ values of each objective function f_m

$$w_m{}^{\mathbf{x}} = \frac{(f_m{}^{\max} - f_j(\mathbf{x}))/(f_m{}^{\max} - f_m{}^{\min})}{\sum_{k=1}^{M}(f_k{}^{\max} - f_k(\mathbf{x}))/(f_k{}^{\max} - f_k{}^{\min})}. \tag{10}$$

Tako projekt koji je dobar za neku funkciju cilja f_m dobiva visoku težinsku vrijednost. Kada su težinski faktori određeni, počinje lokalna pretraga za svaki **x** neovisno o svrsi optimizacije (slika 4). Rješenja iz lokalne pretrage mogu biti dominirana, te ih je stoga potrebno eliminirati.

Therefore, a design that is good for a specific objective f_m receives a high weight factor value. Once weight factors are calculated, the local search begins with each **x** independently of the purpose of optimizing as illustrated in Figure 4. The solutions from the local search can dominate each other, so it is necessary to eliminate those that are dominated.



Slika 4 Tehnika lokalnog pretraživanja, a) metoda klasterizacije, b) hibridna metoda [9]
Figure 4. The local search technique, a) clustering method, b) hybrid method [9]

3. OMNI-OPTIMIZACIJA

Jedna od poteškoća pri rješavanju realnih optimizacijskih problema je činjenica da se problemi javljaju u raznolikom broju formi i tipova. Neki se problemi rješavaju prema samo jednom cilju, neki imaju čvrsta ograničenja, a neki mogu imati više od jednoga optimalnog rješenja. Kada se korisnik suoči s problemom, najprije ga mora analizirati, a tek onda odabrati prikladan algoritam za njegovo rješavanje. Razlog tome je što algoritam efikasan za pronalaženje optimuma kod optimizacijskog problema s jednom funkcijom cilja ne mora nužno biti efikasan i primjenljiv za pronalaženje višestrukih optimalnih rješenja prisutnih kod drugoga problema. Stoga, za rukovanje različitim tipovima problema korisnik mora poznavati različite algoritme od kojih je svaki specijaliziran za pojedinu klasu optimizacijskog problema.

3.1. NSGA-II kao omni-optimizator

Za rješenje navedenog problema predlaže se algoritam, [10] koji je sposoban rukovati s jednom i više funkcija cilja. U osnovi to je NSGA-II s dodatkom promjenljive udaljenosti gustoće.
Kod problema s jednom funkcijom cilja procedura je sljedeća: provjera dominacije između dva rješenja degenerira u traženju boljeg rješenja u vrijednost cilja i u slučaju kašnjenja odabire se različitije rješenje na osnovi varijable udaljenosti gustoće. Predloženi algoritam koristi dva pristupa za procjenu gustoće rješenja; gustoća u ciljnom prostoru i gustoća u projektnom prostoru. Dakle, ako dva rješenja imaju istu vrijednost cilja njihova će udaljenost gustoće u ciljnom prostoru biti ista, ali će se onda odabrati rješenje koje ima veću vrijednost gustoće udaljenosti u projektnom prostoru. Algoritam proračunava obje udaljenosti gustoće za svaki projekt, a za usporedbu se koristi viša vrijednost.
Kod rješavanja problema s više funkcija cilja, omni-optimizacijska procedura je slična onoj kod NSGA-II, ali

3. OMNI-OPTIMIZATION

One of the difficulties in solving real-world optimization problems is that they appear in different forms and types. Some problems have to be solved for only one objective, some for multiple-conflicting objectives, some problems may be highly constrained, and some may have more than one optimal solution. When faced with such problems, a user first analyses the underlying problem and chooses a suitable algorithm to solve it. This is because an algorithm efficient for finding the sole optimum in a single-objective optimization problem cannot be adequately applied to find multiple optimal solutions present in another optimization problem.
Therefore, to handle different kinds of problems, a user needs to know different algorithms, each specialized in solving a particular class of optimization problem.

3.1. NSGA-II as omni-optimizer

To overcome the previously mentioned problem, [10] suggests an omni-optimizer, an algorithm capable of solving single- and multi-objective optimization problems. This algorithm is, in fact, NSGA-II with an added crowding variable distance.
When dealing with a single-objective problem, the procedure transforms to the following: the check for dominance between two solutions degenerates to finding
the better solution in objective value, and in case of a tie, a more diverse solution is chosen, based on crowding variable distance. The proposed algorithm uses two approaches for evaluation of crowding: crowding in objective space and crowding in design space. Therefore, if two solutions have an equal value of objective, their crowding distance in objective space will be the same, but then the solution with a higher value of crowding distance in the design space is preferred. The algorithm computes both crowding distances for every design, and

s modifikacijom kriterija ε-dominiranosti koji se koristi za klasifikaciju rješenja u različite fronte kod rangiranja. Budući da se s ciljem održavanja raznolikosti izvodi gomilanje projektnog i ciljnog prostora, predložena metoda može se nositi s jednomodalnim i višemodalnim problemima. Neki su problemi riješeni pomoću realnoga, a neki pomoću binarnoga kodiranja i uz vjerojatnost križanja od 0,8 i vjerojatnost mutacije od 1/N. Korišten je i prikladno mali ε parametar na osnovi dobrog poznavanja problema. Predloženi omni-optimizator pokazao se jednako dobrim, pa čak i boljim od NSGA-II za sve testne probleme.

the higher one is used for comparison.
When dealing with the multi-objective problem, the omni-optimization procedure is similar to NSGA-II, with the modification that the ε-dominance criterion is used for classifying solutions into different fronts in the ranking procedure. Since both objective space and variable space crowding is performed for maintaining diversity among solutions, the proposed omni-optimizer can cope with uni- and multi-modal problems. Some problems were solved with real and some with binary variable encoding, for which the crossover probability of 0,8 and mutation probability of 1/*N* was used. The ε parameter that was used was adequately small, based on known behaviour of the test problems. The proposed omni-optimizer behaved equally good or better than NSGA-II in all tested problems.

3.2. Vektorizacija i optimizacija (VOP)

Poboljšana koncepcija optimizacije strukture predložena je u [11], [12]. Autori su predložili prebacivanje ograničenja u dodatne ciljeve te izvršenja optimizacije prema izvornom problemu. Ova je transformacija nazvana vektorizacija i podijeljena je u dvije reprezentacije. U osnovi radi se o jednostavnom genetskom algoritmu s odabirom pomoću kola ruleta, jednostrukim križanjem, binarnim kodiranjem i 'bit-wise'mutacijom. Jedina prednost koju ima VOP je njegova funkcija dobrote koja omogućuje da se VOP koristi kao omni-optimizator slijedeći vektorizaciju izvornog problema. Koncept vektorizacije transformira izvorni višeciljni problem optimizacije strukture definiran kao:

3.2. Vectorizaton and optimization (VOP)

One concept for enhancing structural optimization has recently been proposed in [11], [12]. Authors suggest altering the constraints into additional objectives and optimizing them alongside the original. This transformation is addressed as vectorization, and is divided into two representations. Basically, it is a simple genetic algorithm with a weighted roulette wheel selection, single-point cross-over, binary encoding and bit-wise mutation. The only advanced feature that VOP possesses is its fitness function that allows VOP to be used as the omni-optimizer, following the vectorization of the original optimization problem. The concept of vectorization transforms original multi-objective optimization problem that is defined with the following:

$$\min_{\mathbf{x}\in\mathbf{X}}\left\{ \mathrm{f}_1(\mathbf{x}),...,\mathrm{f}_M(\mathbf{x}) \middle| \mathbf{g}_j(\mathbf{x}) \geq 0, j \in [1,J] \right\} \tag{11}$$

gdje je **X** skup mogućih alternativa definiranih između gornje i donje granice varijable. Isti se problem može napisati u vektoriziranom obliku kao:

where **X** is the set of all possible alternatives defined between the lower and upper bounds of the variables. The same problem can be written in the vectorized form:

$$\min_{\mathbf{x}\in\mathbf{X}}\left\{ \mathrm{f}_1(\mathbf{x}),...,\mathrm{f}_M(\mathbf{x}),\mathrm{f}_{M+1}(\mathbf{x}),...,\mathrm{f}_{M+j}(\mathbf{x}),...,\mathrm{f}_{M+J}(\mathbf{x}) \right\} \tag{12}$$

Ograničenja $\mathrm{g}_j(\mathbf{x})$ sada predstavljaju dodatne ciljeve $\{\mathrm{f}_{M+1}(\mathbf{x}),...,\mathrm{f}_{M+J}(\mathbf{x})\}$ i u gornjoj jednadžbi su pretvorena, pomoću dva pristupa, od kojih je prvi apsolutni prikaz:

The constraints $\mathrm{g}_j(\mathbf{x})$, which are now additional objectives $\{\mathrm{f}_{M+1}(\mathbf{x}),...,\mathrm{f}_{M+J}(\mathbf{x})\}$ in the equation above, are converted applying two approaches, "absolute" representation:

$$\mathrm{f}_{M+j}(\boldsymbol{x}) = \left|\mathrm{g}_j(\boldsymbol{x}), \forall j \in [1,J]\right. \tag{13}$$

a drugi Heaviside prikaz [13], kao:

and the Heaviside representation [13], given as:

$$\mathrm{f}_{M+j}(\boldsymbol{x}) = \begin{matrix} \mathrm{g}_j(\boldsymbol{x}), & \text{if } \mathrm{g}_j(\boldsymbol{x}) < 0 \\ 0, & \text{otherwise} \end{matrix}, \forall j \in [1,J] \tag{14}$$

Autori su u [12] pokazali prednosti vektorizacijskog pristupa u odnosu na skalarni pri optimizaciji strukture brzog trajekta. Bolji rezultati postignuti su apsolutnim predstavljanjem ograničenja. Također su pokazali da se istim algoritmom može rukovati s dodatnim funkcijama cilja čime problem prelazi u višeciljni, a u VOP-u se postiže jednostavnim postavljanjem broja ciljeva na M=2, u (12). Princip je prikazan na optimizaciji položaja vertikalnog težišta presjeka glavnog rebra i mase strukture, pri čemu je korištena sljedeća funkcija dobrote:

Authors [12] showed benefit when using vectorized approach in comparison with scalar version, optimizing the main frame of a fast ferry. Better results were achieved by using the "absolute" representation of constraints. They showed that the same algorithm is capable of dealing with additional objective function, transforming the problem from single- to multi-objective. This was rather easy for VOP since it meant simply increasing the number of objectives to M=2, (12). The principle was demonstrated on optimizing the vertical centre of gravity on the fast ferry's main frame, beside the weight of the structure, with the fitness function:

$$\varphi\left(\mathbf{x}^i\right)=\begin{cases}\max \mathrm{d}\left(\mathbf{x}\right)+\dfrac{1}{\mathrm{d}\left(\mathbf{x}^i\right)}, \text{ if } \mathbf{x}^i\in\hat{\mathbf{X}}\\ \max \mathrm{d}\left(\mathbf{x}\right)-\mathrm{d}\left(\mathbf{x}^i\right), \text{ otherwise}\end{cases} \tag{15}$$

gdje je $\hat{\mathbf{X}}$ član Pareto-fronte. Navedena funkcija dobrota rangira projekte unutar populacije na osnovi dobivene Pareto-optimalnosti i udaljenosti d do referentne točke **I** u ciljnom prostoru. U tom slučaju **I** je točka koja sadrži minimalnu vrijednost cilja unutar populacije. Budući da su u problem uključeni ciljevi različitoga fizikalnog značenja: masa, debljina, naprezanje, ciljni je prostor potrebno normalizirati kako bi se izbjegli nagli skokovi u intenzitetima raznih funkcija cilja u (11). Normalizacija se izvodi ograničenjem tog prostora unutar jediničnog intervala, gdje 0 predstavlja minimum, a 1 maksimum vrijednosti cilja za tekuću populaciju. Dakle $\mathbf{I}=\{0\}$, uz primjenu težinske euklidske funkcije kao mjere, udaljenost do **I** može se izračunati kao:

where $\hat{\mathbf{X}}$ stands for the Pareto-front member. This fitness ranks designs on the basis of the obtained Pareto optimality within a population and the distance d to the reference point **I** in an objective space. In this case, **I** is the point containing the minimum values of every objective within a population. As the problem deals with objectives of different physical meaning, such as weight, thickness, stress, etc., the objective space needs to be normalized to avoid pitfalls caused by large differences in the magnitudes of functions in Eq. (11). The normalization is performed by bounding this space within a unit interval, where 0 now presents the minimum of the objective for current population, and 1 is its maximum. Hence, $\mathbf{I}=\{0\}$, and if the weighted Euclidean function is applied as a measure, the distance to **I** is found as:

$$\begin{aligned}&\mathrm{d}\left(\boldsymbol{x}\right)=\left\{\sum_j w_j\left[\mathrm{f}_j^{\mathrm{n}}\left(\boldsymbol{x}\right)\right]^2+\sum_m w_m\left[\mathrm{f}_m^{\mathrm{n}}\left(\boldsymbol{x}\right)\right]^2\right\}^{1/2},\\ &\text{s.t. } 0<w_j<1;\ 0<w_m<1\\ &\sum_j w_j+\sum_m w_m=1\end{aligned} \tag{16}$$

4. ZAKLJUČAK

Evolucijski algoritmi, kao optimizacijski alati, kod većine inženjerskih problema dokazali su svoju valjanost, budući da uspješno upravljaju velikim brojem varijabli i ograničenja, a u posljednje vrijeme i funkcija cilja. Proces usavršavanja tih alata ostvaruje se hibridnim rješenjima koja spajaju najbolje od nekoliko različitih pristupa. Stvaranje jednog algoritma koji bi bio primjenjiv na sve probleme je nemoguće. Budući da je optimizacija sve popularnija, mnogobrojni istraživači su razvili različite algoritme. U ovom je radu prikazano nekoliko najvažnijih pristupa i metoda genetskog algoritma s više funkcija cilja s ciljem usmjeravanja stručnjaka koji ulaze u područje strukturne optimizacije na one metode koje su dokazale svoju valjanost u praktičnim primjenama.

4. CONCLUSION

Evolutionary algorithms have proven their worthiness in a great variety of practical problems, handling many variables, constraints and objectives. However, the need to improve them is always present. This has lately been done by hybridization, combining different approaches in order to get the best of them. To devise an algorithm which will be applicable to all problems is impossible. Since optimization is becoming more and more popular, numerous researchers develop special types of algorithms. In this review, for the purpose of introducing and directing the experts entering in the field of structural optimization, only a few of the most important approaches and multi-objective algorithms have been presented.

5. POPIS OZNAKA

projektni prostor	$\mathbf{X}$
funkcija cilja	$f(x)$
funkcija ograničenja	$g(x)$
normalizirana euklidska udaljenost	d_{ij}
gornja i donja granica varijable	x_p^u, x_p^l
zajednički parametar	σ_{share}
redukcijski faktor	m_i
zajednička funkcija	Sh
zajednička dobrota	f_k
udaljenost gustoće	$L_{distance}$
veličina populacije	N
dozvoljena tolerancija	ε_m
težinska funkcija cilja	$F(x)$
težinski koeficijent	w_m^x
funkcija dobrote	$f(x)$

5. LIST OF SYMBOLS

design space	
objective function	
constraints	
normalised Euclidian distance measure	
upper and lower limit of variable	
sharing parameter	
niche count	
sharing function	
shared fitness	
crowding distance	
size of population	
allowable tolerance	
weighted objective function	
weighted coefficient	
fitness function	

LITERATURA
REFERENCES

[1] Deb, K., Multi-*Objective Optimization using Evolut. Algorithms*, Wiley, Chichester,UK, 2001.

[2] Goldberg, D.E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Reading, MA: Addison-Wesley, 1989.

[3] Zitzler, E., Deb, K., Thiele, L., *Comparison of MOEA: Empirical Results*, Evolutionary Computation, Vol. 8, No. 2, 2000, pp. 173-195

[4] Deb, K., *Unveiling Innovative Design Principles by Means of Multiple Conflicting Objectives*, KanGAL Report Number 2002007, 2002.

[5] Deb, K., *Multi-objective Evolutionary Algorithms: Introducing Bias Among Pareto-optimal solutions*, KanGAL Report Number 99002, 1999.

[6] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., *A Fast and Elitist Multiobjective Genetic Algorithm – NSGA-II,* IEEE Transactions on Evolutionary Computation, 6/1, 2002. pp. 182-197

[7] Zitzler, E., Laumanns, M., Thiele, L., *SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization*, EMDOC, CIMNE, Barcelona, Spain, 2002.

[8] Deb, K., Mohan, M., Mishra, S., *A Fast Multi-objective Evolutionary Algorithm for Finding Well-Spread Pareto-Optimal Solutions*, KanGAL Report Number 2003002, 2003.

[9] Deb, K., Goel, T., *A Hybrid MOEA to Engineering Shape Design,* Lecture Notes in Computer Science, Springer, Berlin, 2001., p 385-399.

[10] Deb, K., Tiwari, S., *Omni-optimizer: A Procedure for Single and Multi-objective Optimization*, in Coello, C.A. *et al.* (Eds.): *EMO 2005*, Springer-Verlag, Berlin Heidelberg, 2005., pp. 47-61

[11] Klanac, A., Jelovica, J., *A concept of omni-optimization for ship structural design*,, *Advancements in marine structures*, Taylor & Francis, London, 2007. pp. 471-480

[12] Klanac, A., Jelovica, J., *Vectorization in the structural optimization of a fast ferry*, Brodogradnja, 58/1, 2007., pp. 11-17

[13] Osyczka, A., Krenich, S., Tamura, H., Goldberg, D., *A Bicriterion Approach to Constrained Optimization Problems using GA*EO – An Intern. Journal on the Internet, 2/1, 2000., pp. 43-54.

Primljeno / Received: 17.06.2009.

Prihvaćeno / Accepted: 15.10.2009.

Pregledni članak

Subject review

Adresa autora / Authors' address
Izv. prof. dr. sc. Albert Zamarin, dipl. ing.
V. asist. dr. sc. Marko Hadjina, dipl. ing.
Tehnički fakultet Sveučilišta u Rijeci
Vukovarska 58, 51000 Rijeka, HRVATSKA
zamarin@riteh.hr, hadjina@riteh.hr
Jasmin Jelovica, dipl. ing.
Helsinki University of Technology
Tietotie 1B, PL 5300 Helsinki, FINLAND
jasmin.jelovica@tkk.fi

Adresa autora / Authors' address
Assoc. Prof. D. Sc. Albert Zamarin, Naval Architect
Senior assist. D.Sc. Marko Hadjina, Naval Architect
Faculty of Engineering University of Rijeka
Vukovarska 58, 51000 Rijeka, CROATIA
zamarin@riteh.hr, hadjina@riteh.hr
M.Sc. Jasmin Jelovica, Naval Architect
Helsinki University of Technology
Tietotie 1B, PL 5300 Helsinki, FINLAND
jasmin.jelovica@tkk.fi